One Identity Safeguard for Privileged Sessions 6.13.0

REST API Reference Guide
Copyright 2022 One Identity LLC.
ALL RIGHTS RESERVED.

This guide contains proprietary information protected by copyright. The software described in this guide is furnished under a software license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of the applicable agreement. No part of this guide may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose other than the purchaser’s personal use without the written permission of One Identity LLC.

The information in this document is provided in connection with One Identity products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of One Identity LLC products. EXCEPT AS SET FORTH IN THE TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, ONE IDENTITY ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ONE IDENTITY BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ONE IDENTITY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. One Identity makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. One Identity does not make any commitment to update the information contained in this document.

If you have any questions regarding your potential use of this material, contact:

One Identity LLC.
Attn: LEGAL Dept
4 Polaris Way
Aliso Viejo, CA 92656

Refer to our Web site (http://www.OneIdentity.com) for regional and international office information.

Patents

One Identity is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at http://www.OneIdentity.com/legal/patents.aspx.

Trademarks

One Identity and the One Identity logo are trademarks and registered trademarks of One Identity LLC. in the U.S.A. and other countries. For a complete list of One Identity trademarks, please visit our website at www.OneIdentity.com/legal. All other trademarks are the property of their respective owners.

Legend

⚠️ WARNING: A WARNING icon highlights a potential risk of bodily injury or property damage, for which industry-standard safety precautions are advised. This icon is often associated with electrical hazards related to hardware.

⚠️ CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.

SPS REST API Reference Guide
Updated - 01 March 2022, 15:53
Version - 6.13.0
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naming options</td>
<td>86</td>
</tr>
<tr>
<td>Network addresses</td>
<td>88</td>
</tr>
<tr>
<td>Routing table</td>
<td>94</td>
</tr>
<tr>
<td>Local services of SPS</td>
<td>96</td>
</tr>
<tr>
<td>Local services: Web login for administrators</td>
<td>100</td>
</tr>
<tr>
<td>Local services: Web login for users</td>
<td>104</td>
</tr>
<tr>
<td>Local services: cluster interface</td>
<td>109</td>
</tr>
<tr>
<td>System backup policy</td>
<td>111</td>
</tr>
<tr>
<td>Encrypting system backup policy</td>
<td>113</td>
</tr>
<tr>
<td>Date and time</td>
<td>116</td>
</tr>
<tr>
<td>Date &amp; time</td>
<td>116</td>
</tr>
<tr>
<td>NTP servers</td>
<td>118</td>
</tr>
<tr>
<td>Timezone</td>
<td>121</td>
</tr>
<tr>
<td>Logs, monitoring and alerts</td>
<td>129</td>
</tr>
<tr>
<td>Management options</td>
<td>129</td>
</tr>
<tr>
<td>Web gateway authentication</td>
<td>133</td>
</tr>
<tr>
<td>Syslog server settings</td>
<td>135</td>
</tr>
<tr>
<td>Disk fill-up prevention</td>
<td>142</td>
</tr>
<tr>
<td>Mail settings</td>
<td>144</td>
</tr>
<tr>
<td>Health monitoring</td>
<td>152</td>
</tr>
<tr>
<td>SNMP settings</td>
<td>155</td>
</tr>
<tr>
<td>SNMP traps</td>
<td>157</td>
</tr>
<tr>
<td>Local services: access for SNMP agents</td>
<td>164</td>
</tr>
<tr>
<td>Alerting</td>
<td>171</td>
</tr>
<tr>
<td>System alerts</td>
<td>173</td>
</tr>
<tr>
<td>Traffic alerts</td>
<td>181</td>
</tr>
<tr>
<td>Trust stores</td>
<td>187</td>
</tr>
<tr>
<td>Enabling One Identity Safeguard Remote Access without Starling Join</td>
<td>194</td>
</tr>
<tr>
<td>Configuring Starling Join</td>
<td>198</td>
</tr>
<tr>
<td>Retrieving the status of services related to Starling Join/Unjoin</td>
<td>203</td>
</tr>
<tr>
<td><strong>User management and access control</strong></td>
<td>206</td>
</tr>
<tr>
<td>User management and access control</td>
<td>206</td>
</tr>
<tr>
<td>Login settings</td>
<td>208</td>
</tr>
<tr>
<td>Privileges of usergroups</td>
<td>211</td>
</tr>
<tr>
<td>Audit data access rules</td>
<td>216</td>
</tr>
</tbody>
</table>
Active sessions .......................................................................................................................... 220
Manage users and usergroups locally on SPS ........................................................................ 222
Manage usergroups locally on SPS ....................................................................................... 224
Manage users locally on SPS ................................................................................................ 228
Configuring LDAP servers .................................................................................................... 231
Configuring SPS login methods ............................................................................................ 244

Managing SPS ......................................................................................................................... 256
Troubleshooting options ........................................................................................................ 256
Internal certificates ................................................................................................................ 259
Passwords stored on SPS ........................................................................................................ 262
Private keys stored on SPS ...................................................................................................... 267
Private keys generated on SPS .............................................................................................. 271
Certificates stored on SPS ....................................................................................................... 287
Local services: enabling SSH access to the SPS host ............................................................. 292
Manage the SPS license .......................................................................................................... 298
Change contact information ................................................................................................. 302
Splunk integration .................................................................................................................. 304
Splunk integration .................................................................................................................. 309
Manage SPS clusters ............................................................................................................... 315
  Promote a SPS node to be the Central Management node in a new cluster ......................... 318
  Join node(s) to the cluster .................................................................................................... 319
  Query join status ................................................................................................................ 321
  Assign a role to a node ......................................................................................................... 323
  Query nodes ......................................................................................................................... 324
  Query one particular node ................................................................................................. 325
  Query the status of all nodes in the cluster ......................................................................... 326
  Query the status of one particular node ............................................................................ 332
Upload and enable a configuration synchronization plugin .................................................. 338
Disable a configuration synchronization plugin ...................................................................... 342
Configuration tools in SPS ...................................................................................................... 343
  Resolving hostnames to IP addresses ............................................................................... 345
  Testing LDAP server connection ....................................................................................... 347

General connection settings .................................................................................................. 354
Channel policy ......................................................................................................................... 354
Session alerts ................................................................. 750
Session events .................................................................. 754
Indexing sessions .......................................................... 759
Session audit trail downloads ......................................... 762
Local services: configuring the indexer ......................... 764
Indexer policies .............................................................. 772

**Reporting** ................................................................... 780
Reporting ......................................................................... 780
Reports ............................................................................ 782
Built-in subchapters ....................................................... 792
Pre-defined reports ......................................................... 796
Content subchapters ...................................................... 801
Custom subchapters ....................................................... 809
Connection statistics subchapters .................................. 814

**Health and maintenance** ............................................. 821
Monitor appliance health status ...................................... 821

**Advanced authentication and authorization** ................ 824
Usermapping policy ........................................................ 824
Plugins ............................................................................ 829
  Upload a plugin ............................................................ 831
  Delete a plugin ............................................................. 833
  Check the integrity of a plugin ...................................... 834
Authentication and authorization plugins ....................... 836
  Configuring Authentication and Authorization plugin instances 840
Credential store plugins ................................................ 844
Credential stores .......................................................... 847

**Completing the Welcome Wizard using REST** .............. 855
Completing the Welcome Wizard using REST ................. 855

**Enable and configure analytics using REST** ................. 862
Enable One Identity Safeguard for Privileged Analytics ...... 862
Configure One Identity Safeguard for Privileged Analytics ... 864

**About us** .................................................................... 870
Contacting us ................................................................ 870
Introduction

Starting with One Identity Safeguard for Privileged Sessions version 4 F2, certain parts and features of SPS can be configured using a RESTful API (Representational State Transfer Application Programming Interface). The REST server conforms to the Hypermedia as the Engine of Application State (HATEOAS).

The SPS REST API uses JSON over HTTPS. The REST server has a single entry point and all resources are available at paths (URLs) returned in the response for a request sent to the entry point. The only path that is guaranteed not to change is /api/authenticati1on. Every other path should be reached by navigating the links returned.

The SPS REST API allows you to create, read, update and delete (CRUD) the configuration resources of SPS.

In this tutorial, all examples are displayed with curl, but you can use any other HTTP client. In the examples it is assumed that the REST server is listening on the default HTTP port of SPS (443).

If you receive the "417 - Expectation Failed" error code when using curl, use curl with the - -http1.0 or the -H "Expect: " option.

Message format

Response headers

The following headers are included in every response. Other headers are specific to responses to specific requests.

- Allow: The SPS REST API allows you to create, read, update and delete (CRUD) the configuration resources of SPS. The value of the header lists the available actions for the resource or object.
- Content-Language: The language of the response. Currently only English (en) is supported.
- Content-Type: All messages are JavaScript Object Notation (JSON) objects. The SPS REST server sends all REST API responses in application/json format.
**Response body**

The response body contains JSON objects. These objects always contain a `meta` field with links to different parts of the REST service. In most cases, the following entries can be found in the `meta` object. Error messages are returned in the `error` element.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>meta</td>
<td></td>
<td>Top level element, contains links to different parts of the REST service</td>
<td></td>
</tr>
<tr>
<td>changes</td>
<td>string</td>
<td>Path to the transaction changelog</td>
<td>This value is always /api/transaction/changes. For details, see Reviewing the changelog of a transaction on page 35.</td>
</tr>
<tr>
<td>remaining_</td>
<td>integer</td>
<td>Time left until the session times out in seconds</td>
<td>SPS closes idle sessions after a period of inactivity. This value shows the number of seconds left until the timeout. For details on setting the session timeout, see Web interface.</td>
</tr>
<tr>
<td>seconds</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>href</td>
<td>string (relative path)</td>
<td>Path of the resource that returned the response. When creating a new object, this is the URL of the created object.</td>
<td>For example, /api/authentication</td>
</tr>
<tr>
<td>parent</td>
<td>string (relative path)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>next</td>
<td>string (relative path)</td>
<td>Path of the next sibling of the current resource</td>
<td>For example, /api/configuration</td>
</tr>
<tr>
<td>prev</td>
<td>string (relative path)</td>
<td>Path of the previous sibling of the current</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>first</td>
<td>string (relative path)</td>
<td>Path of the first sibling of the current resource</td>
<td></td>
</tr>
<tr>
<td>last</td>
<td>string (relative path)</td>
<td>Path of the last sibling of the current resource</td>
<td></td>
</tr>
<tr>
<td>transaction</td>
<td>string ( /api/transaction)</td>
<td>The endpoint of the transaction log</td>
<td>For details on how SPS handles transactions, see How to configure SPS using REST on page 14.</td>
</tr>
<tr>
<td>items</td>
<td>list of JSON objects</td>
<td>List of endpoints (objects) available from the current endpoint</td>
<td>Each object in the list contains a key and a meta object for the endpoint. For example:</td>
</tr>
</tbody>
</table>

```json
{
    "meta": {
        "href": "/api/ssh-host-keys",
        "parent": "/api"
    },
    "items": [
        {
            "key": "ssh-rsa-10.10.100.1:22",
            "meta": {
                "href": "/api/ssh-host-keys/ssh-rsa-10.10.100.1:22"
            }
        },
        {
            "key": "ssh-rsa-10.10.20.35:22",
            "meta": {
                "href": "/api/ssh-host-keys/ssh-rsa-10.10.20.35:22"
            }
        }
    ]
}
```
For example:

```json
{
    "meta": {
        "href": "/api",
        "next": "/api/configuration"
    }
}
```

### Error responses

All error responses are JSON objects with the following keys.

- **meta**: JSON object containing navigation links. For details, see Message format on page 10.
- **error**: JSON object containing information about the error.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td></td>
<td>Top level element, contains links to different parts of the REST service</td>
<td>For example, Unauthenticated, or NodeNotFound. For a complete list, see Application level error codes on page 37.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The type of the error that occurred</td>
<td>For example, Unauthenticated, or NodeNotFound. For a complete list, see Application level error codes on page 37.</td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>A textual message that describes the error</td>
<td>For example, Unable to locate the requested path.</td>
</tr>
<tr>
<td>details</td>
<td>JSON object</td>
<td>List of additional information about the error (for example:</td>
<td></td>
</tr>
</tbody>
</table>
The following is a complete error response.

```json
{
    "error": {
        "details": {
            "path": "password"
        },
        "message": "Syntax error: The given password is too short. Minimum password length: 10",
        "type": "SyntacticError"
    },
    "meta": {
        "href": "/api/configuration/no/such/path",
        "parent": "/api/configuration"
    }
}
```

**How to configure SPS using REST**

The SPS REST server uses a transactional model for configuration management.

Certain endpoints require transaction for sending/receiving POST, PUT, GET and so on requests. A transaction creates a "snapshot" of the configuration and will perform all changes on that snapshot. For example, when using transaction in case of a GET request, your requests will be performed on a consistent state of the configuration as opposed to a configuration that might change in the meantime due to user interaction.

The following endpoints require transaction:

- https://<IP-address-of-SPS>/api/configuration/
- https://<IP-address-of-SPS>/api/cluster/
- https://<IP-address-of-SPS>/api/user/password/
- https://<IP-address-of-SPS>/api/upload/
Modifying the configuration has the following main steps. The steps are explained in detail in later sections of the tutorial. You find a simple transaction with detailed requests and responses in How to configure SPS using REST: a sample transaction on page 16.

1. Authenticate on the SPS REST server, and receive a session_id. For details, see Authenticate to the SPS REST API on page 19.

2. Open a transaction. This transaction will collect the changes and modifications you do, compared to the SPS configuration that is active at the time of opening the transaction. It is similar to a shopping cart, where your modifications are the items in the cart. For details, see Open a transaction on page 29.
   Opening a transaction locks the configuration of SPS similarly to accessing SPS from the web interface. At the open transaction stage this step is optional.

3. Change and modify the configuration of SPS as you need. Note that to modify the configuration, you must have the required privileges. For details, see "Managing user rights and usergroups" in the Administration Guide. For details on navigating and modifying the configuration of SPS, see Navigating the configuration of SPS on page 40 and Modifying the configuration of SPS on page 43.

4. Commit your transaction to submit your changes to SPS (this is similar to clicking Checkout in a web shop). For details, see Commit a transaction on page 31.
   If the Users & Access Control > Settings > Accounting settings > Require commit log option is selected in the SPS web interface, you must include a commit message (a message object) in the request. This message will be visible on the Users & Access Control > Configuration History page of the SPS web interface. Note that on the Users & Access Control > Configuration History page, changes performed using the REST API are listed as changes to the REST server/REST configuration page.
   If you do not want to commit your changes, and would like to restart with the original configuration of SPS, you can simply delete the transaction. This is similar to the rollback transaction in SQL. If your session times out, your transaction is deleted automatically. For details, see Delete a transaction on page 33.
   Note that committing a transaction locks the configuration of SPS similarly to accessing SPS from the web interface. For more information, see "Multiple users and locking" in the Administration Guide.

5. SPS checks and validates the changes in your transaction. If other users have changed the configuration of SPS since you opened the transaction, SPS tries to merge your changes to the current configuration.

6. If your changes are valid, SPS applies them and you have successfully changed the configuration of SPS. Otherwise, the REST server returns an error response.
How to configure SPS using REST: a sample transaction

This procedure shows a sample transaction with detailed requests and responses. For details on the transaction model, see How to configure SPS using REST on page 14.

1. Authenticate on the SPS REST server, and receive a session_id.

```
curl --basic --user <username>:<password> --cookie-jar cookies --insecure https://<SPS-IP-address>/api/authentication
```

Response status: 200
--- BEGIN RESPONSE BODY ---
```
{
  "meta": {
    "href": "/api",
    "next": "/api",
    "transaction": "/api/transaction"
  }
}
--- END RESPONSE BODY ---

2. Open a transaction.

```
curl --data "" --cookie cookies --insecure -X POST https://<IP-address-of-SPS>/api/transaction
```

Response status: 200
--- BEGIN RESPONSE BODY ---
```
{
  "meta": {
    "href": "/api/transaction",
    "parent": "/api"
  }
}
--- END RESPONSE BODY ---

3. Retrieve a resource. The following example shows the resource corresponding to the Users & Access Control > Settings page of the SPS web interface.

```
curl --cookie cookies --insecure https://<IP-address-of-SPS>/api/configuration/aaa/settings
```

Response status: 200
--- BEGIN RESPONSE BODY ---
```
4. Change and modify the configuration of SPS as you need. The following example configures SPS to check the password strength of the passwords for users of the SPS web interface.

```json
# Body of the PUT request. You can read it from a file, for example, body.json
{
    "backend": {
        "cracklib_enabled": true,
        "expiration_days": 0,
        "minimum_password_strength": "good",
        "remember_previous_passwords": 10,
        "selection": "local"
    },
    "method": {
        "selection": "passwd"
    },
    "require_commitlog": false
}
```

# Command to use
curl -H "Content-Type: application/json" -d @body.json --cookie cookies --
insecure -X PUT https://<IP-address-of-SPS>/api/configuration/aaa/settings

Response status: 200
--- BEGIN RESPONSE BODY ---
{
   "meta": {
      "first": "/api/configuration/aaa/settings",
      "href": "/api/configuration/aaa/settings",
      "last": "/api/configuration/aaa/settings",
      "next": null,
      "parent": "/api/configuration/aaa",
      "previous": null,
      "transaction": "/api/transaction"
   }
}
--- END RESPONSE BODY ---

5. Commit your transaction to submit your changes to SPS.

```
curl -H "Content-Type: application/json" -d '{"status": "commit","message": "My commit message"}' --cookie cookies --insecure -X PUT https://<IP-address-of-SPS>/api/transaction
```

Response status: 200
--- BEGIN RESPONSE BODY ---
{
   "meta": {
      "href": "/api/transaction",
      "parent": "/api"
   }
}
--- END RESPONSE BODY ---

If the Users & Access Control > Settings > Accounting settings > Require commit log option is selected in the SPS web interface, you must include a commit message (a message object) in the request. This message will be visible on the Users & Access Control > Configuration History page of the SPS web interface. Note that on the Users & Access Control > Configuration History page, changes performed using the REST API are listed as changes to the REST server/REST configuration page.

6. If your changes are valid, SPS applies them and you have successfully changed the configuration of SPS. Otherwise, the REST server returns an error response.
Using the SPS REST API

The following sections give you a general overview of how the SPS REST API works.

Authenticate to the SPS REST API

Prerequisites:

- The REST server must permit password authentication to the SPS web interface. If only certificate-based authentication is permitted, see Authenticate to the SPS REST API using X.509 certificate on page 22.
  
  To check the permitted authentication method, query the /api/authentication/types endpoint.
  
  - If the types field of the response includes the x509 object, certificate-based authentication is permitted.
  - If it includes only the basic object, password authentication is permitted.
  - If it includes both fields, then certificate-based authentication is permitted for the users, but the admin user can authenticate with password as well. Note that in this case, SPS assumes that the admin user will authenticate with a password, and expects password-authentication on the /api/authentication endpoint. To authenticate with a certificate, use the /api/authentication?type=x509 endpoint.
  
- You can access the REST server on the same IP address and port that you use to access the SPS web interface. Note that management (administrator) access must be enabled on the interface. For details on configuring management access, see "Configuring user and administrator login addresses" in the Administration Guide.
  
- For the user to have full access over the SPS REST API, they must have the REST server privilege. The user privileges on the web UI and REST API are synchronized. For example, if the user has the ICA Control / Connections privilege then they can access this page on the web UI and also the /api/configuration/ica/connections endpoint on the REST API. For details, see "Modifying group privileges" in the Administration Guide.
Note that the system time of SPS and the client must be synchronized. The authentication cookie is valid for twenty minutes, and both SPS and most REST clients validate this. As a result, if the system time of SPS and the client is significantly different from each other, the authentication seems to be successful, but you will not be able to actually access SPS. (If the session_id is missing from the cookies file, check the system clocks.)

Make sure that user credentials are encoded in UTF-8.

The authentication procedure:

1. To authenticate on the SPS REST server, send a GET request over HTTPS using the basic HTTP authentication method, including your username and password to the /api/authentication resource.

2. If the authentication is successful, the server returns the 200 status code, and a meta object in the response body. Also, the HTTP headers of the response include an HTTP cookie named session_id. This cookie is used to identify the client in every subsequent HTTP request.

3. For every subsequent request, include the session_id header with the value of the received session ID. For example:

   session_id 087658d7e30c252b015dd761b6f7ccbd5

4. The authenticated session times out after 20 minutes of inactivity.

   Note that the system time of SPS and the client must be synchronized. The authentication cookie is valid for twenty minutes, and both SPS and most REST clients validate this. As a result, if the system time of SPS and the client is significantly different from each other, the authentication seems to be successful, but you will not be able to actually access SPS. (If the session_id is missing from the cookies file, check the system clocks.)

URL

GET https://<IP-address-of-SPS>/api/authentication

Headers

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>Contains the username and password of the user</td>
<td>Required</td>
<td>The string Basic followed by the username:password encoded using the RFC2045-MIME. For example, Basic YWtaW46YQ==</td>
</tr>
</tbody>
</table>
Sample request

Example: Authenticate to the SPS REST server using curl

The following command authenticates on SPS using the `curl` HTTP client. The `--insecure` option used in the example is used to bypass verifying the certificate of SPS. (Alternatively, you can use the `--cacert` option or the `CURL_CA_BUNDLE` environment variable to specify the Certificate Authority to verify the certificate of SPS. For details, see the `curl` man page).

When using the REST API in production environments, make sure to download the CA certificate of SPS from Basic Settings > Management > SSL certificate > CA X.509 certificate, and validate the certificate of SPS using this CA certificate, or with the CA certificate you used to sign the Server X.509 certificate of SPS.

```bash
curl --basic --user <username>:<password> --cookie-jar cookies --insecure https://<SPS-IP-address>/api/authentication
```

The cookie containing the session ID is also received (you can display it for example with the `tail -l cookies` command).

```
localhost FALSE FALSE 1395325830 session_id 600dc0ffeec0ffeec0ffeec0ffeec0ffeec0ffee
```

The following command retrieves the configuration of SPS, using the session ID received during the authentication.

```bash
curl --cookie cookies --insecure https://<IP-address-of-SPS>/api/configuration
```

Response

The following is a sample response received if the authentication is successful.

For details of the `meta` object, see Message format on page 10.

```
{
    "meta": {
        "href": "/api",
        "next": "/api",
        "transaction": "/api/transaction"
    }
}
```
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>Successful authentication</td>
</tr>
<tr>
<td>400</td>
<td>InvalidAuthenticationRequest</td>
<td>Unable to authenticate: no valid credentials found.</td>
</tr>
<tr>
<td>401</td>
<td>AuthenticationFailure</td>
<td>Authenticating the user with the given credentials has failed.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>You tried using an unsupported HTTP method. Use the GET method for authentication.</td>
</tr>
</tbody>
</table>

Authenticate to the SPS REST API using X.509 certificate

Prerequisites:

- The REST server must permit certificate authentication to the SPS web interface. If only password-based authentication is permitted, see Authenticate to the SPS REST API on page 19.

To check the permitted authentication method, query the /api/authentication/types endpoint.

- If the types field of the response includes the x509 object, certificate-based authentication is permitted.
- If it includes only the basic object, password authentication is permitted.
- If it includes both fields, then certificate-based authentication is permitted for the users, but the admin user can authenticate with password as well. Note that in this case, SPS assumes that the admin user will authenticate with a password, and expects password-authentication on the /api/authentication endpoint. To authenticate with a certificate, use the /api/authentication?type=x509 endpoint.

- You can access the REST server on the same IP address and port that you use to access the SPS web interface. Note that management (administrator) access must be enabled on the interface. For details on configuring management access, see "Configuring user and administrator login addresses" in the Administration Guide.

- For the user to have full access over the SPS REST API, they must have the REST server privilege. The user privileges on the web UI and REST API are synchronized. For example, if the user has the ICA Control / Connections privilege then they can...
access this page on the web UI and also the /api/configuration/ica/connections endpoint on the REST API. For details, see "Modifying group privileges" in the Administration Guide.

- Note that the system time of SPS and the client must be synchronized. The authentication cookie is valid for twenty minutes, and both SPS and most REST clients validate this. As a result, if the system time of SPS and the client is significantly different from each other, the authentication seems to be successful, but you will not be able to actually access SPS. (If the session_id is missing from the cookies file, check the system clocks.)

- Make sure that user credentials are encoded in UTF-8.

The authentication procedure:

1. To authenticate on the SPS REST server, send an HTTPS GET request, including your certificate to the /api/authentication?type=x509 resource. The certificate must be signed by the Trusted CA certificate that is configured on the Users & Access Control > Settings > X.509 > AUTHENTICATION CA field of the SPS web interface, or the /api/configuration/aaa/settings resource.

2. If the authentication is successful, the server responds with an HTTP 302 redirect to the /api/ resource, and also, sets an HTTP cookie named session_id. This cookie is used to identify the client in every subsequent HTTP request. The response body also includes a meta object.

3. For every subsequent request, include the session_id header with the value of the received session ID. For example:

   session_id 087658d7e30cdc2552b015dd761b6f7ccb25bbd5

4. The authenticated session times out after 20 minutes of inactivity.

   Note that the system time of SPS and the client must be synchronized. The authentication cookie is valid for twenty minutes, and both SPS and most REST clients validate this. As a result, if the system time of SPS and the client is significantly different from each other, the authentication seems to be successful, but you will not be able to actually access SPS. (If the session_id is missing from the cookies file, check the system clocks.)

URL

GET https:<IP-address-of-SPS>/api/authentication

Headers

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>Contains the username and</td>
<td>Required</td>
<td>The string Basic followed by the username:password encoded using the</td>
</tr>
</tbody>
</table>
## Sample request

### Example: Authenticate to the SPS REST server using curl

The following command authenticates on SPS using the curl HTTP client. The `--insecure` option used in the example is used to bypass verifying the certificate of SPS. (Alternatively, you can use the `--cacert` option or the `CURL_CA_BUNDLE` environment variable to specify the Certificate Authority to verify the certificate of SPS. For details, see the [curl man page](https://curl.se/docs/manpage.html)).

When using the REST API in production environments, make sure to download the CA certificate of SPS from **Basic Settings > Management > SSL certificate > CA X.509 certificate**, and validate the certificate of SPS using this CA certificate, or with the CA certificate you used to sign the **Server X.509 certificate** of SPS.

```
curl --basic --user <username>:<password> --cookie-jar cookies --insecure https://<SPS-IP-address>/api/authentication
```

The cookie containing the session ID is also received (you can display it for example with the `tail -l cookies` command).

```
localhost FALSE / FALSE 1395325830 session_id 600dc0ffeecc0ffeecc0ffeecc0ffeecc
```

The following command retrieves the configuration of SPS, using the session ID received during the authentication.

```
curl --cookie cookies --insecure https://<IP-address-of-SPS>/api/configuration
```

### Response

The following is a sample response received if the authentication is successful. For details of the meta object, see [Message format](#) on page 10.
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>302</td>
<td>OK</td>
<td>Successful authentication. If the authentication is successful, the server returns the 302 status code, and a meta object in the response body. Also, the HTTP headers of the response include an HTTP cookie named session_id. This cookie is used to identify the client in every subsequent HTTP request. The Location header in the response is /api/.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidAuthenticationRequest</td>
<td>Unable to authenticate: no valid credentials found. SPS returns this message if password fallback is enabled for the admin user, but the admin tries to authenticate with a certificate on the /api/authentication endpoint. To authenticate with a certificate, use the /api/authentication?type=x509 endpoint.</td>
</tr>
<tr>
<td>401</td>
<td>AuthenticationFailure</td>
<td>Authenticating the user with the given credentials has failed.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>You tried using an unsupported HTTP method. Use the GET method for authentication.</td>
</tr>
</tbody>
</table>

Retrieve user information

You can check the endpoints and methods that a particular user is authorized to access.

Prerequisites:

- The user must be logged in.
URL

GET https:<IP-address-of-SPS>/api/user_info

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d8030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command retrieves user information from SPS about the logged in user, using the session ID received during the authentication.

curl --cookie cookies https://<IP-address-of-SPS>/api/user_info

This information is also available on the /api/user/info and /api/userinfo endpoints.

Response

The following is a sample response received if the request to retrieve user information is successful.

For details of the meta object, see Message format on page 10.

```
{
    "user": {
        "name": "admin",
    }
    "endpoints": [
        {
            "methods": [
                "DELETE",
                "GET",
                "POST",
                "PUT"
            ]
        }
    ]
```
Element | Type | Description
--- | --- | ---
user | Top-level element, contains the details of the user whose access rights information has been retrieved.

name | string | The username of the logged-in user whose information has been retrieved.

endpoints | Top-level element, contains the details of the endpoints that the user is authorized to access.

methods | string | The methods that user is authorized to use, and the permitted HTTP method (for example, GET, POST) for each endpoint. This information is also available on the /api/endpoints endpoint.

url | string | The resource that the user is authorized to access.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>User information has been retrieved successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
</tbody>
</table>
Checking the transaction status

Before changing anything in the configuration of SPS, you must POST a request to open a transaction.

- For details about the transaction model of SPS see How to configure SPS using REST on page 14.
- To check the configuration changes you made in the transaction, see Reviewing the changelog of a transaction on page 35.

**URL**

```
GET https:<IP-address-of-SPS>/api/transaction/
```

**Sample request**

The following command retrieves the transaction status of SPS, using the session ID received during the authentication.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/transaction
```

**Response**

The following is a sample response received if opening the transaction is successful. For details of the meta object, see Message format on page 10.

```
{
    "key": "transaction",
    "meta": {
        "href": "/api/transaction",
        "parent": "/api"
    },
    "transaction": {
        "status": "closed"
    }
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>transaction</td>
<td>Top level</td>
<td>Top level element, contains the details of the current transaction</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
<td>The status of the current transaction. By default, or after a successful</td>
</tr>
<tr>
<td></td>
<td></td>
<td>commit it is closed. After successfully opening a transaction, it is open.</td>
</tr>
</tbody>
</table>

## Open a transaction

The REST API of SPS manages the changes of the configuration in transaction. You can open a transaction with a POST request, but the first change of the configuration will open the transaction automatically. For details about the transaction model of SPS see [How to configure SPS using REST](#) on page 14.

### URL

POST https:<IP-address-of-SPS>/api/transaction

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
</table>
| session_id  | Contains the authentication token of the user | Required | The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see [Authenticate to the SPS REST API](#) on page 19.

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### POST body

Note that you must:

- either send an empty body in the POST request,
- or include a Content-Length: 0 header.

Otherwise the SPS REST server returns a 411 - Length Required error.
Sample request

The following command opens a new transaction on SPS, using the session ID received during the authentication.

```
curl -X POST --data "" --cookie cookies https://<IP-address-of-SPS>/api/transaction
```

Response

The following is a sample response received if opening the transaction is successful. For details of the `meta` object, see Message format on page 10.

```
{
    "meta": {
        "href": "/api/transaction",
        "parent": "/api"
    }
}
```

After opening a transaction successfully, the transaction status changes to `open`.

```
{
    "body": {
        "commit_message": "optional|required",
        "status": "open"
    },
    "key": "transaction",
    "meta": {
        "changes": "/api/transaction/changes",
        "href": "/api/transaction",
        "parent": "/api"
    }
}
```

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>Transaction opened successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires author-</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------</td>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>You tried using an unsupported HTTP method. Use the POST method to open a transaction.</td>
</tr>
<tr>
<td>409</td>
<td>WebGuiOrRpcApiConfigInProgress</td>
<td>The configuration of SPS is locked. Opening a new transaction is not allowed while another user is modifying configuration through interfaces other than the REST API. For example, web GUI, console, and so on.</td>
</tr>
<tr>
<td>411</td>
<td>UnsupportedMethod</td>
<td>You must send a body (which can be empty) in this POST request, otherwise the SPS REST server returns a 411 - Length Required error.</td>
</tr>
</tbody>
</table>

**Commit a transaction**

To submit your changes to SPS, you have to commit the transaction by using a PUT request with a JSON object. For details about the transaction model of SPS, see How to configure SPS using REST on page 14.

Note that committing a transaction locks the configuration of SPS similarly to accessing SPS from the web interface. For more information, see "Multiple users and locking" in the Administration Guide.

**URL**

```
PUT https:<IP-address-of-SPS>/api/transaction
```
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

PUT body

The PUT request must include the following JSON object in its body.

```json
{
    "status": "commit"
}
```

If the **Users & Access Control > Settings > Accounting settings > Require commit log** option is selected in the SPS web interface, you must include a commit message (a message object) in the request. This message will be visible on the **Users & Access Control > Configuration History** page of the SPS web interface. Note that on the **Users & Access Control > Configuration History** page, changes performed using the REST API are listed as changes to the **REST server/REST configuration** page.

```json
{
    "status": "commit",
    "message": "My commit message"
}
```

Sample request

The following command commits a transaction to SPS, using the session ID received during the authentication.

```
curl -d '{"status": "commit","message": "My commit message"}' --cookie cookies -X PUT https://<IP-address-of-SPS>/api/transaction
```
Response

The following is a sample response received if committing the transaction is successful. For details of the meta object, see Message format on page 10.

After a successful commit, the transaction status changes to closed. To make other changes, you have to open a new transaction.

```json
{
  "meta": {
    "href": "/api/transaction",
    "parent": "/api"
  },
  "key": "transaction",
  "transaction": {
    "status": "closed"
  }
}
```

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>Transaction committed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>You tried using an unsupported HTTP method. Use the PUT method to commit a transaction.</td>
</tr>
</tbody>
</table>

Delete a transaction

To delete your changes, you have to delete the transaction. This is similar to the rollback transaction in SQL. For details about the transaction model of SPS, see How to configure SPS using REST on page 14. Deleting the transaction also deletes the configuration lock of SPS.
**URL**

DELETE https://<IP-address-of-SPS>/api/transaction

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d830e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command deletes a transaction, reverting the configuration to the state it was in when the transaction was opened, or to the current configuration available on SPS (if another user has modified it since you opened the transaction).

```
curl --cookie cookies -X DELETE https://<IP-address-of-SPS>/api/transaction
```

**Response**

The following is a sample response received if deleting the transaction is successful. For details of the meta object, see **Message format** on page 10.

```
{
    "meta": {
        "href": "/api/transaction",
        "parent": "/api"
    }
}
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see **Application level error codes** on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>Transaction deleted successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>You tried using an unsupported HTTP method. Use the DELETE method to reset a transaction.</td>
</tr>
</tbody>
</table>

## Reviewing the changelog of a transaction

To review your changes, retrieve the changelog of the transaction. For details about the transaction model of SPS, see [How to configure SPS using REST](page 14).

### URL

GET https://<IP-address-of-SPS>/api/transaction/changes

### Cookies

<table>
<thead>
<tr>
<th>Cookie Name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see [Authenticate to the SPS REST API](page 19).</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command retrieves the changelog of the transaction.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/transaction/changes
```

Response

The response contains the list of changes performed in the transaction, as list of JSON objects. Every change has a type and a path, other elements depend on the type of the transaction. For example, when you delete an object, the changelog includes the deleted object in the `old_value` field.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>new_order</code></td>
<td>list</td>
<td>The new order of a list after the change. This field is available for reorder transactions.</td>
</tr>
<tr>
<td><code>new_value</code></td>
<td>string or JSON object</td>
<td>The value of the object after the change. For example, the new value of a parameter.</td>
</tr>
<tr>
<td><code>old_order</code></td>
<td>string or JSON object</td>
<td>The order of a list before the change. This field is available for reorder transactions.</td>
</tr>
<tr>
<td><code>old_value</code></td>
<td>string or JSON object</td>
<td>The value of the object before the change. For example, the value of a deleted object.</td>
</tr>
<tr>
<td><code>path</code></td>
<td>string</td>
<td>Path of the changed endpoint or object.</td>
</tr>
<tr>
<td><code>type</code></td>
<td>string</td>
<td>The type of the change. One of: create, delete, reorder, replace</td>
</tr>
</tbody>
</table>

The following is a sample response received if the changelog is empty.

```
{
  "meta": {
    "href": "/api/transaction/changes",
    "parent": "/api/transaction",
    "transaction": "/api/transaction"
  },
  "changes": []
}
```

The following is a sample changelog received after deleting a Channel policy.

```
{
  "meta": {
    "href": "/api/transaction/changes",
    "parent": "/api/transaction",
    "transaction": "/api/transaction"
  }
}
```
}, "changes": [
  {
    "old_value": {
      "name": "deny",
      "rules": []
    },
    "path": "/api/configuration/ssh/channel_policies/94615110156697e93121f3",
    "type": "delete"
  }
]}

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>Transaction changelog has been retrieved successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>You tried using an unsupported HTTP method. Use the GET method to retrieve the changelog a transaction.</td>
</tr>
</tbody>
</table>

**Application level error codes**

In addition to the standard HTTP status codes, in certain cases, the SPS REST server provides additional information in the response about the error. The following table contains a brief description of such errors. For more details, see the error object in the response body.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>InvalidRequestBody</td>
<td>The request body sent by the user has an invalid format. This may be an error with the encoding or the body is not a properly encoded JSON value.</td>
</tr>
<tr>
<td>400</td>
<td>ConfigTreeNotAvailable</td>
<td>An error occurred while preparing the configuration tree for the REST API.</td>
</tr>
<tr>
<td>400</td>
<td>SyntacticError</td>
<td>A value to be set is not accepted syntactically. The details section contains the path that was found to be invalid.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidPath</td>
<td>The path provided by the client contains a syntax error. Path components are restricted to contain only lowercase alphanumeric characters, the dash (-) and the underscore (_) characters. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>400</td>
<td>SemanticError</td>
<td>The configuration contains semantic errors, inconsistencies or other problems that would put the system into an unreliable state if the configuration had been applied. The details section contains the errors that were found in the configuration.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>401</td>
<td>AuthenticationFailure</td>
<td>Authenticating the user with the given credentials has failed.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NodeNotFound</td>
<td>The requested endpoint does not exist in the configuration. The details section contains the path that you tried to access, but could not be retrieved.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>404</td>
<td>NodeNotAvailable</td>
<td>The requested endpoint exists in the configuration, however, it is not available directly. The details section contains the path that you tried to access, but could not be retrieved.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>An attempt was made to change a configuration subtree in an unsupported way. The method <code>&lt;method&gt;</code> is not allowed for this node.</td>
</tr>
<tr>
<td>409</td>
<td>MidAirCollisionSemanticError</td>
<td>This error occurs when the configuration has been changed by another client between starting and committing a transaction, and the changes in the transaction would interfere semantically with the changes of that other user. The recommended strategy to resolve this error is to review the changes made in the failing transaction, then roll it back, start a new transaction, redo the changes, and finally, commit the new transaction.</td>
</tr>
<tr>
<td>409</td>
<td>WebGuiOrRpcApiConfigInProgress</td>
<td>The configuration of SPS is locked. Opening a new transaction is not allowed while another user is modifying configuration through interfaces other than the REST API. For example, web GUI, console, and so on.</td>
</tr>
<tr>
<td>409</td>
<td>MidAirCollision</td>
<td>This error occurs when the configuration has been changed by another client between starting and committing a transaction, and the changes in the transaction would overwrite or interfere with the changes of that other user. The recommended strategy to resolve this error is to review the changes made in the failing transaction, then roll it back, start a new transaction, redo the changes, and finally, commit the new transaction.</td>
</tr>
<tr>
<td>409</td>
<td>NoTransaction</td>
<td>An attempt was made to change the configuration when no transaction was open.</td>
</tr>
<tr>
<td>409</td>
<td>DoubleTransaction</td>
<td>This error is returned when the client attempts to open a transaction while</td>
</tr>
</tbody>
</table>

Using the SPS REST API
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>417</td>
<td>Expectation Failed</td>
<td>If you receive the &quot;417 - Expectation Failed&quot; error code when using curl, use curl with the --http1.0 or the -H &quot;Expect:&quot; option.</td>
</tr>
<tr>
<td>500</td>
<td>CommitMessageMissing</td>
<td>This error is returned when a commit message is required for committing a transaction, but it was not provided in the commit request.</td>
</tr>
<tr>
<td>500</td>
<td>TransactionCommitError</td>
<td>Unexpected internal errors during committing a transaction are interpreted as TransactionCommitError.</td>
</tr>
<tr>
<td>500</td>
<td>AuthorizationError</td>
<td>The request could not be authorized due to an unexpected internal error.</td>
</tr>
</tbody>
</table>

**Navigating the configuration of SPS**

The main starting point of navigating the SPS configuration using REST is the https:<IP-address-of-SPS>/api/configuration endpoint. If you query this endpoint, the response contains a list of other endpoints that you can follow to list the various resources of SPS, or to list the objects of a specific resource. For example, https:<IP-address-of-SPS>/api/configuration/rdp lists resources related to controlling the Remote Desktop (RDP) protocol, while https:<IP-address-of-SPS>/api/configuration/rdp/channel_policies lists the available RDP Channel Policies.

Note that when you want to create an object that references another object (for example, a Channel Policy that uses a Content Policy), then the referenced object (in this case, the Content Policy) must already exist. For details, see Create a new object on page 45.

To modify or delete an object, you need the ID of the object. For details, see Change an object on page 49 and Delete an object on page 43.

The following is a sample command to query the https:<IP-address-of-SPS>/api/configuration endpoint, and a sample response.

```bash
curl --cookie cookies https:<IP-address-of-SPS>/api/configuration
```

Response status: 200
--- BEGIN RESPONSE BODY ---
```
{
  "meta": {
    "first": "/api/configuration",
```
"href": "/api/configuration",
"last": "/api/configuration",
"next": null,
"parent": null,
"previous": null,
"transaction": "/api/transaction"
},
"items": [
{
  "key": "aaa",
  "meta": {
    "href": "/api/configuration/aaa"
  }
},
{
  "key": "alerting",
  "meta": {
    "href": "/api/configuration/alerting"
  }
},
{
  "key": "datetime",
  "meta": {
    "href": "/api/configuration/datetime"
  }
},
{
  "key": "http",
  "meta": {
    "href": "/api/configuration/http"
  }
},
{
  "key": "ica",
  "meta": {
    "href": "/api/configuration/ica"
  }
},
{
  "key": "local_services",
  "meta": {
    "href": "/api/configuration/local_services"
  }
},
{
  "key": "management",
  "meta": {
    "href": "/api/configuration/management"
"network",
"passwords",
"plugins",
"policies",
"private_keys",
"rdp",
"reporting",
"ssh"
Modifying the configuration of SPS

The following sections describe deleting, creating and changing objects.

Delete an object

To delete a configuration object (for example, a policy), use a DELETE request with the ID of the object as the key.
You cannot delete policies or objects that are used in other policies (for example, you cannot delete a Time policy that is used in a Channel policy).

To delete an element of a list (for example, a user from a local user database), use a PUT request. The body the request should include the entire object, but remove the element you want to delete from the related list of the object.

You cannot delete built-in policies that are available on SPS by default.

You must commit your changes to take effect. For details, see Commit a transaction on page 31.

**URL**

```
DELETE https:<IP-address-of-SPS>/api/configuration/<endpoint>/<object-id>
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command deletes an RDP Channel policy.

```
curl --cookie cookies -X DELETE -https:<IP-address-of-SPS>/api/configuration/rdp/channel_policies/<object-id>
```

**Response**

The following is a sample response received.
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>The resource was successfully deleted.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict</td>
<td>No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction.</td>
</tr>
</tbody>
</table>

Create a new object

To create a new object (for example, a new policy), complete the following steps.

1. Authenticate and open a transaction.
2. Post the new object as a JSON object to the appropriate resource URL.
3. If successful, the REST server creates an ID for the new object, and returns it in the key field of the response.
4. Commit the transaction.
Note the following points when you create a request:

- Note that you cannot simply use the JSON from the response of a similar object. If the object contains references to other resources (for example, a Channel policy references a Time policy), then the JSON object contains an embedded meta object. To get a valid JSON that you can use, you have to replace this embedded object with the ID (key) of the referenced object. For example, the following is a reference to a Time policy:

  ```json
  "time_policy": {
    "key": "-100",
    "meta": {
      "href": "/api/configuration/policies/time_policies/-100"
    }
  }
  ```

In a POST or PUT request, you have to change it to the following:

  ```json
  "time_policy": "-100",
  ```

Starting with version 6.1.0, when querying a list of objects, the API response includes the body of the referenced objects as well, not only its reference key, but only if they are immediate child nodes.

- You have to include empty fields in the object as well, for example:

  ```json
  "users": [
    { "certificates": [], "passwords": ["<reference-to-password>" ], "public_keys": [], "username": "myusername" }
  ]
  ```

- The API ignores any unrecognized or nonexistent keys that appear in the body of POST and PUT requests. For example, if you mistype the name of an optional key, it will be silently ignored.

- The body wrapper that is displayed in the response is not needed when you create or modify an object, for example:

  ```json
  {
    "name": "my-local-user-database",
    "users": [
      { "certificates": [], "passwords": ["<reference-to-password>" ], "public_keys": [], "username": "myusername" }
    ]
  }
  ```

URL

POST https://<IP-address-of-SPS>/api/configuration/<path-to-the-parent-resource>
### Table 1: Headers

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the type of the data sent. SPS uses the JSON format</td>
<td>Required</td>
<td>application/json</td>
</tr>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API.</td>
</tr>
</tbody>
</table>

### Sample request

The following command creates a new RDP Channel policy. The data content of the request is read from the file `body.json`.

```bash
curl -H "Content-Type: application/json" -d @body.json --cookie session_id=1aca4793549c6f22aecd98bc1047d1bf32dd76ef -X POST https://<object-id>/api/configuration/rdp/channel_policies/
```

For a simple RDP Channel policy that uses the default settings and allows only the Drawing channel, the JSON object is the following.

```json
{
    "name": "drawing-only",
    "rules": [
    {
        "actions": {
            "audit": true,
            "content_policy": null,
            "four_eyes": false,
            "ids": false
        },
        "allowed_for": {
            "clients": [],
            "gateway_groups": [],
            "remote_groups": [],
            "servers": [],
            "time_policy": "-100"
        }
    }
}
```
Response

The following is a sample response received, showing the properties of Content policy objects.

For details of the meta object, see Message format on page 10.

```
{
  "key": "f79bcc85-bb8b-4fa5-a141-eb4cf2b6ef33",
  "meta": {
    "href": "/api/configuration/rdp/channel_policies/f79bcc85-bb8b-4fa5-a141-eb4cf2b6ef33",
    "parent": "/api/configuration/rdp/channel_policies",
    "transaction": "/api/transaction"
  }
}
```

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>The request body format is invalid. The data is not a properly formatted JSON object.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict</td>
<td>No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction on page 29.</td>
</tr>
</tbody>
</table>
Change an object

To modify or update an object, use a PUT request on the object you want to change. In the body of the request, you have to upload the entire object, not only the parameter that you want to change.

To delete an element of a list (for example, a user from a local user database), use a PUT request. The body the request should include the entire object, but remove the element you want to delete from the related list of the object.

Note the following points when you create a request:

- Note that you cannot simply use the JSON from the response of a similar object. If the object contains references to other resources (for example, a Channel policy references a Time policy), then the JSON object contains an embedded meta object. To get a valid JSON that you can use, you have to replace this embedded object with the ID (key) of the referenced object. For example, the following is a reference to a Time policy:

```
"time_policy": {
    "key": "-100",
    "meta": {
        "href": "/api/configuration/policies/time_policies/-100"
    }
}
```

In a POST or PUT request, you have to change it to the following:

```
"time_policy": "-100",
```

Starting with version 6.1.0, when querying a list of objects, the API response includes the body of the referenced objects as well, not only its reference key, but only if they are immediate child nodes.

- You have to include empty fields in the object as well, for example:

```
"users": [
    {
        "certificates": [],
        "passwords": [ "<reference-to-password>"
    },
    "public_keys": [],
    "username": "myusername"
]
```
The API ignores any unrecognized or nonexistent keys that appear in the body of POST and PUT requests. For example, if you mistype the name of an optional key, it will be silently ignored.

The body wrapper that is displayed in the response is not needed when you create or modify an object, for example:

```json
{
    "name": "my-local-user-database",
    "users": [
        {
            "certificates": [],
            "passwords": [ "<reference-to-password>" ],
            "public_keys": [],
            "username": "myusername"
        }
    ]
}
```

**URL**


**Table 2: Headers**

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the type of the data sent. SPS uses the JSON format</td>
<td>Required</td>
<td>application/json</td>
</tr>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e6576347309e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API.</td>
</tr>
</tbody>
</table>

**Sample request**

The following command updates an RDP Channel policy. The data content of the request is read from the file body.json.

```
curl -H "Content-Type: application/json" -d @body.json --cookie session_id=07640a0bf14cdd3618f5ae2b0b482a786c7a604 -X PUT https://10.40.255.17/api/configuration/rdp/channel_policies/<id-of-the-object-to-modify>
```
For a simple RDP Channel policy that uses the default settings and allows only the Drawing channel, the JSON object is the following.

```json
{
    "name": "drawing-only",
    "rules": [
        {
            "actions": {
                "audit": true,
                "content_policy": null,
                "four_eyes": false,
                "ids": false
            },
            "allowed_for": {
                "clients": [],
                "gateway_groups": [],
                "remote_groups": [],
                "servers": [],
                "time_policy": "-100"
            },
            "channel": "#drawing"
        }
    ]
}
```

**Response**

The following is a sample response received.

For details of the meta object, see Message format on page 10.

```json
{
    "meta": {
        "first": "/api/configuration/rdp/channel_policies/-20100",
        "href": "/api/configuration/rdp/channel_policies/<id-of-the-modified-object>",
        "last": "/api/configuration/rdp/channel_policies/<id-of-the-modified-object>",
        "next": null,
        "parent": "/api/configuration/rdp/channel_policies",
        "previous": "/api/configuration/rdp/channel_policies/655555",
        "transaction": "/api/transaction"
    }
}
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>The request body format is invalid. The data is not a properly formatted JSON object.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict</td>
<td>No open Transaction is available. Open a transaction before using this request. For details, see <a href="#">Open a transaction</a> on page 29.</td>
</tr>
<tr>
<td>417</td>
<td>Expectation Failed</td>
<td>If you receive the &quot;417 - Expectation Failed&quot; error code when using curl, use curl with the --http1.0 or the -H &quot;Expect: &quot; option.</td>
</tr>
</tbody>
</table>
Basic settings

Retrieve basic firmware and host information

The /api/info endpoint contains generic information about the SPS host. Note that part of this information is available without authentication.

URL

GET https://<IP-address-of-SPS>/api/info

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format). |

Sample request

The following command displays the information about SPS that is available without authentication.
curl https://10.40.255.171/api/info

The following command displays the information about SPS that is available for authenticated users.

curl --cookie cookies https://<IP-address-of-SPS>/api/info

Response

The following is a sample response received by an anonymous user.

For details of the meta object, see Message format on page 10.

```
{
  "body": {
    "domainname": "example",
    "hostname": "scbwriter",
    "nickname": null,
    "plugin_sdk_version": {
      "feature": "1.4",
      "full": "1.4.4"
    },
    "support_link": "mailto:scb-administrator@example.com"
  },
  "key": "about_info",
  "meta": {
    "href": "/api/info",
    "parent": "/api"
  }
}
```

The following is a sample response received by an authenticated user.

```
{
  "body": {
    "analytics_enabled": false,
    "build_date": "2018-06-15T20:18:40+00:00",
    "config_hash": "2abde4c81d9b544bf3fae4f4b9657fc",
    "domainname": "example",
    "firmware_version": "5.7.0",
    "hostname": "scbwriter",
    "nickname": null,
    "plugin_sdk_version": {
      "feature": "1.4",
      "full": "1.4.4"
    },
    "roles": [
      "central-management",
      "search-master"
    ]
  }
}
```
```
{
    "analytics_enabled": false,
    "build_date": "2023-04-15T12:00:00Z",
    "config_hash": "5F7",
    "domainname": "example.com",
    "hostname": "host1.example.com",
    "nickname": "examplenick",
    "plugin_sdk_version": "1.2.3",
    "support_link": "mailto:scb-administrator@example.com",
    "version": "5 F7"
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytics_enabled</td>
<td>Indicates whether or not the One Identity Safeguard for Privileged Analytics module has been enabled.</td>
</tr>
<tr>
<td>build_date</td>
<td>Build date of the SPS firmware. This element is included in the response only for authenticated users.</td>
</tr>
<tr>
<td>config_hash</td>
<td>Contains the hash of the XML database running on the given SPS host.</td>
</tr>
<tr>
<td>domainname</td>
<td>Name of the domain used on the network. You can configure this parameter on the /api/configuration/network/naming endpoint. For details, see Naming options on page 86.</td>
</tr>
<tr>
<td>hostname</td>
<td>Name of the machine running SPS. You can configure this parameter on the /api/configuration/network/naming endpoint. For details, see Naming options on page 86.</td>
</tr>
<tr>
<td>nickname</td>
<td>The nickname of the SPS host. Use it to distinguish the devices. It is displayed in the core and boot login shells. You can configure this parameter on the /api/configuration/network/naming endpoint. For details, see Naming options on page 86.</td>
</tr>
<tr>
<td>plugin_sdk_version</td>
<td>The version number of the Plugin SDK.</td>
</tr>
<tr>
<td></td>
<td>- The value of feature represents the feature release version.</td>
</tr>
<tr>
<td></td>
<td>- The value of full represents the minor release version.</td>
</tr>
<tr>
<td>support_link</td>
<td>The e-mail address of the SPS administrator, as set in the admin_address parameter of the /api/configuration/management/email endpoint. For details, see Mail settings on page 144.</td>
</tr>
<tr>
<td>firmware_version</td>
<td>The version number of the firmware running on SPS, for</td>
</tr>
</tbody>
</table>
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
</tbody>
</table>

### Listing integrated products with SPS

List basic information about products that are integrated with One Identity Safeguard for Privileged Sessions (SPS).

**URL**

```
GET https://<IP-address-of-SPS>/api/integrated_products
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example,</td>
</tr>
</tbody>
</table>
### Operations

Operations with the `/integrated_products` endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
</table>
| Retrieving information about products integrated with SPS | GET         | /api/integrated_products | When SPS is joined to One Identity Starling, but One Identity Starling is not available, you will receive the following warning message:  

  Information about the integrated Starling products cannot be retrieved. Check the following:  
  - The Starling cloud service is available.  
  - Your SPS appliance is connected to the Internet.  

When your credentials to access One Identity Starling are invalid, you will receive the following warning message:  

  The credentials used... |

### Cookie Table

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td></td>
<td></td>
<td>a1f71d030e657634730b9e887cb59a5e656162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>for accessing Starling are invalid. This may happen because SPS was un-joined from Starling and restored to a previous joined state. Re-join the SPS to get valid credentials.</td>
</tr>
</tbody>
</table>

**Sample request**

The following command lists products that are integrated with SPS.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/integrated_products
```

**Response**

The following is a sample response received when SPS is joined with One Identity Starling, but not with any other products integrated with the One Identity Starling platform.

For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "name": "Defender",
            "link": null,
            "activated": false
        },
        {
            "name": "Connect",
            "link": null,
            "activated": false
        },
        {
            "name": "Governance",
            "link": null,
            "activated": false
        },
        {
            "name": "RemoteAccess",
            "link": null,
```
The following is a sample response received when SPS is joined with SPP.

```
{
  "items": [
    {
      "name": "Safeguard for Privileged Passwords",
      "activated": true,
      "link": "https://10.10.10.10"
    }
  ]
}
```

The following is a sample response received when SPS is joined with One Identity Starling, and it is integrated with certain One Identity Starling products.

```
{
  "items": [
    {
      "name": "Defender",
      "link": "https://2fa.cloud.oneidentity.com",
      "activated": true
    },
    {
      "name": "Connect",
      "link": "https://connect.cloud.oneidentity.com",
      "activated": true
    },
    {
      "name": "Governance",
      "link": null,
      "activated": false
    }
  ]
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>object array</td>
<td>A list of One Identity Starling products</td>
<td>When there are no products integrated with</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that are integrated SPS.</td>
<td>SPS, the items field returns empty:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{ &quot;items&quot;: []</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>items.name</td>
<td>string</td>
<td>The name of the integrated One Identity Starling product.</td>
<td></td>
</tr>
<tr>
<td>items.link</td>
<td>format (uri)</td>
<td>The URL of the integrated One Identity Starling product.</td>
<td>If the product is not integrated, the value of the link parameter will be null.</td>
</tr>
<tr>
<td>items.activated</td>
<td>boolean</td>
<td>Indicates whether the product is integrated with SPS or not.</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• true - the product is integrated with SPS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• false - the product is not integrated with SPS</td>
</tr>
</tbody>
</table>

**HTTP response codes**

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

**Firmware management**

A list of endpoints managing SPS firmware images.

**URL**

GET https://<IP-address-of-SPS>/api/firmware
## Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

## Sample request

The following command lists management configuration endpoints.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/firmware
```

## Response

The following is a sample response received when firmware-related configuration endpoints are listed.

For details of the meta object, see [Message format](#) on page 10.

```
{
   "items": [
      {
         "key": "slots",
         "meta": {
            "href": "/api/firmware/slots"
         }
      },
      {
         "key": "test",
         "meta": {
            "href": "/api/firmware/test"
         }
      },
      {
         "key": "upgrade",
         "meta": {
            "href": "/api/firmware/upgrade"
         }
      }
   ]
}
```
Endpoints | Description
--- | ---
slots | Retrieve information about SPS firmware images maintained on the device in locations called slots.
test | Use the /firmware/test endpoint as a precheck tool to trigger an upgrade test without an actual upgrade.
upgrade | Use the /firmware/upgrade endpoint to upgrade SPS to a new firmware.
upload | Use the /upload/firmware endpoint to upload new firmware to SPS.

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

Retrieving information about SPS firmware image slots

Retrieve information about SPS firmware images maintained on the device in locations called slots.

URL

GET https://<IP-address-of-SPS>/api/firmware/slots
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
</table>
| session_id  | Contains the authentication token of the user | Required | The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. **NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format)..

Operations

Operations with the /firmware/slots endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieving information from firmware image slots</td>
<td>GET</td>
<td>/api/firmware/slots</td>
<td></td>
</tr>
<tr>
<td>Retrieving information from a specific firmware image slot</td>
<td>GET</td>
<td>/api/firmware/slots/&lt;slot_id&gt;</td>
<td>Example: /api/firmware/slots/1</td>
</tr>
<tr>
<td>Deleting a firmware image slot</td>
<td>DELETE</td>
<td>/api/firmware/slots/&lt;slot_id&gt;</td>
<td><strong>NOTE:</strong> Deleting the current of after reboot firmware is not allowed.</td>
</tr>
</tbody>
</table>

Sample request

The following command lists all available firmware image slots.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/firmware/slots
```
Response

The following is a sample response received when available firmware image slots are listed.

For details of the meta object, see Message format on page 10.

```json
{
   "items": [
      {
         "body": {
            "after_reboot": false,
            "current": true,
            "upgrade_news": null,
            "upgrade_notes": "some notes",
            "version": "6.5.0"
         },
         "key": "1",
         "meta": {
            "href": "/api/firmware/slots/1"
         }
      },
      {
         "body": {
            "after_reboot": true,
            "current": false,
            "upgrade_news": null,
            "upgrade_notes": "some notes",
            "version": "6.6.0"
         },
         "key": "2",
         "meta": {
            "href": "/api/firmware/slots/2"
         }
      },
      {
         "body": {},
         "key": "3",
         "meta": {
            "href": "/api/firmware/slots/3"
         }
      },
      {
         "body": {
            "after_reboot": false,
            "current": false,
            "upgrade_news": null,
            "upgrade_notes": "some notes",
            "version": "6.7.0"
         }
      }
   ]
}
```
The following is a sample response received when a specific firmware image slot is listed.

```
{
    "body": {
        "after_reboot": false,
        "current": true,
        "upgrade_news": null,
        "upgrade_notes": "some notes",
        "version": "6.5.0"
    },
    "key": "1",
    "meta": {
        "href": "/api/firmware/slots/1",
        "parent": "/api/firmware/slots"
    }
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>after_reboot</td>
<td>boolean</td>
<td>This flag shows that the firmware is selected to be the active firmware</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• true - the</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------</td>
<td>---------</td>
<td>-----------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>firmware is the active firmware after upgrade.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false - the firmware is not the active firmware after</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>upgrade</td>
<td></td>
</tr>
<tr>
<td>current</td>
<td>boolean</td>
<td>Indicates whether the firmware is active.</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true - the current firmware is the latest version</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false - the current firmware is not the latest version</td>
<td></td>
</tr>
<tr>
<td>upgrade_news</td>
<td>string</td>
<td>Displays SPS Upgrade Notes that is relevant to the</td>
<td>If there is no such information available, the value will be null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>current firmware.</td>
<td></td>
</tr>
<tr>
<td>upgrade_notes</td>
<td>string</td>
<td>The content of the SPS Upgrade Notes.</td>
<td></td>
</tr>
<tr>
<td>version</td>
<td>number</td>
<td>The version number of that specific SPS firmware image.</td>
<td></td>
</tr>
</tbody>
</table>

**HTTP response codes**

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

**Testing new SPS firmware before upgrade**

Use the /firmware/test endpoint as a precheck tool to trigger an upgrade test without an actual upgrade. The test reveals whether the current state of SPS is compatible with the new firmware. Possible areas where errors can occur:

- version compatibility - the new firmware is compatible with the upgrade policy
- storage space - there is enough free storage space for the upgrade?
- configuration compatibility - the current configuration settings are supported in the
new firmware

- whether or not there are manually modified files on the firmware (tainted firmware)

URL

GET https://<IP-address-of-SPS>/api/firmware/test

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Operations

Operations with the /firmware/test endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testing new SPS firmware before upgrade</td>
<td>POST</td>
<td>/api/firmware/test</td>
<td></td>
</tr>
</tbody>
</table>

Sample request

The following request triggers a firmware upgrade test.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/firmware/test --data '{"slot_id": <slot_id>}'
```

| NOTE: You require a payload first, from which you choose the firmware you want to test. |

Response

There can be three possible outcomes:
- the upgrade was successful:

```
{
  "body": "...",  "key": <slot_id>
}
```

- the upgrade was not successful due to invalid input
  - the slot_id is missing from the payload
    ```json
    {"error": {"type": "IncompleteRequestBodyError"}}
    ```
  - the tested firmware slot is empty
    ```json
    {"error": {"type": "ResourceNotFound"}}
    ```

- the upgrade was not successful due to an error:
  ```json
  {"error": {"type": "FirmwareTestFailed"}}
  ```

The following is a sample response received when the upgrade test was successful.
For details of the `meta` object, see `Message format` on page 10.

```
{
  "body": {
    "test_summary": "HA check started\nHA check finished" },
  "key": "1",
  "meta": {
    "href": "/api/firmware/test",
    "parent": "/api/firmware",
    "slot": "/api/firmware/slots/1",
    "upgrade": "/api/firmware/upgrade"
  }
}
```

The following is a sample response received when the upgrade test was not successful, because the firmware slot is empty.

```
{
  "error": {
    "details": {
      "mount_point": "/firmware/test",
      "resource": "3"
    },
    "message": "Resource was not found",
    "type": "ResourceNotFound"
  },
  "meta": {
```
Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.details</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>error.details.mount_point</td>
<td>string</td>
<td>The reference point - in this case a URL path - at which details of the error can be accessed.</td>
<td>This is a fix value: /firmware/test</td>
</tr>
<tr>
<td>error.details.resource</td>
<td>number</td>
<td>The identifier of the firmware image slot.</td>
<td></td>
</tr>
<tr>
<td>error.message</td>
<td>string</td>
<td>The content of the error message.</td>
<td></td>
</tr>
<tr>
<td>error.type</td>
<td>string</td>
<td>The type of the error message.</td>
<td></td>
</tr>
</tbody>
</table>

The following is a sample response received when the upgrade test was not successful, because the slot_id is missing from the payload.

```
{
    "error": {
        "details": {
            "missing_paths": [
                "slot_id"
            ],
            "message": "Some paths were missing from the request body",
            "type": "IncompleteRequestBodyError"
        },
        "meta": {
            "href": "/api/firmware/test",
            "parent": "/api/firmware"
        }
    }
}
```

Elements of the response message body include:
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.details</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>error.details.missing_</td>
<td>array</td>
<td>A list of missing URL path parameters.</td>
<td>In this case, there can be only one value here, which is slot_id.</td>
</tr>
<tr>
<td>error.details.missing_</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>error.message</td>
<td>string</td>
<td>The content of the error message.</td>
<td></td>
</tr>
<tr>
<td>error.type</td>
<td>string</td>
<td>The type of the error message.</td>
<td></td>
</tr>
</tbody>
</table>

The following is a sample response received when the upgrade test was not successful, because an error was found during testing.

```json
{
    "error": {
        "details": {
            "exit_code": 1,
            "test_summary": "HA check started\nHA check failed"
        },
        "message": "The firmware test failed",
        "type": "FirmwareTestFailed"
    },
    "meta": {
        "href": "/api/firmware/test",
        "parent": "/api/firmware"
    }
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.details</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>error.details.exit_code</td>
<td>number</td>
<td>Possible values:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>⚫ 0 - the firmware test passed successfully</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>⚫ 1 - the firmware test failed</td>
<td></td>
</tr>
<tr>
<td>error.details.test_</td>
<td>string</td>
<td>The summary of the test</td>
<td>For example:</td>
</tr>
</tbody>
</table>
### HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

### Upgrading SPS to a new firmware

Use the `/firmware/upgrade` endpoint to upgrade SPS to a new firmware.

**URL**

GET https://<IP-address-of-SPS>/api/firmware/upgrade

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Operations**

Operations with the `/firmware/upgrade` endpoint include:
<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upgrading SPS to a new firmware</td>
<td>POST</td>
<td>/api/firmware/upgrade</td>
<td></td>
</tr>
</tbody>
</table>

### Sample request

The following command upgrades SPS to a new firmware.

```shell
curl --cookie cookies https://<IP-address-of-SPS>/api/firmware/upgrade --data '{"slot_id": <slot_id>, "message": "..." | null}'
```

**NOTE:** The value of `message` can be null, if `require_commit_log` is disabled. If `require_commit_log` is enabled, then `message` is filled.

### Response

There can be three possible outcomes:

- the upgrade was successful:
  ```json
  {"body": "...", "key": <slot_id>}
  ```

- the upgrade was not successful due to invalid input
  - the `slot_id` is missing from the payload
    ```json
    {"error": {"type": "IncompleteRequestBodyError"}}
    ```
  - the tested firmware slot is empty
    ```json
    {"error": {"type": "ResourceNotFound"}}
    ```

- the upgrade was not successful due to an error:
  ```json
  {"error": {"type": "FirmwareTestFailed"}}
  ```

The following is a sample response received when a SPS is upgraded to a new firmware. For details of the `meta` object, see Message format on page 10.

```json
{
  "body":
  { "test_summary": "HA check started\nHA check finished" },
  "key": "1",
  "meta": {
    "href": "/api/firmware/upgrade",
```
The following is a sample response received when a firmware upgrade is attempted on an empty slot.

```json
{
    "error": {
        "details": {
            "mount_point": "/firmware/upgrade",
            "resource": "3"
        },
        "message": "Resource was not found",
        "type": "ResourceNotFoundException"
    },
    "meta": {
        "href": "/api/firmware/upgrade",
        "parent": "/api/firmware"
    }
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.details</td>
<td>object</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| error.details.mount_point | string     | The reference point - in this case a URL path - at which details of the error can be accessed. | This is a fix value: 
|                          |            | /firmware/upgrade                                                          |                                    |
| error.details.resource    | number     | The identifier of the firmware image slot.                                 |                                    |
| error.message            | string     | The content of the error message.                                          |                                    |
| error.type               | string     | The type of the error message.                                             |                                    |

The following is a sample response received when message is missing from the request body during upgrade.
Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.details</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>error.details.missing_paths</td>
<td>array</td>
<td>A list of missing URL path parameters.</td>
<td>In this case, there can be only one value here, which is message.</td>
</tr>
<tr>
<td>error.message</td>
<td>string</td>
<td>The content of the error message.</td>
<td></td>
</tr>
<tr>
<td>error.type</td>
<td>string</td>
<td>The type of the error message.</td>
<td></td>
</tr>
</tbody>
</table>

The following is a sample response received when the firmware test fails during upgrade.

```json
{
    "error": {
        "details": {
            "exit_code": 1,
            "test_summary": "HA check started\nHA check failed"
        },
        "message": "The firmware test failed",
        "type": "FirmwareTestFailed"
    },
    "meta": {
```
Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.details</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>error.details.exit_code</td>
<td>number</td>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0 - the firmware test passed successfully</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1 - the firmware test failed</td>
</tr>
<tr>
<td>error.details.test_summary</td>
<td>string</td>
<td>The summary of the upgrade test in free text format.</td>
<td>For example:</td>
</tr>
<tr>
<td>error.message</td>
<td>string</td>
<td>The content of the error message.</td>
<td></td>
</tr>
<tr>
<td>error.type</td>
<td>string</td>
<td>The type of the error message.</td>
<td></td>
</tr>
</tbody>
</table>

**HTTP response codes**

HTTP response codes comprise of standard or endpoint-specific HTTP status and error codes. The following table lists the endpoint-specific HTTP response codes for this request.

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status/Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>UpgradeMessageMissing</td>
<td>The upgrade request did not contain a message. Adding a message is required.</td>
</tr>
</tbody>
</table>

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.
Uploading new firmware to SPS

Use the /upload/firmware endpoint to upload new firmware to SPS.

**URL**

POST https://<IP-address-of-SPS>/api/upload/firmware

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

**Operations**

Operations with the /upload/firmware endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uploading SPS firmware images</td>
<td>POST</td>
<td>/api/upload/firmware</td>
<td>The <strong>enctype</strong> attribute of the POST request must be <strong>multipart/form-data</strong>.</td>
</tr>
</tbody>
</table>

**Sample request**

The following command uploads a new firmware to SPS.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/upload/firmware --form firmware=@<sps.iso>
```
Where `<sps.iso>` is the path of the new firmware.

**Response**

The following is a sample response received when a new firmware image is uploaded. For details of the `meta` object, see Message format on page 10.

```json
{
    "body": {
        "after_reboot": true,
        "current": false,
        "upgrade_notes": null,
        "upgrade_news": "some notes",
        "version": "6.6.0"
    },
    "key": "2",
    "meta": {
        "href": "/api/upload/firmware",
        "slot": "/api/firmware/slots/2",
        "test": "/api/firmware/test",
        "upgrade": "/api/firmware/upgrade",
        "number_of_empty_slots": 2
    }
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>after_reboot</td>
<td>boolean</td>
<td>This flag shows that the firmware is selected to be the active firmware after upgrade.</td>
<td>In this particular case, the value of <code>after_reboot</code> will always be false, as the firmware has not yet been chosen for upgrade.</td>
</tr>
<tr>
<td>current</td>
<td>boolean</td>
<td>Indicates whether the firmware is active.</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• true - the current firmware is the latest version</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• false - the current firmware is not the latest version</td>
</tr>
<tr>
<td>upgrade_news</td>
<td>string</td>
<td>Displays SPS Upgrade Notes that is relevant to the current firmware.</td>
<td>If there is no such information available, the value will be null.</td>
</tr>
</tbody>
</table>
### HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

### Network settings

#### Network configuration options

Contains the endpoints for configuring networking on SPS.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/network
```

#### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists network configuration options.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network

Response

The following is a sample response received when listing network configuration options.
For details of the meta object, see Message format on page 10.

{  
  "items": [  
    {  
      "key": "dns",  
      "meta": {  
        "href": "/api/configuration/network/dns"  
      }  
    },  
    {  
      "key": "ip_forwarding_rule_pairs",  
      "meta": {  
        "href": "/api/configuration/network/ip_forwarding_rule_pairs"  
      }  
    },  
    {  
      "key": "naming",  
      "meta": {  
        "href": "/api/configuration/network/naming"  
      }  
    },  
    {  
      "key": "nics",  
      "meta": {  
        "href": "/api/configuration/network/nics"  
      }  
    },  
    {  
      "key": "routing",  
      "meta": {  
        "href": "/api/configuration/network/routing"  
      }  
    }  
  ],  
  "meta": {  
    "first": "/api/configuration/aaa",  
    "href": "/api/configuration/network/",  
    "last": "/api/configuration/x509",  
    ...  
  }  
}
Element | Description
---|---
dns | The address of the primary and secondary DNS server.
ip_forwarding_rule_pairs | Rules for routing between the network interfaces.
naming | DNS search domain, hostname, and appliance nickname settings.
nics | References the endpoints of the three physical network interfaces.
routing | Routing table. Defines the address of the gateway server for each configured subnet.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**DNS servers**

Contains the address of the primary and secondary DNS server.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/network/dns
### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d03e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API on page 19</a>.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists the configured DNS servers.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/dns
```

### Response

The following is a sample response received when listing the configured DNS servers.

For details of the meta object, see [Message format on page 10](#).

```json
{
    "body": {
        "primary": "192.168.56.1",
        "secondary": null
    },
    "key": "dns",
    "meta": {
        "first": "/api/configuration/network/dns",
        "href": "/api/configuration/network/dns",
        "last": "/api/configuration/network/routing",
        "next": "/api/configuration/network/ip_forwarding_rule_pairs",
        "parent": "/api/configuration/network",
        "previous": null,
        "transaction": "/api/transaction"
    }
}
```
Modify the address of the DNS servers

To modify the address of a DNS server, you have to:

1. **Open a transaction.**
   
   For more information, see *Open a transaction* on page 29.

2. **Modify the JSON object of the endpoint.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/network/dns endpoint. You can find a detailed description of the available parameters listed in *Element*.

3. **Commit your changes.**
   
   For more information, see *Commit a transaction* on page 31.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see *Application level error codes* on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Routing between interfaces

Configures routing between network interfaces. To use an interface in single-interface router mode, configure both interface_a and interface_b elements to reference that same interface.

URL

GET https://<IP-address-of-SPS>/api/configuration/network/ip_forwarding_rule_pairs

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists interface routing rules.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/ip_forwarding_rule_pairs

Response

The following is a sample response received when listing interface routing rules. For details of the meta object, see Message format on page 10.
```json
{
    "body": [
        {
            "interface_a": {
                "key": "nic1.interfaces.ff7574025754b3df1647001",
                "meta": {
                    "href": "/api/configuration/network/nics/nic1/interfaces/ff7574025754b3df1647001"
                }
            },
            "interface_b": {
                "key": "nic1.interfaces.ff7574025754b3df1647001",
                "meta": {
                    "href": "/api/configuration/network/nics/nic1/interfaces/ff7574025754b3df1647001"
                }
            }
        },
        {
            "key": "ip_forwarding_rule_pairs",
            "meta": {
                "first": "/api/configuration/network/dns",
                "href": "/api/configuration/network/ip_forwarding_rule_pairs",
                "last": "/api/configuration/network/routing",
                "next": "/api/configuration/network/naming",
                "parent": "/api/configuration/network",
                "previous": "/api/configuration/network/dns",
                "transaction": "/api/transaction"
            }
        }
    ]
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (list)</td>
<td>Contains the rules for routing between the network interfaces.</td>
</tr>
<tr>
<td>interface_a</td>
<td>string</td>
<td>References the identifier of the network interface. You can configure network interfaces at the /api/configuration/network/nics/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a network interface, use the value of the returned key as the value of the interface_a element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>interface_b</td>
<td>string</td>
<td>References the identifier of the network interface. You can configure network interfaces at the</td>
</tr>
</tbody>
</table>
Add a rule for routing between the network interfaces

To add a rule, you have to:

1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new list of rules.**
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/network/ip_forwarding_rule_pairs endpoint. You can find a detailed description of the available parameters listed in Element. If the POST request is successful, the response includes the key of the new rule.

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

Modify a rule for routing between the network interfaces

To modify a rule, you have to:

1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the list of rules.**
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/network/ip_forwarding_rule_pairs endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.
**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Naming options**

Contains the settings for the DNS search domain, hostname, and appliance nickname.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/network/naming

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format)....</td>
</tr>
</tbody>
</table>
Sample request

The following command lists the naming settings.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/naming
```

Response

The following is a sample response received when listing naming settings.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "domainname": "example",
        "hostname": "api-docs",
        "nickname": null
    },
    "key": "naming",
    "meta": {
        "first": "/api/configuration/network/dns",
        "href": "/api/configuration/network/naming",
        "last": "/api/configuration/network/routing",
        "next": "/api/configuration/network/nics",
        "parent": "/api/configuration/network",
        "previous": "/api/configuration/network/ip_forwarding_rule_pairs",
        "transaction": "/api/transaction"
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>Contains the naming settings.</td>
</tr>
<tr>
<td></td>
<td>element</td>
<td>(string)</td>
</tr>
<tr>
<td>domainname</td>
<td>string</td>
<td>The domain name of the network.</td>
</tr>
<tr>
<td>hostname</td>
<td>string</td>
<td>The hostname of SPS.</td>
</tr>
<tr>
<td>nickname</td>
<td>string</td>
<td>The nickname for the appliance. Use this name to distinguish between multiple SPS appliances on the network. This name is visible in the boot and core login shells.</td>
</tr>
</tbody>
</table>

Modify a name

To modify a name, you have to:
1. **Open a transaction.**
   For more information, see *Open a transaction* on page 29.

2. **Modify the JSON object of the endpoint.**
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/network/naming endpoint. You can find a detailed description of the available parameters listed in *Element*.

3. **Commit your changes.**
   For more information, see *Commit a transaction* on page 31.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see *Application level error codes* on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Network addresses

Contains the network addresses configured for each physical NIC.

### URL

GET https://<IP-address-of-SPS>/api/configuration/network/nics
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Sample request

The following command lists the endpoints for the physical network interfaces.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/nics/
```

The following commands retrieve the properties of a specific physical network interface.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/nics/nic1

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/nics/nic2

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/nics/nic3
```

Response

The following is a sample response received when listing physical network interfaces. For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "nic1",
      "meta": {
        ...
      }
    }
  ]
}
```
When retrieving the endpoint of a specific physical network interface, the response is the following.

```json
{
    "body": {
        "interfaces": {
            "@order": ["ff7574025754b3df1647001"],
            "ff7574025754b3df1647001": {
                "addresses": {
                    "1": "198.51.100.123/24",
                    "6001481625b7c21ef97598": "2001:db8:1234::5678/48",
                    "@order": ["1", "6001481625b7c21ef97598"]
                },
                "mtu": 1500,
                "name": "external",
                "source_based_routes": [
                    {
                        "gateway": "198.51.100.1",
                        "target_network": "203.0.113.0/24"
                    },
                    {
                        "gateway": "2001:db8:1234::1",
```
"target_network": "2001:db8:aaaa::/48"
}
],
"vlantag": 0
}
},
"name": "eth0",
"speed": "auto"
}
,"key": "nic1",
"meta": {
"first": "/api/configuration/network/nics/nic1",
"href": "/api/configuration/network/nics/nic1",
"last": "/api/configuration/network/nics/nic3",
"next": "/api/configuration/network/nics/nic2",
"parent": "/api/configuration/network/nics",
"previous": null,
"remaining_seconds": 10800,
"transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the physical network interface (nic1, nic2 or nic3).</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the properties of the physical network interface.</td>
</tr>
<tr>
<td>interfaces</td>
<td>Top level item</td>
<td>Contains the configuration of all virtual interfaces on the physical NIC.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The system name of the physical network interface (eth0, eth1 or eth2). Do not change this value.</td>
</tr>
<tr>
<td>speed</td>
<td>string</td>
<td>The speed of the physical network interface. The default value is auto. Change this setting only for troubleshooting purposes. Possible values are:</td>
</tr>
</tbody>
</table>
|             |        | - auto
|             |        |   Negotiate the network speed automatically. This is the default value. |
|             |        | - 10-half
|             |        |   10BaseT/Half. |
|             |        | - 100-half
<p>|             |        |   100BaseT/Half. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-full</td>
<td>string</td>
<td>10BaseT/Full.</td>
</tr>
<tr>
<td>100-full</td>
<td>string</td>
<td>100BaseT/Full.</td>
</tr>
<tr>
<td>1000-full</td>
<td>string</td>
<td>1000BaseT/Full.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of interfaces</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>@order</td>
<td>list</td>
<td>Lists the keys of the interfaces in the order they are be displayed on the SPS web UI.</td>
</tr>
<tr>
<td>&lt;key-of-an-interface&gt;</td>
<td>string</td>
<td>Contains the addresses, name, and vlantag of the network interface. Each physical NIC has an automatically created interface key, where the value of the vlanid element is set to 0. To add a valid virtual network interface to the physical NIC, create an additional interface, and assign a value between 1 and 4094 to its vlanid element.</td>
</tr>
<tr>
<td>addresses</td>
<td>string</td>
<td>Contains the addresses of the interface, and their display order.</td>
</tr>
<tr>
<td>&lt;key-of-address&gt;</td>
<td>string</td>
<td>Contains the IP address range.</td>
</tr>
<tr>
<td>@order</td>
<td>list</td>
<td>Lists the keys of the addresses in the order they are displayed on the SPS web UI.</td>
</tr>
<tr>
<td>mtu</td>
<td>integer</td>
<td>Maximum Transmission Unit (MTU) to set per network interface (VLAN or network interface card). Default value: 1500</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the interface, as displayed on the SPS web UI.</td>
</tr>
<tr>
<td>source_</td>
<td>list</td>
<td>Contains details of the network.</td>
</tr>
<tr>
<td>Elements of interfaces</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>based_routes</td>
<td>routing rule specific to packets coming out of this particular interface.</td>
<td></td>
</tr>
<tr>
<td>vlantag</td>
<td>string</td>
<td>The ID of the interface. For the physical interface, the value is 0. For virtual interfaces, the value is between 1 and 4094.</td>
</tr>
</tbody>
</table>

**CAUTION:**
Do not set the VLAN ID unless your network environment is already configured to use this VLAN. Otherwise, your SPS appliance will be unavailable using this interface.

<table>
<thead>
<tr>
<th>Elements of source_based_routes</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gateway</td>
<td>string</td>
<td>The IPv4 or IPv6 address of the gateway used to access the network set in this routing rule.</td>
</tr>
<tr>
<td>target_network</td>
<td>string</td>
<td>The IPv4 or IPv6 address of the host or network accessible via this routing rule.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but</td>
</tr>
</tbody>
</table>
Routing table

Contains the address of the gateway server for each configured subnet.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/network/routing

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the configured subnets and the corresponding gateway servers.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/network/routing
```

**Response**

The following is a sample response received when viewing the routing table.

For details of the meta object, see Message format on page 10.
{  "body": [  {  "gateway": "192.168.56.1",
   "target_network": "0.0.0.0/0"
  }  ],
  "key": "routing",
  "meta": {
   "first": "/api/configuration/network/dns",
   "href": "/api/configuration/network/routing",
   "last": "/api/configuration/network/routing",
   "next": null,
   "parent": "/api/configuration/network",
   "previous": "/api/configuration/network/nics",
   "transaction": "/api/transaction"
  }
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (list)</td>
<td>Contains the routing table.</td>
</tr>
<tr>
<td>gateway</td>
<td>string</td>
<td>The IP address of the gateway server.</td>
</tr>
<tr>
<td>target_network</td>
<td>string</td>
<td>The network id (IP address and subnet mask) of the subnet.</td>
</tr>
</tbody>
</table>

Add a subnet

To add a subnet, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new routing table.**
   
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/network/routing endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.
**Modify the routing table**

To modify the routing table, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the routing table.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/network/routing endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Local services of SPS**

Contains the endpoints for configuring the local services of SPS.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/local_services
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the local services.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/local_services
```

Response

The following is a sample response received when listing local services.

For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "admin_web",
            "meta": {
                "href": "/api/configuration/local_services/admin_web"
            }
        },
        {
            "key": "analytics",
            "meta": {
                "href": "/api/configuration/local_services/analytics"
            }
        },
        {
            "key": "cluster",
            "meta": {
                "href": "/api/configuration/local_services/cluster"
            }
        }
    ]
}
```
"href": "/api/configuration/local_services/cluster"
},
{
  "key": "indexer",
  "meta": {
    "href": "/api/configuration/local_services/indexer"
  }
},
{
  "key": "postgresql",
  "meta": {
    "href": "/api/configuration/local_services/postgresql"
  }
},
{
  "key": "snmp_agent",
  "meta": {
    "href": "/api/configuration/local_services/snmp_agent"
  }
},
{
  "key": "ssh",
  "meta": {
    "href": "/api/configuration/local_services/ssh"
  }
},
{
  "key": "user_web",
  "meta": {
    "href": "/api/configuration/local_services/user_web"
  }
}],
"meta": {
  "first": "/api/configuration/aaa",
  "href": "/api/configuration/local_services",
  "last": "/api/configuration/x509",
  "next": "/api/configuration/management",
  "parent": "/api/configuration",
  "previous": "/api/configuration/ica",
  "transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin_web</td>
<td>Web login for administrators and users: On this address, users can,</td>
</tr>
</tbody>
</table>
depending on their access privileges, modify the configuration of SPS, and perform authentication-related activities (gateway authentication, 4-eyes authorization).

analytics  Enables One Identity Safeguard for Privileged Analytics.
To enable One Identity Safeguard for Privileged Analytics and analyze the behavior of your users, One Identity Safeguard for Privileged Sessions (SPS) requires a special license. Also, depending on the number of your users and sessions, the performance and sizing of SPS must be considered. If you are interested in One Identity Safeguard for Privileged Analytics, contact our Sales Team, or your One Identity representative. For details on One Identity Safeguard for Privileged Analytics, see the One Identity One Identity Safeguard for Privileged Analytics website. For details on enabling One Identity Safeguard for Privileged Analytics, see Safeguard for Privileged Analytics Configuration Guide.

cluster  Configure the cluster service of SPS.

indexer  Configure the indexer services of SPS, including remote indexing.

postgresql  Configure direct remote access to the connection database of SPS.

snmp_agent  Configure the SNMP server of SPS.

ssh  Configure remote SSH access to SPS.

user_web  Web login for users only: The configuration of SPS cannot be viewed or altered from this address. Users (even ones with administrator privileges) can only perform gateway authentication and 4-eyes authorization.

Status and error codes
The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Local services: Web login for administrators

The SPS administrators and users can, depending on their access privileges, modify the configuration of SPS, and perform authentication-related activities (gateway authentication, 4-eyes authorization). On this endpoint you can configure on which interfaces can the administrators access SPS, and optionally restrict the access to these interfaces.

URL

GET https://<IP-address-of-SPS>/api/configuration/local_services/admin_web

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configuration options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/local_services/admin_web
```

Response

The following is a sample response received when listing the configuration options. For details of the meta object, see Message format on page 10.
{  
  "body": {  
    "access_restriction": {  
      "allowed_from": [  
        "10.40.0.0/16"  
      ],  
      "enabled": true  
    },  
    "listen": [  
      {  
        "address": {  
          "key": "nic1.interfaces.ff754025754b3df1647001.addresses.1",  
          "meta": {  
            "href": "/api/configuration/network/nics/nic1#interfaces/ff754025754b3df1647001/addresses/1"  
          }  
        },  
        "http_port": 80,  
        "https_port": 443  
      }  
    ],  
    "key": "admin_web",  
    "meta": {  
      "first": "/api/configuration/local_services/admin_web",  
      "href": "/api/configuration/local_services/admin_web",  
      "last": "/api/configuration/local_services/user_web",  
      "next": "/api/configuration/local_services/indexer",  
      "parent": "/api/configuration/local_services",  
      "previous": null,  
      "transaction": "/api/transaction"  
    }  
  }  
}

**Element** | **Type** | **Description**
---|---|---
key | string | Top level element, contains the ID of the endpoint.
body | | Contains the configuration options of the SPS web interface.
access_restriction | JSON | Enables and configures limitations on the clients that can
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>restriction</td>
<td>object</td>
<td>access the web interface, based on the IP address of the clients.</td>
</tr>
<tr>
<td>allowed_from</td>
<td>list</td>
<td>The list of IP networks from where the administrators are permitted to access this management interface. To specify the IP addresses or networks, use the IPv4-Address/prefix format, for example, 10.40.0.0/16.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set it to true to restrict access to the specified client addresses.</td>
</tr>
<tr>
<td>bruteforce_protection</td>
<td>boolean</td>
<td>Enables protection against brute-force attacks by denying access after failed login attempts for increasingly longer period. Enabled by default.</td>
</tr>
<tr>
<td>listen</td>
<td>list</td>
<td>Selects the network interface, IP address, and port where the clients can access the web interface.</td>
</tr>
<tr>
<td>address</td>
<td>JSON object</td>
<td>A reference to a configured network interface and IP address where this local service accepts connections. For example, if querying the interface /api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/ returns the following response:</td>
</tr>
</tbody>
</table>

```json
{
  "body": {
    "interfaces": {
      "@order": [
        "ff7574025754b3df1647001",
        "ff7574025754b3df1647001": {
          "addresses": {
            "1": "10.40.255.171/24",
            "@order": ["1"
          ]
        },
        "name": "default",
        "vlantag": 0
      },
      "name": "eth0",
      "speed": "auto"
    }
  }
}
```
Then the listening address of the local service is the following.

```
nic1.interfaces.ff7574025754b3df1647001.addresses.1
```

This is the format you have to use when configuring the address of the local service using REST:

```
"address": "nic1.interfaces.ff7574025754b3df1647001.addresses.1"
```

When querying a local services endpoint, the response will contain a reference to the IP address of the interface in the following format:

```
"address": {
  "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1",
  "meta": {
    "href": "/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001_addresses/1"
  }
},
```
### Element Type Description

http_port integer The port number where SPS accepts HTTP connections. Note that SPS automatically redirects connections from this port to the HTTPS port set in https_port.

http_s_port integer The port number where SPS accepts HTTPS connections.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Local services: Web login for users

The SPS users can perform authentication-related activities (gateway authentication, 4-eyes authorization). On this endpoint you can configure on which interfaces can the users access SPS, and optionally restrict the access to these interfaces.

### URL

GET https://<IP-address-of-SPS>/api/configuration/local_services/user_web
### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following request command lists the configuration options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/local_services/user_web
```

### Response

The following is a sample response received when listing the configuration options. For details of the meta object, see [Message format](#) on page 10.

```
{
   "body": {
      "access_restriction": {
         "allowed_from": [
            "10.46.0.0/16"
         ],
         "enabled": true
      },
      "listen": [
         {
            "address": {
               "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1",
               "meta": {
                  "href": "/api/configuration/network/nics/nic1/interfaces/ff7574025754b3df1647001/addresses/1"
               }
            }
         }
      ]
   }
}
```
Element | Type | Description
---|---|---
key | string | Top level element, contains the ID of the endpoint.
body | Top level element (string) | Contains the configuration options of the SPS web interface.
access_restriction | JSON object | Enables and configures limitations on the clients that can access the web interface, based on the IP address of the clients.
allowed_from | list | The list of IP networks from where the administrators are permitted to access this management interface. To specify the IP addresses or networks, use the IPv4-Address/prefix format, for example, 10.40.0.0/16.
enabled | boolean | Set it to true to restrict access to the specified client addresses.
bruteforce_protection | boolean | Enables protection against brute-force attacks by denying access after failed login attempts for increasingly longer period. Enabled by default.
listen | list | Selects the network interface, IP address, and port where the clients can access the web interface.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>JSON object</td>
<td>A reference to a configured network interface and IP address where this local service accepts connections. For example, if querying the interface /api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/ returns the following response:</td>
</tr>
</tbody>
</table>

```
{
  "body": {
    "interfaces": {
      "@order": [
        "ff7574025754b3df1647001"
      ],
      "ff7574025754b3df1647001": {
        "addresses": {
          "1": "10.40.255.171/24",
          "@order": [
            "1"
          ]
        },
        "name": "default",
        "vlantag": 0
      }
    },
    "name": "eth0",
    "speed": "auto"
  },
  "key": "nic1",
  "meta": {
    "first": "/api/configuration/network/nics/nic1",
    "href": "/api/configuration/network/nics/nic1",
    "last": "/api/configuration/network/nics/nic3",
    "next": "/api/configuration/network/nics/nic2",
    "parent": "/api/configuration/network/nics",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

Then the listening address of the local service is the following.
This is the format you have to use when configuring the address of the local service using REST:

"address": "nic1.interfaces.ff7574025754b3df1647001.addresses.1"

When querying a local services endpoint, the response will contain a reference to the IP address of the interface in the following format:

"address": {
  "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1",
  "meta": {
    "href": "/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/1"
  }
},

http_port integer The port number where SPS accepts HTTP connections. Note that SPS automatically redirects connections from this port to the HTTPS port set in https_port.

http_s_port integer The port number where SPS accepts HTTPS connections.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Local services: cluster interface**

To enable cluster management, enable the cluster interface on all nodes that you want to be part of your One Identity Safeguard for Privileged Sessions (SPS) cluster. Complete the following steps on each node of the cluster.

| NOTE: All nodes in a cluster must run the same version of SPS. |

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/local_services/cluster
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

| NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format). |

**Sample request**

The following command lists the configuration options.
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/local_services/cluster

Response

The following is a sample response received when listing the configuration options.

For details of the meta object, see Message format on page 10.

```json
{
    "body": {
        "enabled": true,
        "listen_address": {
            "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.2553887595ce3ca7f1eae4",
            "meta": {
                "href": "/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/2553887595ce3ca7f1eae4"
            }
        },
        "key": "cluster",
        "meta": {
            "first": "/api/configuration/local_services/admin_web",
            "href": "/api/configuration/local_services/cluster",
            "last": "/api/configuration/local_services/user_web",
            "next": "/api/configuration/local_services/indexer",
            "parent": "/api/configuration/local_services",
            "previous": "/api/configuration/local_services/analytics",
            "remaining_seconds": 600,
            "transaction": "/api/transaction"
        }
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>By default, this option is set to false. Set it to true to enable the cluster interface.</td>
</tr>
<tr>
<td>listen_address</td>
<td>Top level element (string)</td>
<td>Contains the key of the network interface that is used as the cluster interface.</td>
</tr>
</tbody>
</table>

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
### System backup policy

System backup uses a backup policy to create a snapshot of the configuration of One Identity Safeguard for Privileged Sessions (SPS) to a remote backup server. For details on how backup policies work, see "Data and configuration backups" in the Administration Guide. For details on configuring a backup policy using the REST API, see Backup policy. To encrypt the backup, see Encrypting system backup policy.

#### URL

GET https://<IP-address-of-SPS>/api/configuration/management/system_backup

#### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
</table>
| session_id  | Contains the authentication token of the user | Required | The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format)
Sample request

The following command lists the system backup settings of SPS.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/system_backup
```

Response

The following is a sample response received when listing the endpoints for date and time settings.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "backup_policy": "<key-to-a-backup-policy>"
    },
    "key": "system_backup",
    "meta": {
        "first": "https://<IP-address-of-SPS>/api/configuration/management/certificates",
        "href": "https://<IP-address-of-SPS>/api/configuration/management/system_backup",
        "last": "https://<IP-address-of-SPS>/api/configuration/management/webinterface",
        "parent": "https://<IP-address-of-SPS>/api/configuration/management",
        "previous": "https://<IP-address-of-SPS>/api/configuration/management/syslog",
        "remaining_seconds": 600,
        "transaction": "https://<IP-address-of-SPS>/api/transaction"
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>backup_policy</td>
<td>Contains the ID of the backup policy to use for system backups. For details on configuring a backup policy using the REST API, see Backup policy.</td>
</tr>
</tbody>
</table>

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Encrypting system backup policy**

System backup uses a backup policy to create a snapshot of the configuration of One Identity Safeguard for Privileged Sessions (SPS) to a remote backup server. For details on how backup policies work, see "Data and configuration backups" in the Administration Guide. For details on configuring a backup policy using the REST API, see Backup policy. This section describes how to create encrypted system backups.

**URL**

GET https://<IP-address-of-SPS>/api/management/exported_configuration_encryption

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the system backup settings of SPS.
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/exported_configuration_encryption

Response

The following is a sample response received when listing the endpoints for date and time settings.

For details of the meta object, see Message format on page 10.

```
{
   "body": {
      "encryption": {
         "enabled": true,
         "gpg_public_key": {
            "fingerprint": "2F2E3967EDAD2F288E54EE8693B99C4F545B7670",
            "public_key": "-----BEGIN PGP PUBLIC KEY BLOCK-----
mQGNBF3rnZ0BDADHdz5/kCkr17T8w861AGGXdGK/1wxunjTCx6tfhSsFREwmKjhfr\nYTlNxsodALXtAphHeNAeUwwXjYDJe1ALMvCrVtLp7Ht8tqnmnt2NwUsmffIF3ga\nnd10sH2UjT5x6AajKvFwH5xkS0QHiLFrUT5Wd0ucT5t8jEdj80A7Z6.....
CT1WwbMg5VoXQ3Rpp8evcUTzy3+ra/GosCSaFSrE31pyXkULB9+EAU7W\n23YDiM21csIaqX+XDGMe5Hq4PMho7cqSMyb\n
-----END PGP PUBLIC KEY BLOCK-----",
            "uids": [
               "Demo User <example@oneidentity.com>",
            ]
         }
      },
      "key": "exported_configuration_encryption",
      "meta": {
         "first": "/api/configuration/management/certificates",
         "href": "/api/configuration/management/exported_configuration_encryption",
         "last": "/api/configuration/management/webinterface",
         "next": "/api/configuration/management/health_monitoring",
         "parent": "/api/configuration/management",
         "previous": "/api/configuration/management/email",
         "remaining_seconds": 600,
         "transaction": "/api/transaction"
      }
   }
```

SPS 6.13.0 REST API Reference Guide
Basic settings

114
<table>
<thead>
<tr>
<th>Elements of encryption</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-y-p-e</td>
<td>Defines encryption settings for system backups.</td>
</tr>
<tr>
<td>en</td>
<td>T-ry-pe</td>
</tr>
<tr>
<td>cr</td>
<td>o-p</td>
</tr>
<tr>
<td>yp</td>
<td>l-e-v-e-l</td>
</tr>
<tr>
<td>ti</td>
<td>e-l-e-m-e-n-t</td>
</tr>
<tr>
<td>on</td>
<td></td>
</tr>
</tbody>
</table>

**en**

When set to True, enables encryption of the system backups. Enabling encryption requires setting the `gpg_public_key` option.

**gpg**

Contains the fingerprint, `public_key`, and the list of uids of the GPG public key used to encrypt system backups. For example:

```json
"gpg_public_key": {
  "fingerprint": "2F2E3967EDAD2F288E54EE8693B99C4F545B7670",
  "public_key": "BEGIN PGP PUBLIC KEY BLOCK-----
\nmQGNBF3rnZ0BDADHdz5/kCkr17T8w861AGGXdGK/1-
wxunTCx6tf-
hSsFREWmKjh-
fr\nYTLNxs-
odALXtAphHeNAeUwwrXjYDje1ALMvCDrVtLp7Ht8tq-
mNt2NwUSm-
ffIF3ga\nD10sH2ujT5Xt6XAJeKvFfWeH5xk5QHIClFLfUT5wDoUcTEsR8jEd-
j80A7Z6hKyF29g\...
R40Niv4Ge6aYNeDr-
k3yTBo6bBYDR7NKA70REXCFqcyCeYB121UQ\n-bb5aTZAa8W808IRmy-
bxpRxAaHZX8apBgDLKwW48kL0nOC907htgcyY1spzG7z7i\nTryxLBl/CT1W-
```
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Date and time

Date & time

Contains the endpoints for configuring date and time on SPS.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/datetime
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists endpoints for configuring date and time settings on SPS.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/datetime
```

Response

The following is a sample response received when listing the endpoints for date and time settings.

For details of the meta object, see **Message format** on page 10.

```
{
   "items": [
   {
      "key": "ntp_servers",
      "meta": {
         "href": "/api/configuration/datetime/ntp_servers"
      }
   },
   {
      "key": "timezone",
      "meta": {
         "href": "/api/configuration/datetime/timezone"
      }
   }
   ],
   "meta": {
      "first": "/api/configuration/aaa",
```

SPS 6.13.0 REST API Reference Guide
Basic settings
Element                  Description
----------------------------------------------------------
ntp_servers             NTP server addresses.
timezone               Timezone settings.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

NTP servers

This endpoint contains NTP server addresses.

URL

GET https://<IP-address-of-SPS>/api/configuration/datetime/ntp_servers
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists NTP server addresses.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/datetime/ntp_servers
```

Response

The following is a sample response received when listing NTP server addresses.

For details of the meta object, see Message format on page 10.

```
{
   "body": [
      {
         "selection": "fqdn",
         "value": "pool.ntp.org"
      }
   ],
   "key": "ntp_servers",
   "meta": {
      "first": "https://<IP-address-of-SPS>/api/configuration/datetime/ntp_servers",
      "href": "https://<IP-address-of-SPS>/api/configuration/datetime/ntp_servers",
      "last": "https://<IP-address-of-SPS>/api/configuration/datetime/timezone",
      "next": "https://<IP-address-of-SPS>/api/configuration/datetime/timezone"
   }
}
```
"parent": "/api/configuration/datetime",
"previous": null,
"transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (list)</td>
<td>Contains the list of NTP server addresses.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the address type (IP or domain name). Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fqdn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The NTP server address is provided as a fully qualified domain name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ip</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The NTP server address is provided as an IP address.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The address of the NTP server.</td>
</tr>
</tbody>
</table>

**Add an NTP server**

To add an NTP server's address, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new NTP server address list.**
   
   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/datetime/ntp_servers` endpoint. You can find a detailed description of the available parameters listed in **Element**.

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

**Modify an NTP server address**

To modify an NTP server's address, you have to:
1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the NTP server address list.**
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/datetime/ntp_servers endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

### Status and error codes
The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Timezone
Configures the time zone.

### URL

```
GET https://<IP-address-of-SPS>/api/configuration/datetime/timezone
```
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command displays the configured time zone.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/datetime/timezone
```

Response

The following is a sample response received when querying the configured time zone.

For details of the meta object, see Message format on page 10.

```
{
  "body": "America/New_York",
  "key": "timezone",
  "meta": {
    "first": "/api/configuration/datetime/ntp_servers",
    "href": "/api/configuration/datetime/timezone",
    "last": "/api/configuration/datetime/timezone",
    "next": null,
    "parent": "/api/configuration/datetime",
    "previous": "/api/configuration/datetime/ntp_servers",
    "transaction": "/api/transaction"
  }
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>string</td>
<td>Contains the configured time zone. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Abidjan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Accra</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Addis_Ababa</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Algiers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Asmara</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Asmera</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Bamako</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Bangui</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Banjul</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Bissau</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Blantyre</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Brazzaville</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Bujumbura</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Africa/Cairo</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Africa/Casablanca</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Ceuta</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Conakry</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Dakar</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Dar_es_Salaam</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Djibouti</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Douala</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/El_Aaiun</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Freetown</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Gaborone</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Harare</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Johannesburg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Kampala</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Khartoum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Kigali</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Kinshasa</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Lagos</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Africa/Libreville</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>t</td>
<td></td>
<td>Africa/Ouagadougou</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Africa/Porto-Novó</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Africa/São_Tome</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Africa/Timbuktu</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Africa/Tripoli</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Africa/Tunís</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Africa/Windhoek</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Adak</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Anchorage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Anguilla</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Antigua</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Araguaína</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Argentina/Buenos_Aires</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Argentina/Catamarca</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Argentina/ComodRivadavia</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Argentina/Cordoba</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Argentina/Jujuy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Argentina/La_Rioja</td>
</tr>
<tr>
<td></td>
<td></td>
<td>America/Argentina/Mendoza</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>America/El_Salvador</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Ensenada</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Fort_Wayne</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Fortaleza</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Glace_Bay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Godthab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Goose_Bay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Grand_Turk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Grenada</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Guadeloupe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Guatemala</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Guayaquil</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Guyana</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Halifax</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Havana</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Hermosillo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Indiana/Indianapolis</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Indiana/Knox</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Indiana/Marengo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Indiana/Petersburg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Indiana/Tell_City</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Indiana/Vevay</td>
<td></td>
<td></td>
</tr>
<tr>
<td>America/Indiana/Vincennes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------</td>
<td>---------------</td>
</tr>
<tr>
<td>a/Baghdad</td>
<td></td>
<td>Asia/Bahrain</td>
</tr>
<tr>
<td>Asia/Bahrain</td>
<td></td>
<td>Asia/Baku</td>
</tr>
<tr>
<td>Asia/Baku</td>
<td></td>
<td>Asia/Bangkok</td>
</tr>
<tr>
<td>Asia/Bangkok</td>
<td></td>
<td>Asia/Beirut</td>
</tr>
<tr>
<td>Asia/Beirut</td>
<td></td>
<td>Asia/Bishkek</td>
</tr>
<tr>
<td>Asia/Bishkek</td>
<td></td>
<td>Asia/Brunei</td>
</tr>
<tr>
<td>Asia/Brunei</td>
<td></td>
<td>Asia/Calcutta</td>
</tr>
<tr>
<td>Asia/Calcutta</td>
<td></td>
<td>Asia/Choibalsan</td>
</tr>
<tr>
<td>Asia/Choibalsan</td>
<td></td>
<td>Asia/Chongqing</td>
</tr>
<tr>
<td>Asia/Chongqing</td>
<td></td>
<td>Asia/Chungking</td>
</tr>
<tr>
<td>Asia/Chungking</td>
<td></td>
<td>Asia/Colombo</td>
</tr>
<tr>
<td>Asia/Colombo</td>
<td></td>
<td>Asia/Dacca</td>
</tr>
<tr>
<td>Asia/Dacca</td>
<td></td>
<td>Asia/Damascus</td>
</tr>
<tr>
<td>Asia/Damascus</td>
<td></td>
<td>Asia/Dhaka</td>
</tr>
<tr>
<td>Asia/Dhaka</td>
<td></td>
<td>Asia/Dili</td>
</tr>
<tr>
<td>Asia/Dili</td>
<td></td>
<td>Asia/Dubai</td>
</tr>
<tr>
<td>Asia/Dubai</td>
<td></td>
<td>Asia/Dushanbe</td>
</tr>
<tr>
<td>Asia/Dushanbe</td>
<td></td>
<td>Asia/Gaza</td>
</tr>
<tr>
<td>Asia/Gaza</td>
<td></td>
<td>Asia/Harbin</td>
</tr>
<tr>
<td>Asia/Harbin</td>
<td></td>
<td>Asia/Hong_Kong</td>
</tr>
<tr>
<td>Asia/Hong_Kong</td>
<td></td>
<td>Asia/Hovd</td>
</tr>
<tr>
<td>Asia/Hovd</td>
<td></td>
<td>Asia/Irkutsk</td>
</tr>
<tr>
<td>Asia/Irkutsk</td>
<td></td>
<td>Asia/Istanbul</td>
</tr>
<tr>
<td>Asia/Istanbul</td>
<td></td>
<td>Asia/Jakarta</td>
</tr>
<tr>
<td>Asia/Jakarta</td>
<td></td>
<td>Asia/Jayapura</td>
</tr>
<tr>
<td>Asia/Jayapura</td>
<td></td>
<td>Asia/Jerusalem</td>
</tr>
<tr>
<td>Asia/Jerusalem</td>
<td></td>
<td>Asia/Kabul</td>
</tr>
<tr>
<td>Asia/Kabul</td>
<td></td>
<td>Asia/Kamchatka</td>
</tr>
<tr>
<td>Asia/Kamchatka</td>
<td></td>
<td>Asia/Karachi</td>
</tr>
<tr>
<td>Asia/Karachi</td>
<td></td>
<td>Asia/Kashgar</td>
</tr>
</tbody>
</table>
Modify the time zone

To modify time zone, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the body element.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/datetime/timezone` endpoint. You can find a detailed description of the available time zone values listed in [Element](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes](#) on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Logs, monitoring and alerts

Management options

Contains the configuration endpoints for managing SPS.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/management

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists management configuration endpoints.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management
```

**Response**

The following is a sample response received when listing management endpoints. For details of the meta object, see Message format on page 10.
{
  "items": [
    {
      "key": "certificates",
      "meta": {
        "href": "/api/configuration/management/certificates"
      }
    },
    {
      "key": "disk_fillup_prevention",
      "meta": {
        "href": "/api/configuration/management/disk_fillup_prevention"
      }
    },
    {
      "key": "email",
      "meta": {
        "href": "/api/configuration/management/email"
      }
    },
    {
      "key": "exported_configuration_encryption",
      "meta": {
        "href": "/api/configuration/management/exported_configuration_encryption"
      }
    },
    {
      "key": "health_monitoring",
      "meta": {
        "href": "/api/configuration/management/health_monitoring"
      }
    },
    {
      "key": "license",
      "meta": {
        "href": "/api/configuration/management/license"
      }
    },
    {
      "key": "root_password",
      "meta": {
        "href": "/api/configuration/management/root_password"
      }
    },
    {
      "key": "snmp",
      "meta": {
        "href": "/api/configuration/management/snmp"
      }
    }
  ]
}
"href": "/api/configuration/management/snmp"
}
],
"meta": {
"first": "/api/configuration/aaa",
"href": "/api/configuration/management"
}
Endpoints

<table>
<thead>
<tr>
<th>Endpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificates</td>
<td>References the certificates of SPS’s internal Certificate Authority,</td>
</tr>
<tr>
<td></td>
<td>Timestamping Authority, and the SSL certificate of the web interface.</td>
</tr>
<tr>
<td>disk_fillup_prevention</td>
<td>Disk fill-up prevention.</td>
</tr>
<tr>
<td>email</td>
<td>SMTP server address and authentication, administrator e-mail, and</td>
</tr>
<tr>
<td></td>
<td>e-mail addresses for alerts and reports.</td>
</tr>
<tr>
<td>exported_configuration_encryption</td>
<td>SMTP server address and authentication, administrator e-mail, and</td>
</tr>
<tr>
<td></td>
<td>e-mail addresses for alerts and reports.</td>
</tr>
<tr>
<td>health_monitoring</td>
<td>Configuration settings for monitoring the utilization of SPS.</td>
</tr>
<tr>
<td>snmp</td>
<td>SNMP settings.</td>
</tr>
<tr>
<td>syslog</td>
<td>Syslog server address and authentication.</td>
</tr>
<tr>
<td>webinterface</td>
<td>Configuration settings for the SPS web interface.</td>
</tr>
<tr>
<td>web_gateway_authentication</td>
<td>Configuration of the banner on the login screen before web gateway</td>
</tr>
<tr>
<td></td>
<td>authentication.</td>
</tr>
</tbody>
</table>

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>--------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Web gateway authentication**

Use the `/web_gateway_authentication` endpoint to configure the banner on the login screen before web gateway authentication in order to make users aware of being audited before logging in, or that they are accessing a government website, and so on.

**URL**

| HTTP method | URL | 
|-------------|-----|---|
| PUT         | `https://<IP-address-of-SPS>/api/configuration/management/web_gateway_authentication` |

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Operations with the `/web_gateway_authentication` endpoint include:**

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring the web gateway authentication banner</td>
<td>PUT</td>
<td><code>/api/configure/management/web_gateway_authentication</code></td>
</tr>
<tr>
<td>Querying the web gateway authentication banner</td>
<td>GET</td>
<td><code>/api/configure/management/web_gateway_authentication</code></td>
</tr>
</tbody>
</table>
Sample request

The following command configures the web gateway authentication banner.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/web_gateway_authentication
```

Response

The following is a sample response when the configured web gateway authentication banner is queried.

For details of the meta object, see Message format on page 10.

```json
{
    "key": "web_gateway_authentication",
    "body": {
        "auto_assign": false,
        "banner": {
            "enabled": true,
            "text": "Web gateway\nBANNER"
        }
    }
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>object</td>
<td>Top level element.</td>
</tr>
<tr>
<td>body.auto_assign</td>
<td>boolean</td>
<td>Enable auto-assign to permit your users to authenticate on the SPS web interface once, and open sessions without repeating the gateway authentication.</td>
</tr>
<tr>
<td>body.banner</td>
<td>object</td>
<td>Top level element.</td>
</tr>
<tr>
<td>body.banner.enabled</td>
<td>boolean</td>
<td>Shows whether the banner is enabled or not.</td>
</tr>
<tr>
<td>body.banner.text</td>
<td>string</td>
<td>The text of the banner.</td>
</tr>
</tbody>
</table>

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>InvalidRequestBody</td>
<td>The request body sent by the user has an invalid format. This may be an error with the encoding or the body is not a properly encoded JSON value.</td>
</tr>
<tr>
<td>400</td>
<td>SemanticError</td>
<td>The configuration contains semantic errors, inconsistencies or other problems that would put the system into an unreliable state if the configuration had been applied. The details section contains the errors that were found in the configuration.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
</tbody>
</table>

**Syslog server settings**

SPS can send its system log messages to remote syslog servers, for example, syslog-ng Premium Edition, syslog-ng Store Box, Splunk, or HPE ArcSight Data Platform.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/management/syslog

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connec-
### Sample request

The following command lists the syslog server settings.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/syslog
```

### Response

The following is a sample response received when listing syslog server settings.

For details of the `meta` object, see Message format on page 10.

```
{
   "body": {
      "certificates": {
         "ca": "<ca-cert>",
         "client": {
            "key": "191725ec-b71b-47ab-9e87-561a5d9e2bb7",
            "meta": {
               "href": "/api/configuration/x509/191725ec-b71b-47ab-9e87-561a5d9e2bb7"
            }
         }
      },
      "include_node_id": true,
      "receivers": [
         {
            "address": {
               "selection": "ip",
               "value": "10.20.30.40"
            },
            "port": 514,
            "protocol": {
               "ip_protocol": "tcp",
               "protocol_type": "legacy-bsd",
               "tls_enabled": false
            }
         }
      ],
      "server_key_check": "optional-trusted"
   }
}
```
Element | Type | Description
--- | --- | ---
key | string | Top level element, contains the ID of the endpoint.
body | Top level element (string) | Contains the syslog server configuration settings.
certificates | Top level item | Contains the certificates of the client (SPS), and the certificate of the CA.
ca | string | The CA certificate of the Certificate Authority. Configure this option if the value of the tls_enabled element is set to true.
client | string | Configure this option if the value of the tls_enabled element is set to true, and the syslog server requires mutual authentication. Otherwise, set its value to null.

References the identifier of the client's (SPS's) X.509 certificate. You can configure certificates at the /api/configuration/x509/ endpoint.

To modify or add an X.509 certificate, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key).

include_node_id | boolean | Set to true to display separate hostnames for syslog messages sent by the nodes of a SPS HA cluster.
The node ID included in the hostname field of the syslog message is the MAC address of the node's HA interface. Messages of the core firmware are always sent by the master node.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>receivers</td>
<td>Top level</td>
<td>Contains the addresses of the syslog servers.</td>
</tr>
<tr>
<td>server_key_check</td>
<td>string</td>
<td>Configures validating the syslog server's certificate with the CA. The following values are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• optional-trusted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the server sends a certificate, SPS checks if it is valid (not expired) and that the Common Name of the certificate contains the domain name or the IP address of the server. If these checks fail, SPS rejects the connection. However, SPS accepts the connection if the server does not send a certificate.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• optional-untrusted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPS accepts any certificate shown by the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• required-trusted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPS verifies the certificate shown by the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• required-untrusted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPS requests a certificate from the server, and rejects the connection if no certificate is received, if the certificate is not valid (expired), or if the Common Name of the certificate does not contain the domain name or the IP address of the server.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of receivers</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>Top level</td>
<td>Contains the address of the syslog server.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the address type (IP or domain name).</td>
</tr>
<tr>
<td>Elements of receivers</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>value</strong></td>
<td>string</td>
<td>The address of the syslog server, corresponding to the format set in the selection field.</td>
</tr>
<tr>
<td><strong>port</strong></td>
<td>int</td>
<td>The port of the syslog server.</td>
</tr>
<tr>
<td><strong>protocol</strong></td>
<td>Top level item</td>
<td>Contains the syslog protocol settings.</td>
</tr>
<tr>
<td><strong>ip_protocol</strong></td>
<td>string</td>
<td>Configures the IP protocol. The following options are available:</td>
</tr>
</tbody>
</table>
|                       |       | - tcp  
|                       |       |  TCP protocol. |
|                       |       | - udp  
|                       |       |  UDP protocol. |
| **protocol_type**     | string | Configures the syslog protocol. The following options are available: |
|                       |       | - legacy-bsd  
|                       |       |  BSD-syslog protocol. |
|                       |       | - syslog  
|                       |       |  IETF-syslog protocol. |
| **tls_enabled**       | string | Set to true to enable TLS encryption. |
|                       |       |  If TLS is enabled, the value of the ca and client elements cannot be null. |

**Examples:**

Default settings: no external syslog servers.
Upload CA certificates

SPS uses only the key part of the CA certificate.

You can choose to upload a single certificate or a certificate chain.

To use a certificate with the SPS API, remove all data, and substitute line breaks with \n. The same is true for a certificate chain: copy individual certificates one after the other, and substitute line breaks with \n.

The following is an example certificate, as used on the SPS web interface:

```
-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFAAODCbjzELMAkGA1UEBhMC
Q0ExEDA0BgNvBAsTBIrU0MelGkqj9Y02R5zz33/v2G7a0g/0zCRpW1I7qHh7
-----END CERTIFICATE-----
```

The same certificate, as accepted by the SPS API:

```
-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFAAODCbjzELMAkGA1UEBhMC
Q0ExEDA0BgNvBAsTBIrU0MelGkqj9Y02R5zz33/v2G7a0g/0zCRpW1I7qHh7
-----END CERTIFICATE-----
```
Modify syslog server settings

To modify the syslog server settings, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the endpoint.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/management/syslog endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Disk fill-up prevention

Contains the configuration options for preventing disk fill-up.

#### URL

GET https://<IP-address-of-SPS>/api/configuration/management/disk_fillup_prevention

#### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API on page 19</a>.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

#### Sample request

The following command lists disk fill-up prevention options.
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/disk_fillup_prevention

**Response**

The following is a sample response received when listing disk fill-up prevention settings. For details of the meta object, see *Message format* on page 10.

```
{
    "body": {
        "archiving_enabled": false,
        "enabled": true,
        "used_space_ratio_limit": 80
    },
    "key": "disk_fillup_prevention",
    "meta": {
        "first": "/api/configuration/management/certificates",
        "href": "/api/configuration/management/disk_fillup_prevention",
        "last": "/api/configuration/management/webinterface",
        "next": "/api/configuration/management/email",
        "parent": "/api/configuration/management",
        "previous": "/api/configuration/management/certificates",
        "transaction": "/api/transaction"
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration settings for disk fill-up prevention.</td>
</tr>
<tr>
<td>archiving_enabled</td>
<td>boolean</td>
<td>Set to true to automatically start all configured archiving/cleanup jobs when disk usage goes over the value of the used_space_ratio_limit element.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable disk fill-up prevention.</td>
</tr>
<tr>
<td>used_space_ratio_limit</td>
<td>int</td>
<td>Disk utilization limit, in percent. When used disk space reaches this limit, SPS disconnects all clients. Set to 0 to turn the feature off.</td>
</tr>
</tbody>
</table>

**Modify disk fill-up prevention settings**

To modify the disk fill-up prevention settings, you have to:
1. **Open a transaction.**
   For more information, see *Open a transaction* on page 29.

2. **Modify the JSON object of the disk fill-up configuration endpoint.**
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/management/disk_fillup_prevention` endpoint. You can find a detailed description of the available parameters listed in *Element*.

3. **Commit your changes.**
   For more information, see *Commit a transaction* on page 31.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see *Application level error codes* on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Mail settings

Configuration settings for SMTP server address and authentication, administrator e-mail, and e-mail addresses for alerts and reports.

### URL

```
GET https://<IP-address-of-SPS>/api/configuration/management/email
```
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists mail settings.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/email
```

Response

The following is a sample response received when listing mail settings.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "admin_address": "<admin-email>",
        "alerting_address": "<alerts-target-email>",
        "reporting_address": "<reports-target-email>",
        "sender_address": null,
        "smtp_auth": {
            "enabled": false
        },
        "smtp_encryption": {
            "selection": "disabled"
        },
        "smtp_server": {
            "selection": "ip",
            "value": "<smtp-server-ip>"
        }
    },
```
"key": "email",
"meta": {
  "first": "/api/configuration/management/certificates",
  "href": "/api/configuration/management/email",
  "last": "/api/configuration/management/webinterface",
  "next": "/api/configuration/management/health_monitoring",
  "parent": "/api/configuration/management",
  "previous": "/api/configuration/management/disk_fillup_prevention",
  "transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration options for e-mail.</td>
</tr>
<tr>
<td>admin_address</td>
<td>string</td>
<td>The e-mail address of the administrator of SPS.</td>
</tr>
<tr>
<td>alerting_address</td>
<td>string</td>
<td>The e-mail address where monitoring alerts are sent.</td>
</tr>
<tr>
<td>reporting_address</td>
<td>string</td>
<td>The e-mail address where traffic reports are sent.</td>
</tr>
<tr>
<td>sender_address</td>
<td>string</td>
<td>The address of the sender (SPS).</td>
</tr>
<tr>
<td>smtp_auth</td>
<td>Top level item</td>
<td>Configures authentication to the SMTP server.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable authenticating to the SMTP server.</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>References the password of the authenticating user. You configure passwords at the /api/configuration/passwords/ endpoint. To modify or add a password, use the value of the returned key as the value of the password element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>username</td>
<td>string</td>
<td>The username for authenticating to the SMTP server.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>smtp_encryption</td>
<td>Top level item</td>
<td>Configuration settings for encrypting the communication between SPS and the SMTP server.</td>
</tr>
<tr>
<td>smtp_server</td>
<td>Top level item</td>
<td>Contains the address of the SMTP server.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the address type (IP or domain name). Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fqdn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The SMTP server address is provided as a fully qualified domain name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ip</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The SMTP server address is provided as an IP address.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The address of the SMTP server.</td>
</tr>
</tbody>
</table>

**Elements of smtp_encryption**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>client_authentication</td>
<td>Top level item</td>
<td>Configures authenticating as a client with an X.509 certificate.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Can only be enabled if the value of the selection element is set to starttls.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable authenticating as a client with an X.509 certificate.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Can only be enabled if the value of the selection element of smmp_encryption is set to starttls.</td>
</tr>
<tr>
<td>x509_identity</td>
<td></td>
<td>References the identifier of the authenticating client's X.509 certificate. You can configure certificates at the/api/configuration/x509/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an X.509 certificate, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key). For details, see Certificates stored on SPS on page 287.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures encrypted communication with the SMTP server. The following values are possible:</td>
</tr>
</tbody>
</table>
Elements of `smtp_encryption` | Type | Description
--- | --- | ---
| | | • disabled
Disables e-mail encryption.
• starttls
Enables STARTTLS encryption.

**server_certificate_check** | **Top level item** | Configuration settings for validating the SMTP server's certificate.
Can only be enabled if the value of the selection element is set to starttls.

**enabled** | **boolean** | Set to true to enable validating the SMTP server's certificate.
Can only be enabled if the value of the selection element of `smtp_encryption` is set to starttls.

**server_certificate_ca** | **string** | The CA certificate of the Certificate Authority.

**Examples:**
Enable authentication to the SMTP server.

```json
{
  "admin_address": "<admin-email>",
  "alerting_address": "<alerts-target-email>",
  "reporting_address": "<reports-target-email>",
  "sender_address": null,
  "smtp_auth": {
    "enabled": true,
    "password": {
      "key": "aec663b5-f5bd-4c93-bb51-36fea3328e58",
      "meta": {
        "href": "/api/configuration/passwords/aec663b5-f5bd-4c93-bb51-36fea3328e58"
      }
    },
    "username": "<smtp-username>"
  },
  "smtp_encryption": {
    "selection": "disabled"
  }
}
```
Configure STARTTLS encryption without certificate checks.

```json
{
  "smtp_server": {
    "selection": "ip",
    "value": "<smtp-server-ip>"
  }
}
```

Configure STARTTLS encryption with server certificate check, and authenticate as client with an X.509 certificate.

```json
{
  "admin_address": "<admin-email>",
  "alerting_address": "<alerts-target-email>",
  "reporting_address": "<reports-target-email>",
  "sender_address": null,
  "smtp_auth": {
    "enabled": true,
    "password": {
      "key": "0210848a-b301-47d5-9023-779c5fe951f7",
      "meta": {
        "href": "/api/configuration/passwords/0210848a-b301-47d5-9023-779c5fe951f7"
      }
    },
    "username": "<smtp-username>"
  },
  "smtp_encryption": {
    "client_authentication": {
      "enabled": false
    },
    "selection": "starttls",
    "server_certificate_check": {
      "enabled": false
    }
  }
}
```
"password": {
  "key": "37716c4f-759d-4900-9740-ea22211498cf",
  "meta": {
    "href": "/api/configuration/passwords/37716c4f-759d-4900-9740-ea22211498cf"
  },
  "username": "<smtp-username>"
},
"smtp_encryption": {
  "client_authentication": {
    "enabled": true,
    "x509_identity": {
      "key": "c3a23e32-d75b-461e-afc0-14d1f6692879",
      "meta": {
        "href": "/api/configuration/x509/c3a23e32-d75b-461e-afc0-14d1f6692879"
      }
    }
  },
  "selection": "starttls",
  "server_certificate_check": {
    "enabled": true,
    "server_certificate_ca": "<ca-cert>"
  }
},
"smtp_server": {
  "selection": "ip",
  "value": "<smtp-server-ip>"
}

CA certificates

CA certificates must not contain any metadata. SPS uses only the key part of the certificate.

To use a certificate with the SPS API, remove all metadata, and substitute line breaks with \n.

The following is an example certificate, as used on the SPS web interface:

```
-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCDSdc5360b5tPQTANBgkhkiG9w0BAQUFADCBjzEMLMAkGA1UEBhMCQ0Ex
EoYDQ0IwJwYXJlMRAwDgYDVQQKMRkwZWNpdHlkMRAwDgYDVQQHEDEIc2Vjb21JaGFy
aW9uMRAwDgYDVQQLEmltYWdlc3RvZ3JhbWV0aW9uMRAwDgYDVQQKEw1Eb2N1bWUx
iCBhY2Nlc3Npb24wDQYJKoZIhvcNBDUyMDIyMDIyMTIwMDIwMDBiNzYwMjI0MDAw
MTIwMDIyMDIyMDIyMTIwMDIwMDBiNzYwMjI0MDAw
-----END CERTIFICATE-----
```
The same certificate, as accepted by the SPS API:

"certificate": "-----BEGIN CERTIFICATE-----

MIIDnDCCAoQCCQDQc3S6Ob5tPQTANBgkqghkiG9w0BAQUFADCBjzELMAkGA1UEBhMC

Q0ExEDEy0DVMBgNV

BAMVMwwgGgYD

MVYz

"-----END CERTIFICATE-----"

Modify mail settings

To modify mail settings, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the endpoint.**

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/management/email endpoint. You can find a detailed SPS 6.13.0 REST API Reference Guide Basic settings
description of the available parameters listed in Element.

3. **Commit your changes.**

For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Health monitoring**

Configuration settings for monitoring the utilization of SPS.

**URL**

```plaintext
GET https://<IP-address-of-SPS>/api/configuration/management/health_monitoring
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860.</td>
</tr>
</tbody>
</table>
### Sample request

The following command lists health monitoring settings.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/health_monitoring
```

### Response

The following is a sample response received when listing health monitoring settings.

For details of the meta object, see Message format on page 10.

```json
{
   "body": {
      "maximum_disk_utilization_ratio": 80,
      "maximum_load1": null,
      "maximum_load15": null,
      "maximum_load5": null,
      "maximum_swap_utilization_ratio": 70
   },
   "key": "health_monitoring",
   "meta": {
      "first": "/api/configuration/management/certificates",
      "href": "/api/configuration/management/health_monitoring",
      "last": "/api/configuration/management/webinterface",
      "next": "/api/configuration/management/snmp",
      "parent": "/api/configuration/management",
      "previous": "/api/configuration/management/email",
      "transaction": "/api/transaction"
   }
}
```

For details on authentication, see Authenticate to the SPS REST API on page 19.

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>Contains health monitoring settings.</td>
</tr>
<tr>
<td></td>
<td>element (string)</td>
<td></td>
</tr>
<tr>
<td>maximum_disk_utilization_ratio</td>
<td>int</td>
<td>The highest allowed value for disk utilization (in %).</td>
</tr>
<tr>
<td>maximum_load1</td>
<td>int</td>
<td>Average maximum for load for 1 minute.</td>
</tr>
<tr>
<td>maximum_load15</td>
<td>int</td>
<td>Average maximum load for 15 minutes.</td>
</tr>
<tr>
<td>maximum_load5</td>
<td>int</td>
<td>Average maximum load for 5 minutes.</td>
</tr>
<tr>
<td>maximum_swap_utilization_ratio</td>
<td>int</td>
<td>The highest allowed value for swap utilization (in %).</td>
</tr>
</tbody>
</table>

### Modify health monitoring settings

To modify health monitoring settings, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the endpoint.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/management/health_monitoring` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes](#) on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the</td>
</tr>
</tbody>
</table>
### Code | Description | Notes
--- | --- | ---
client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.

| 403 | Unauthorized | The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved. |
| 404 | Not Found | The requested object does not exist. |

### SNMP settings

Contains the configuration endpoints for SNMP settings.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/management/snmp

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d03e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format). |

### Sample request

The following command lists the endpoints for SNMP configuration settings.
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/snmp

Response

The following is a sample response received when listing SNMP configuration endpoints. For details of the meta object, see Message format on page 10.

```
{
    "items": [      
        {
            "key": "trap",
            "meta": {
                "href": "/api/configuration/management/snmp/trap"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/management/certificates",
        "href": "/api/configuration/management/snmp",
        "last": "/api/configuration/management/webinterface",
        "next": "/api/configuration/management/splunk_forwarder",
        "parent": "/api/configuration/management",
        "previous": "/api/configuration/management/health_monitoring",
        "transaction": "/api/transaction"
    }
}
```

### Element Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trap</td>
<td>Configuration settings for SNMP traps.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section</td>
</tr>
</tbody>
</table>
contains the path that was attempted to be accessed, but could not be retrieved.

404  NotFound  The requested object does not exist.

**SNMP traps**

Configuration settings for the address and protocol of the SNMP server.

**URL**

GET `https://<IP-address-of-SPS>/api/configuration/management/snmp/trap`

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the configuration of the SNMP server.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/snmp/trap
```

**Response**

The following is a sample response received when listing the address and protocol settings of the SNMP server.

For details of the meta object, see **Message format** on page 10.
```json
{
  "body": {
    "enabled": true,
    "version": {
      "selection": "2c",
      "value": {
        "community": "public",
        "server": {
          "selection": "ip",
          "value": "10.20.30.40"
        }
      }
    }
  },
  "key": "trap",
  "meta": {
    "first": "/api/configuration/management/snmp/trap",
    "href": "/api/configuration/management/snmp/trap",
    "last": "/api/configuration/management/snmp/trap",
    "next": null,
    "parent": "/api/configuration/management/snmp",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the address and protocol settings of the SNMP server.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to send alerts to an SNMP server.</td>
</tr>
<tr>
<td>version</td>
<td>Top level item</td>
<td>Contains the configuration settings for the server address, and the SNMP protocol.</td>
</tr>
</tbody>
</table>

**Elements of version**

<table>
<thead>
<tr>
<th>Selection</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the SNMP protocol to use. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 2c Configures version 2c of the SNMP protocol.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 3</td>
</tr>
<tr>
<td><strong>Elements of version</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td>value</td>
<td>Top level item</td>
<td>Configures version 3 of the SNMP protocol. Contains the SNMP server address, and the protocol-specific settings.</td>
</tr>
<tr>
<td>auth_method</td>
<td>string</td>
<td>Required parameter when using SNMP version 3. Configures encrypted communication with the SNMP server. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- md5: Use MD5 encryption. The auth_password element must reference a valid password.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- sha1: Use SHA1 encryption. The auth_password element must reference a valid password.</td>
</tr>
<tr>
<td>auth_password</td>
<td>string</td>
<td>Required parameter when using SNMP version 3. References the password used for authenticating to the SNMP server. You can create passwords at the <code>/api/configuration/passwords/</code> endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a password, use the value of the returned key as the value of the password element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The referenced password must be at least 8 characters long, and can contain letters (a-z, A-Z), numbers (0-9) the special characters (&quot;#$%&amp;'()<em>+,-.;&lt;=@[]^<code>{}</code></em>/?-) and the space character.</td>
</tr>
<tr>
<td>community</td>
<td>string</td>
<td>Must be used if version 2c of the SNMP protocol is configured. The name of the SNMP community.</td>
</tr>
<tr>
<td>encryption_method</td>
<td>string</td>
<td>Must be used if version 3 of the SNMP protocol is configured. Configures encrypted</td>
</tr>
<tr>
<td>Elements of version</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>communication with the SNMP server. Possible values are:</td>
</tr>
<tr>
<td>encryption_method</td>
<td>string</td>
<td>- none: No encryption. The value of the encryption_password element must also be set to null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- aes: AES encryption. The encryption_password element must reference a valid password.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- des: DES encryption. The encryption_password element must reference a valid password.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>encryption_password string Must be used if version 3 of the SNMP protocol is configured. Set to null if the value of the encryption_method is set to none. References the password used for encrypting the communication with the SNMP server. You can create passwords at the /api/configuration/passwords/ endpoint. To modify or add a password, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key). The referenced password must be at least 8 characters long, and can contain letters (a-z, A-Z), numbers (0-9) the special characters (!&quot;#$%&amp;'()*+,;&lt;=&amp;@^`{</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td>Must be used if version 3 of the SNMP protocol is configured. The Engine ID. Must be a a hexadecimal number at least 10 digits long (for example, 0x0123456789ABCDEF).</td>
</tr>
</tbody>
</table>
### Elements of version

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>Contains the IP address of FQDN of the SNMP server.</td>
</tr>
<tr>
<td>selection</td>
<td>Defines the address type (IP or domain name). Possible values are:</td>
</tr>
<tr>
<td></td>
<td>- fqdn The SNMP server address is provided as a fully qualified domain name.</td>
</tr>
<tr>
<td></td>
<td>- ip The SNMP server address is provided as an IP address.</td>
</tr>
<tr>
<td>value</td>
<td>The address of the SNMP server.</td>
</tr>
<tr>
<td>username</td>
<td>Must be used if version 3 of the SNMP protocol is configured.</td>
</tr>
</tbody>
</table>

**Examples:**

Configure a server with the SNMP v2c protocol.

```json
{
    "enabled": true,
    "version": {
        "selection": "2c",
        "value": {
            "community": "public",
            "server": {
                "selection": "ip",
                "value": "<server-ip>"
            }
        }
    }
}
```

Configure a server with the SNMP v3 protocol, and MD5 authentication.

```json
{
    "enabled": true,
    "version": {
        "selection": "3",
        "value": {
```
Configure a server with the SNMP v3 protocol, SHA1 authentication, and AES-encrypted communication.

```json
{
    "enabled": true,
    "version": {
        "selection": "3",
        "value": {
            "auth_method": "sha",
            "auth_password": {
                "key": "0f5f646d-d6e7-4a4a-bc66-ead670faff3f",
                "meta": {
                    "href": "/api/configuration/passwords/0f5f646d-d6e7-4a4a-bc66-ead670faff3f"
                }
            }
        }
    },
    "encryption_method": "aes",
    "encryption_password": {
        "key": "6237d67a-b6b4-49e0-b0f6-6d68d0f08cc3",
        "meta": {
            "href": "/api/configuration/passwords/6237d67a-b6b4-49e0-b0f6-6d68d0f08cc3"
        }
    },
    "engine_id": "<0x0123456789ABCDEF>",
    "server": {
        "selection": "ip",
        "value": "<server-ip>"
    },
    "username": "<username>"
}
```
Modify SNMP trap settings

To modify the address and protocol settings for the SNMP server, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the SNMP trap endpoint.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/management/snmp/trap endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Local services: access for SNMP agents

External SNMP agents can query the basic status information of SPS. On this endpoint you can configure on which interfaces can the users access SPS, and optionally restrict the access to these interfaces, and configure authentication and encryption settings.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/local_services/snmp_agent
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the configuration options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/local_services/snmp_agent
```

**Response**

The following is a sample response received when listing the configuration options. For details of the meta object, see Message format on page 10.
{"body": {
"access_restriction": {
"enabled": false
},
"enabled": true,
"listen": [
{
"address": {
"key": "nic1.interfaces.ff754025754b3df1647001.addresses.1",
"meta": {
"href": "/api/configuration/network/nics/nic1#interfaces/ff754025754b3df1647001/addresses/1"
}
},
"port": 161
}
],
"system_contact": "mycontact",
"system_description": "mydescription",
"system_location": "mylocation",
"version_2c": {
"community": "mycommunity",
"enabled": true
},
"version_3": {
"enabled": true,
"users": [
{
"auth_method": "sha",
"auth_password": {
"key": "5476940c-ba38-4002-96d4-cb09d6921c68",
"meta": {
"href": "/api/configuration/passwords/5476940c-ba38-4002-96d4-cb09d6921c68"
}
},
"encryption_method": "aes",
"encryption_password": {
"key": "99782a91-63de-4a5c-82ff-b82273894dc7",
"meta": {
"href": "/api/configuration/passwords/99782a91-63de-4a5c-82ff-b82273894dc7"
}
},
"username": "myusername"}
"key": "snmp_agent",
"meta": {
  "first": "/api/configuration/local_services/admin_web",
  "href": "/api/configuration/local_services/snmp_agent",
  "last": "/api/configuration/local_services/user_web",
  "next": "/api/configuration/local_services/ssh",
  "parent": "/api/configuration/local_services",
  "previous": "/api/configuration/local_services/postgresql",
  "transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration options of the SNMP agent.</td>
</tr>
<tr>
<td>access_restriction</td>
<td>JSON object</td>
<td>Enables and configures limitations on the clients that can access the web interface, based on the IP address of the clients.</td>
</tr>
<tr>
<td>allowed_from</td>
<td>list</td>
<td>The list of IP networks from where the administrators are permitted to access this management interface. To specify the IP addresses or networks, use the IPv4-Address/prefix format, for example, 10.40.0.0/16.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set it to true to restrict access to the specified client addresses.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Enables the SNMP server. If this option is set to False, SPS ignores every other option of this endpoint.</td>
</tr>
<tr>
<td>listen</td>
<td>list</td>
<td>Selects the network interface, IP address, and port where the clients can access the web interface.</td>
</tr>
<tr>
<td>address</td>
<td>JSON object</td>
<td>A reference to a configured network interface and IP address where this local service accepts connections. For example, if querying the interface /api/configuration/network/nics/nic1#interfaces/ff754025754b3df1647001 hablar the following</td>
</tr>
</tbody>
</table>
Then the listening address of the local service is the following.

```
nic1.interfaces.ff7574025754b3df1647001.addresses.1
```

This is the format you have to use when configuring the
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td></td>
<td>address of the local service using REST:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;address&quot;: &quot;nic1.interfaces.ff7574025754b3df1647001.addresses.1&quot;</td>
</tr>
</tbody>
</table>

When querying a local services endpoint, the response will contain a reference to the IP address of the interface in the following format:

```
"address": {
  "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1",
  "meta": {
    "href": "/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001(addresses/1"
  }
},
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>integer</td>
<td>The port number where this local service accepts connections.</td>
</tr>
<tr>
<td>system_contact</td>
<td>string</td>
<td>Optional. For example, it can contain the contact information of the SPS administrator.</td>
</tr>
<tr>
<td>system_description</td>
<td>string</td>
<td>Optional. For example, it can contain information of the SPS host.</td>
</tr>
<tr>
<td>system_description</td>
<td>string</td>
<td>Optional. For example, it can contain the location of the SPS appliance.</td>
</tr>
<tr>
<td>version_2c</td>
<td>JSON object</td>
<td>Enables and configures SNMP queries using the SNMP v2c protocol. You can have both the SNMP v2c and v3 protocols enabled at the same time. For example:</td>
</tr>
</tbody>
</table>
|                  |            | "version_2c": {
|                  |            |   "community": "mycommunity",
|                  |            |   "enabled": true
|                  |            | },
<p>| community        | string     | Optional. Specifies the community to use.                                  |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Optional. Enables SNMP queries using the SNMP v2c protocol.</td>
</tr>
<tr>
<td>version_3</td>
<td>JSON object</td>
<td>Enables and configures SNMP queries using the SNMP v3 protocol. You can have both the SNMP v2c and v3 protocols enabled at the same time. You must configure an authentication method and a password, encryption is optional. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;version_3&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;users&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auth_method&quot;: &quot;sha&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auth_password&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;key&quot;: &quot;5476940c-ba38-4002-96d4-cb09d6921c68&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;meta&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;href&quot;: &quot;/api/configuration/passwords/5476940c-ba38-4002-96d4-cb09d6921c68&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;encryption_method&quot;: &quot;aes&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;encryption_password&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;key&quot;: &quot;99782a91-63de-4a5c-82ff-b82273894dc7&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;meta&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;href&quot;: &quot;/api/configuration/passwords/99782a91-63de-4a5c-82ff-b82273894dc7&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;username&quot;: &quot;myusername&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of version_3</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Optional. Enables SNMP queries using the SNMP v2c protocol.</td>
</tr>
<tr>
<td>users</td>
<td>JSON object</td>
<td>Contains the configuration parameters for the SNMP v3 protocol.</td>
</tr>
<tr>
<td>auth_method</td>
<td>string</td>
<td>Required parameter when using SNMP version 3.</td>
</tr>
<tr>
<td>Elements of version_3</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Configures encrypted communication with the SNMP server. Possible values are:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- md5: Use MD5 encryption. The auth_password element must reference a valid password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- sha1: Use SHA1 encryption. The auth_password element must reference a valid password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>auth_password</td>
<td>string</td>
<td>Required parameter when using SNMP version 3. References the password used for authenticating to the SNMP server. You can create passwords at the /api/configuration/passwords/ endpoint. To modify or add a password, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key). The referenced password must be at least 8 characters long, and can contain letters (a-z, A-Z), numbers (0-9) the special characters (!&quot;#$%&amp;'()*+,;&lt;=&amp;@^`{</td>
</tr>
<tr>
<td>encryption_method</td>
<td>string</td>
<td>Configures encrypted communication with the SNMP server. Possible values are:</td>
</tr>
<tr>
<td>- none: No encryption. The value of the encryption_password element must also be set to null.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- aes: AES encryption. The encryption_password element must reference a valid password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- des: DES encryption. The encryption_password element must reference a valid password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>encryption_password</td>
<td>string</td>
<td>Set to null if the value of the encryption_method is set to none. References the password used for encrypting the communication with the SNMP server. You can create passwords at the /api/configuration/passwords/ endpoint. To modify or add a password, use the value of the returned key as the value of the x509_identity element, and remove any child elements</td>
</tr>
</tbody>
</table>
Elements of version_3  Type  Description

(including the key).

The referenced password must be at least 8 characters long, and can contain letters (a-z, A-Z), numbers (0-9) the special characters (!"#$%&'()*+,-.;<=@<\]^`{}`_./:?-) and the space character.

username  string  The username for sending SNMP traps.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Alerting

Contains the endpoints for configuring alerting on SPS.

URL

GET https://<IP-address-of-SPS>/api/configuration/alerting

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_</td>
<td>Contains the</td>
<td>Required</td>
<td>The value of the session ID cookie received</td>
</tr>
</tbody>
</table>
### Cookie

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>authentication token of the user</td>
<td></td>
<td>from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

---

### Sample request

The following command lists alerting configuration endpoints.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/alerting
```

### Response

The following is a sample response received when listing alerting configuration endpoints.

For details of the meta object, see [Message format](#) on page 10.

```
{
  "items": [
    {
      "key": "system_alerts",
      "meta": {
        "href": "/api/configuration/alerting/system_alerts"
      }
    },
    {
      "key": "traffic_alerts",
      "meta": {
        "href": "/api/configuration/alerting/traffic_alerts"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa",
    "href": "/api/configuration/alerting",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/datetime",
  }
}
```
"parent": "/api/configuration",
"previous": "/api/configuration/aaa",
"transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>system_alerts</td>
<td>Configuration options for system-related alerts.</td>
</tr>
<tr>
<td>traffic_alerts</td>
<td>Configuration options for traffic-related alerts.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**System alerts**

Configuration options for sending system-related alerts.

E-mail alerts, when enabled, are sent to the e-mail address configured in the alerting_address element of the /api/configuration/management/email endpoint.

SNMP alerts, when enabled, are sent to the SNMP server configured at the /api/configuration/management/snmp/trap endpoint.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/alerting/system_alerts
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists configuration options for system-related alerts.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/alerting/system_alerts
```

Response

The following is a sample response received when listing configuration options for system-related alerts.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "xcbAlert": {
            "email": false,
            "snmp": false
        },
        "xcbArchiveFailed": {
            "email": false,
            "snmp": false
        },
        "xcbBackupFailed": {
            "email": false,
            "snmp": false
        },
        "xcbBruteForceAttempt": {
            "email": false,
```
"snmp": false
},
"xcbConfigChange": {
  "email": false,
  "snmp": false
},
"xcbDBError": {
  "email": false,
  "snmp": false
},
"xcbDiskFull": {
  "email": false,
  "snmp": false
},
"xcbError": {
  "email": false,
  "snmp": false
},
"xcbFirmwareTainted": {
  "email": false,
  "snmp": false
},
"xcbHWError": {
  "email": false,
  "snmp": false
},
"xcbHaNodeChanged": {
  "email": false,
  "snmp": false
},
"xcbLicenseAlmostExpired": {
  "email": false,
  "snmp": false
},
"xcbLimitReached": {
  "email": false,
  "snmp": false
},
"xcbLoadAvgHigh": {
  "email": false,
  "snmp": false
},
"xcbLogin": {
  "email": false,
  "snmp": false
},
"xcbLoginFailure": {
  "email": false,
"snmp": false,
"xcbLogout": {
  "email": false,
  "snmp": false
},
"xcbRaidStatus": {
  "email": false,
  "snmp": false
},
"xcbSwapFull": {
  "email": false,
  "snmp": false
},
"xcbTimeSyncLost": {
  "email": false,
  "snmp": false
},
"xcbTimestampError": {
  "email": false,
  "snmp": false
}
},
"key": "system_alerts",
"meta": {
  "first": "/api/configuration/alerting/system_alerts",
  "href": "/api/configuration/alerting/system_alerts",
  "last": "/api/configuration/alerting/traffic_alerts",
  "next": "/api/configuration/alerting/traffic_alerts",
  "parent": "/api/configuration/alerting",
  "previous": null,
  "transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration options for system-related alerts.</td>
</tr>
<tr>
<td>xcbAlert</td>
<td>Top level item</td>
<td>General alert.</td>
</tr>
<tr>
<td><strong>Element</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------</td>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbArchiveFailed</td>
<td>Top level item</td>
<td>Data archiving failure.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbBackupFailed</td>
<td>Top level item</td>
<td>Data and configuration backup failure.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbBruteForceAttempt</td>
<td>Top level item</td>
<td>Too many successive failed login attempts.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbConfigChange</td>
<td>Top level item</td>
<td>Configuration change.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbDBError</td>
<td>Top level item</td>
<td>Database error occured.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbDiskFull</td>
<td>Top level item</td>
<td>Disk utilization reached the percentage configured in the maximum_disk_utilization_ratio element of the api/configuration/management/monitoring endpoint.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>xcbError</td>
<td>Top level</td>
<td>General error.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbFirmwareTainted</td>
<td>Top level</td>
<td>The firmware is tainted.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbHWError</td>
<td>Top level</td>
<td>Hardware error.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbHaNodeChanged</td>
<td>Top level</td>
<td>HA node state changed.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbLicenseAlmostExpired</td>
<td>Top level</td>
<td>License expires soon.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbLimitReached</td>
<td>Top level</td>
<td>License limit reached.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbLoadAvgHigh</td>
<td>Top level</td>
<td>The average load exceeded any of the values configured in the maximum_load1, maximum_load5 or maximum_load15 elements of the api/configuration/management/monitoring</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>endpoint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>email boolean</td>
<td></td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp boolean</td>
<td></td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbLogin Top level</td>
<td></td>
<td>Successful login.</td>
</tr>
<tr>
<td>item</td>
<td></td>
<td></td>
</tr>
<tr>
<td>email boolean</td>
<td></td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp boolean</td>
<td></td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbLoginFailure Top</td>
<td></td>
<td>Failed login.</td>
</tr>
<tr>
<td>level item</td>
<td></td>
<td></td>
</tr>
<tr>
<td>email boolean</td>
<td></td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp boolean</td>
<td></td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbLogout Top level</td>
<td></td>
<td>Logout from the web configuration interface.</td>
</tr>
<tr>
<td>item</td>
<td></td>
<td></td>
</tr>
<tr>
<td>email boolean</td>
<td></td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp boolean</td>
<td></td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbRaidStatus Top</td>
<td></td>
<td>RAID status changed.</td>
</tr>
<tr>
<td>level item</td>
<td></td>
<td></td>
</tr>
<tr>
<td>email boolean</td>
<td></td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp boolean</td>
<td></td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbSwapFull Top</td>
<td></td>
<td>The utilization of the swap exceeded the value configured in the maximum</td>
</tr>
<tr>
<td>level item</td>
<td></td>
<td>swap_utilization_ratio element of the api/configuration/management/monitoring</td>
</tr>
<tr>
<td>endpoint</td>
<td></td>
<td>endpoint.</td>
</tr>
<tr>
<td>email boolean</td>
<td></td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp boolean</td>
<td></td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbTimeSyncLost Top</td>
<td></td>
<td>Time sync lost.</td>
</tr>
<tr>
<td>level item</td>
<td></td>
<td></td>
</tr>
<tr>
<td>email boolean</td>
<td></td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>xcbTimestampError</td>
<td>Top level item</td>
<td>Time stamping error.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
</tbody>
</table>

### Modify a system-related alert

To enable or disable an alert, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the endpoint.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/alerting/system_alerts` endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Traffic alerts

Configuration options for sending traffic-related alerts.

E-mail alerts, when enabled, are sent to the e-mail address configured in the alerting_address element of the /api/configuration/management/email endpoint.

SNMP alerts, when enabled, are sent to the SNMP server configured at the /api/configuration/management/snmp/trap endpoint.

URL

GET https://<IP-address-of-SPS>/api/configuration/alerting/traffic_alerts

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configuration options for traffic-related alerts.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/alerting/traffic_alerts
```

Response

The following is a sample response received when listing the configuration options for traffic-related alerts.

For details of the meta object, see Message format on page 10.
{ "body": {
  "scbAuthFailure": {
    "email": false,
    "snmp": false
  },
  "scbAuthSuccess": {
    "email": false,
    "snmp": false
  },
  "scbChannelDenied": {
    "email": false,
    "snmp": false
  },
  "scbConnectionDenied": {
    "email": false,
    "snmp": false
  },
  "scbConnectionFailed": {
    "email": false,
    "snmp": false
  },
  "scbConnectionTimedout": {
    "email": false,
    "snmp": false
  },
  "scbCredStoreClosed": {
    "email": false,
    "snmp": false
  },
  "scbCredStoreDecryptError": {
    "email": false,
    "snmp": false
  },
  "scbCredStoreUnlockFailure": {
    "email": false,
    "snmp": false
  },
  "scbGWAuthFailure": {
    "email": false,
    "snmp": false
  },
  "scbGWAuthSuccess": {
    "email": false,
    "snmp": false
  },
  "scbProtocolViolation": {
    "email": false,
    "snmp": false
  }
}
"snmp": false,
"scbRealTimeAlert": {
  "email": false,
  "snmp": false
},
"scbSshHostKeyLearned": {
  "email": false,
  "snmp": false
},
"scbSshHostKeyMismatch": {
  "email": false,
  "snmp": false
},
"scbUserMappingFailure": {
  "email": false,
  "snmp": false
}
},
"key": "traffic_alerts",
"meta": {
  "first": "/api/configuration/alerting/system_alerts",
  "href": "/api/configuration/alerting/traffic_alerts",
  "last": "/api/configuration/alerting/traffic_alerts",
  "next": null,
  "parent": "/api/configuration/alerting",
  "previous": "/api/configuration/alerting/system_alerts",
  "transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration options for traffic-related alerts.</td>
</tr>
<tr>
<td>scbAuthFailure</td>
<td>Top level item</td>
<td>User authentication failed.</td>
</tr>
<tr>
<td></td>
<td>email</td>
<td>boolean</td>
</tr>
<tr>
<td></td>
<td>snmp</td>
<td>boolean</td>
</tr>
<tr>
<td>scbAuthSuccess</td>
<td>Top level</td>
<td>Successful user authen-</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>scbChannelDenied</td>
<td>Top level item</td>
<td>Channel opening denied.</td>
</tr>
<tr>
<td><strong>email</strong></td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td><strong>snmp</strong></td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbConnectionDenied</td>
<td>Top level item</td>
<td>Connection denied.</td>
</tr>
<tr>
<td><strong>email</strong></td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td><strong>snmp</strong></td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbConnectionFailed</td>
<td>Top level item</td>
<td>Connection to the server failed.</td>
</tr>
<tr>
<td><strong>email</strong></td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td><strong>snmp</strong></td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbConnectionTimedout</td>
<td>Top level item</td>
<td>Connection timed out.</td>
</tr>
<tr>
<td><strong>email</strong></td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td><strong>snmp</strong></td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbCredStoreClosed</td>
<td>Top level item</td>
<td>The requested credential store is closed.</td>
</tr>
<tr>
<td><strong>email</strong></td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td><strong>snmp</strong></td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>scbCredStoreDecryptError</td>
<td>Top level item</td>
<td>Failure to decrypt a credential.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbCredStoreUnlockFailure</td>
<td>Top level item</td>
<td>Failure to unlock the credential store.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbGWAuthFailure</td>
<td>Top level item</td>
<td>The user failed to authenticate on the gateway.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbGWAuthSuccess</td>
<td>Top level item</td>
<td>Successful authentication on the gateway.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbProtocolViolation</td>
<td>Top level item</td>
<td>Protocol violation.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbRealTimeAlert</td>
<td>Top level item</td>
<td>Real-time audit event detected.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------</td>
<td>----------------------------------------------------------</td>
</tr>
<tr>
<td>scbSshHostKeyLearned</td>
<td>Top level</td>
<td>New SSH host key learned.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbSshHostKeyMismatch</td>
<td>Top level</td>
<td>SSH host key mismatch.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
<tr>
<td>scbUserMappingFailure</td>
<td>Top level</td>
<td>User mapping failed on the gateway.</td>
</tr>
<tr>
<td>email</td>
<td>boolean</td>
<td>Set to true to enable e-mail alerts.</td>
</tr>
<tr>
<td>snmp</td>
<td>boolean</td>
<td>Set to true to enable SNMP alerts.</td>
</tr>
</tbody>
</table>

Modify a traffic-related alert

To enable or disable an alert, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the endpoint.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/alerting/traffic_alerts endpoint. You can find a detailed description of the available parameters listed in **Element**.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Trust stores**

Trust stores serve as local certificate storages where users can store the certificate chains of trusted Certificate Authorities (CAs). These certificates are then used to ensure secure communication between external parties and the SPS.

There are two types of trust stores: built-in and custom.

The built-in trust store has well known root CAs (such as Google, Microsoft, Verisign, etc.), and it is not modifiable.

Before establishing secure communication (TLS), SPS verifies the certificate of the other party using the assigned trust store. Only certificates signed by any of the CAs in the trust store are accepted.

**NOTE:** CRL URLs must be listed explicitly in the appropriate field, as those CRL URLs that are embedded in the extensions of the certificates, will be ignored.

**URL**

```bash
GET https://<IP-address-of-SPS>/api/configuration/trust_stores
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860.</td>
</tr>
<tr>
<td>Cookie name</td>
<td>Description</td>
<td>Required</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Operations with the trust_stores endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a trust store</td>
<td>POST</td>
<td>/api/configuration/trust_stores</td>
<td>The name of the trust store must be unique.</td>
</tr>
<tr>
<td>List trust stores</td>
<td>GET</td>
<td>/api/configuration/trust_stores</td>
<td>Users who were not given read access to the trust_stores endpoint explicitly, are still able to retrieve information from it, if they have access to other /configuration related endpoints, which reference trust stores. Examples of trust store referrer ACL (read access):</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• /pages/starlingjoin</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• /config/xcb/aaa/settings</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• /config/scb/pol_ldaps</td>
</tr>
<tr>
<td>Query a trust store</td>
<td>GET</td>
<td>/api/configuration/trust_stores/&lt;id of the trust store&gt;</td>
<td></td>
</tr>
<tr>
<td>Query the built-in trust store</td>
<td>GET</td>
<td>/api/configuration/trust_stores/-7001</td>
<td></td>
</tr>
<tr>
<td>Update a trust store</td>
<td>PUT</td>
<td>/api/configuration/trust_stores/&lt;id of the trust store&gt;</td>
<td>Users who were not given access to the trust_stores endpoint explicitly, but are still able to retrieve information from it because they have access to</td>
</tr>
<tr>
<td>Operation</td>
<td>HTTP method</td>
<td>URL</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------</td>
<td>-------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Delete a trust store</td>
<td>DELETE</td>
<td>/api/configuration/trust_stores/&lt;id of the trust store&gt;</td>
<td>configuration endpoints which reference trust stores, are unable to modify trust stores. With the exception of /config/xcb/management, where the same access level is granted to the trust stores for the user as they have for /config/xcb/management.</td>
</tr>
</tbody>
</table>

**Sample request**

The following command lists the trust stores:

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/trust_stores
```

**Response**

The following is a sample response received when listing trust stores.

For details of the `meta` object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "-7001",
            "meta": {
                "href": "/api/configuration/trust_stores/-7001"
            },
            "body": {
                "name": "Built-in",
                "revocation_check": "none",
                "trust_store_type": "built-in"
            }
        },
        {
            "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXXX",
            "meta": {
                "href": "/api/configuration/trust_stores/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXXX"
            }
        }
    ]
}
```
Elements of the response message body:
<table>
<thead>
<tr>
<th><strong>Elements of items</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
<th><strong>Notes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>object array</td>
<td>List of JSON objects available from the current endpoint.</td>
<td>Each trust store has a unique key. The built-in trust store's ID is &quot;-7001&quot;.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>The ID of the trust store.</td>
<td></td>
</tr>
<tr>
<td>meta</td>
<td>string (url)</td>
<td>The href field contains the URL of the trust store.</td>
<td></td>
</tr>
</tbody>
</table>

**body**

<table>
<thead>
<tr>
<th><strong>Elements of body</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
<th><strong>Notes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>object</td>
<td>Top level element.</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the trust store.</td>
<td>The name field is set by the user and it must be unique. For example: &quot;name&quot;: &quot;My_Custom_Trust_Store&quot;. The built-in trust store's name is &quot;Built-in&quot;.</td>
</tr>
</tbody>
</table>

**authorities**

| **crl_urls** | string array | The crl_urls field contains the list of CRL web addresses (HTTP or HTTPS URLs) used for revocation check. | If a trust store that uses certificate revocation lists (CRLs) does not work properly, it might be due to invalid or inaccessible CRL URLs. Troubleshooting can involve checking whether all URLs of the CA CRL URL list are valid, and can be accessed from the SPS via the Basic Settings / Troubleshooting / Connect to TCP port function in the Web UI. |
| revocation_check | enum | The type of the revocation check. | Possible values: "full", "leaf", "none". "full" - The crl_urls field must contain CRL URLs for all of the |
### Elements of body

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CAs that are part of the chain of a given certificate which is being verified.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;leaf&quot; - The crl_urls field must contain at least the CRL URL of the CA which signed the certificate which is being validated.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;none&quot; - The crl_urls field must be empty.</td>
<td></td>
</tr>
<tr>
<td>trust_store_type</td>
<td>The type of the trust store.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Possible values: &quot;built-in&quot;, &quot;custom&quot;.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The built-in trust store comes with the operation system. This type of trust store is read-only. There is no CRL check involved, and it cannot be removed.</td>
<td></td>
</tr>
</tbody>
</table>

### Elements of authorities

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorities</td>
<td>List of Certificate Authorities.</td>
<td></td>
</tr>
<tr>
<td>fingerprint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>issuer</td>
<td>The name of the entity that signed the certificate.</td>
<td></td>
</tr>
<tr>
<td>pem</td>
<td>The certificate in PEM format.</td>
<td></td>
</tr>
<tr>
<td>subject</td>
<td>The subject of the certificate.</td>
<td></td>
</tr>
</tbody>
</table>

### Elements of fingerprint

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>fingerprint</td>
<td>A two-piece byte sequence consisting of a hash algorithm and a message digest.</td>
<td></td>
</tr>
<tr>
<td>digest</td>
<td>The string of digits produced by the hash algorithm.</td>
<td></td>
</tr>
<tr>
<td>hash_algorithm</td>
<td>The name of the hash algorithm.</td>
<td></td>
</tr>
</tbody>
</table>
**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes on page 37](#).

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
</table>
| 400  | SyntacticError | A value to be set is not accepted syntactically. The details section contains the path that was found to be invalid. Possible syntactic error messages related to trust store:  
  - The user is not allowed to create a built-in trust store or edit or delete the existing one.  
  - When revocation_check is set to "none", the crl_urls field must be empty. The user cannot add any element to crl_urls.  
  - When revocation_check is set to "full" or "leaf", the crl_urls cannot be empty. |
<p>| 400  | SemanticError | The configuration contains semantic errors, inconsistencies or other problems that would put the system into an unreliable state if the configuration had been applied. The details section contains the errors that were found in the configuration. |</p>
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Possible semantic error messages related to trust store:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The name of the trust stores must be unique.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The authorities of a trust store must be unique.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The CRL URLs of a trust store must be unique.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>resource requires authorization to access it. The details section contains the path that</td>
</tr>
<tr>
<td></td>
<td></td>
<td>was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access</td>
</tr>
<tr>
<td></td>
<td></td>
<td>it. The details section contains the path that was attempted to be accessed, but could not</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be retrieved.</td>
</tr>
</tbody>
</table>

## Enabling One Identity Safeguard Remote Access without Starling Join

Enable the One Identity Safeguard Remote Access (SRA) feature of One Identity Starling without **Starling Join** information.

**NOTE:** You cannot configure Starling Join through the resource configuration endpoint (/api/configuration/starling), only through the dedicated /starling/join endpoint.
## URL

```
GET https://<IP-address-of-SPS>/api/configuration/starling
```

## Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

## Operations

Operations with the `/starling` endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query One Identity Starling Join without a join to One Identity Starling</td>
<td>GET</td>
<td>/api/configuration/starling</td>
<td>The value of the <code>join_info</code> field is null.</td>
</tr>
<tr>
<td>Query One Identity Starling Join after join</td>
<td>GET</td>
<td>/api/configuration/starling</td>
<td>The values of the <code>join_info</code> field are <code>environment</code>, <code>product_instance</code>, and <code>product_tims</code>.</td>
</tr>
<tr>
<td>Enable SRA</td>
<td>PUT</td>
<td>/api/configuration/starling</td>
<td>SRA can be enabled only if the node is joined to One Identity Starling. Use the <code>starling/join</code> endpoint to join to One Identity</td>
</tr>
</tbody>
</table>
## Operation HTTP method URL Notes

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable SRA</td>
<td>PUT</td>
<td>/api/configuration/starling</td>
<td>To disable SRA to One Identity Starling, the enabled field must be set to false.</td>
</tr>
</tbody>
</table>

### Enable SRA

To enable SRA with the use of the /starling endpoint, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. Set the **enabled** parameter of **remote_access** to true.

### Sample request

The following command enables SRA to join to One Identity Starling.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/starling
```

```json
{
   "join_info": {
      "environment": "prod",
      "product_instance": "starling-joiner-11111111-1111-1111-1111-111111111111",
      "product_tims": "123-456-789"
   },
   "remote_access": {
      "enabled": true
   }
}
```

Elements of the request message body include:

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>join_info</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>join_info.environment</td>
<td>string</td>
<td>The environment</td>
<td>Possible values: prod.</td>
</tr>
<tr>
<td>Elements</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------</td>
<td>------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>environment</td>
<td></td>
<td>of the product.</td>
<td></td>
</tr>
<tr>
<td>join_info.product_instance</td>
<td>string</td>
<td>The instance of the product.</td>
<td></td>
</tr>
<tr>
<td>join_info.product_tims</td>
<td>string</td>
<td>The TIMS license of the product.</td>
<td>To disable SRA to One Identity Starling, the enabled field must be set to false.</td>
</tr>
<tr>
<td>remote_access</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>remote_access.enabled</td>
<td>boolean</td>
<td>Enables or disables SRA to One Identity Starling.</td>
<td>Possible values: true, false</td>
</tr>
<tr>
<td>key</td>
<td></td>
<td></td>
<td>Possible values: starling</td>
</tr>
</tbody>
</table>

3. **Commit your changes.**

For more information, see *Commit a transaction* on page 31.

**Response**

The response is a regular meta object.

For details of the meta object, see *Message format* on page 10.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see *Application level error codes* on page 37.

Endpoint-specific HTTP response codes for this request:

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status / Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>403</td>
<td>RemoteAccessDependsOnStarlingJoinError</td>
<td>SRA to One Identity Starling can only be enabled, if the node is joined to One Identity Starling. Use the /starling/join endpoint to join to One Identity Starling.</td>
</tr>
<tr>
<td>403</td>
<td>StarlingJoinInfoIsReadOnlyError</td>
<td>You cannot join or unjoin from</td>
</tr>
</tbody>
</table>
One Identity Starling at the resource configuration endpoint (/api/configuration/starling), as the One Identity Starling join_info field is read-only. Use the /starling/join endpoint to join or unjoin from One Identity Starling.

Standard HTTP response codes for this request:

- 400 SyntacticError
- 400 SemanticError
- 401 Unauthenticated

### Configuring Starling Join

Configure the **Starling Join** feature on One Identity Safeguard for Privileged Sessions (SPS) with the use of a RESTful API, to be able to join SPS to One Identity Starling, or unjoin SPS from it.

**NOTE:** You cannot configure Starling Join through the resource configuration endpoint (/api/configuration/starling), only through the dedicated /starling/join endpoint.

### URL

GET https://<IP-address-of-SPS>/api/starling/join

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see</td>
</tr>
</tbody>
</table>
### Cookie name Description Required Values

Authenticate to the SPS REST API on page 19.

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Operations

Operations with the /starling/join endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Querying Starling Join endpoint from a browser without TIMS</td>
<td>GET</td>
<td>/api/starling/join</td>
<td>Headers to be used: Accept: text/html. <strong>NOTE:</strong> If the specified Accept header is not sent with the request, then the Starling Join process is not started, only the status information is returned.</td>
</tr>
<tr>
<td>Querying Starling Join endpoint when joined from a browser without TIMS</td>
<td>GET</td>
<td>/api/starling/join</td>
<td></td>
</tr>
<tr>
<td>Querying Starling Join info without TIMS</td>
<td>GET</td>
<td>/api/starling/join</td>
<td></td>
</tr>
<tr>
<td>Querying Starling Join info when joined without</td>
<td>GET</td>
<td>/api/starling/join</td>
<td></td>
</tr>
<tr>
<td>Operation</td>
<td>HTTP method</td>
<td>URL</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------</td>
<td>------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Querying Starling Join endpoint when joined from a browser with TIMS</td>
<td>GET</td>
<td>/api/starling/join?product_tims=&lt;TIMS value&gt;</td>
<td>Example of TIMS value: product_tims=222-333-444.</td>
</tr>
<tr>
<td>Querying Starling Join info with TIMS</td>
<td>POST</td>
<td>/api/starling/join</td>
<td>NOTE: Unjoining One Identity Starling in a regular way is not possible while One Identity Safeguard Remote Access (SRA) is enabled. To unjoin One Identity Starling, first you must disable SRA.</td>
</tr>
<tr>
<td>Querying Starling Join info when joined with TIMS</td>
<td>POST</td>
<td>/api/starling/join</td>
<td>NOTE: Unjoining One Identity Starling in a regular way is not possible while One Identity Safeguard Remote Access (SRA) is enabled. To unjoin One Identity Starling, first you must disable SRA.</td>
</tr>
<tr>
<td>Joining One Identity Starling</td>
<td>POST</td>
<td>/api/starling/join</td>
<td>NOTE: Unjoining One Identity Starling in a regular way is not possible while One Identity Safeguard Remote Access (SRA) is enabled. To unjoin One Identity Starling, first you must disable SRA.</td>
</tr>
<tr>
<td>Unjoining One Identity Starling</td>
<td>DELETE</td>
<td>/api/starling/join</td>
<td>NOTE: The Force</td>
</tr>
<tr>
<td>Force unjoining</td>
<td>DELETE</td>
<td>/api/starling/join?force=true</td>
<td>NOTE: The Force</td>
</tr>
<tr>
<td>Operation</td>
<td>HTTP method</td>
<td>URL</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>One Identity Starling</td>
<td></td>
<td></td>
<td>Unjoin functionality works even when SRA is enabled.</td>
</tr>
</tbody>
</table>

### Sample request

The following command queries the /starling/join endpoint when joined from a browser without TIMS.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/starling/join
```

### Response

The following is a sample response.

For details of the meta object, see Message format on page 10.

```
{
  "body": {
    "joined": true,
    "product_instance": "starling-joiner-11111111-1111-1111-1111-111111111111",
    "product_name": "Safeguard",
    "product_tims": "123-456-789"
  }
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>joined</td>
<td>boolean</td>
<td>Displays whether or not the user is joined to One Identity Starling or unjoined from it.</td>
<td></td>
</tr>
<tr>
<td>product_instance</td>
<td>string</td>
<td>The instance of the product.</td>
<td></td>
</tr>
<tr>
<td>product_name</td>
<td>string</td>
<td>The name of the product.</td>
<td></td>
</tr>
<tr>
<td>product_tims</td>
<td>string</td>
<td>The TIMS license of the product.</td>
<td></td>
</tr>
</tbody>
</table>
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

Endpoint-specific HTTP response codes for this request:

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status / Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>MissingCredentialStringError</td>
<td>The mandatory credential_string parameter is not specified for One Identity Starling Join.</td>
</tr>
<tr>
<td>400</td>
<td>MissingProductInstanceError</td>
<td>The mandatory product_instance parameter is not specified for One Identity Starling Join.</td>
</tr>
<tr>
<td>403</td>
<td>OpenTransactionError</td>
<td>The attempt to join to One Identity Starling was unsuccessful, as the transaction was still open. To join to One Identity Starling, you must first close the previous transaction.</td>
</tr>
<tr>
<td>403</td>
<td>ForbiddenActionError</td>
<td>Forbidden action. To unjoin from One Identity Starling, use the /starling/join endpoint.</td>
</tr>
<tr>
<td>403</td>
<td>StarlingJoinIsInUseByRemoteAccessError</td>
<td>Unjoining One Identity Starling is not allowed while SRA is in use. Disable SRA in the configuration before unjoining from One Identity Starling.</td>
</tr>
</tbody>
</table>

Standard HTTP response codes for this request:

- 400 InvalidRequestBody
- 400 SyntacticError
- 400 SemanticError
- 401 Unauthenticated
- 403 Unauthorized
Retrieving the status of services related to Starling Join/Unjoin

Use the /status endpoint to retrieve information about the availability of the services needed for Starling Join, or Starling Unjoin.

URL

GET https://<IP-address-of-SPS>/api/starling/join/status

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command returns information about the availability of the services needed for Starling Join, or Starling Unjoin.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/starling/join/status
```

Response

The following is a sample response.

For details of the meta object, see Message format on page 10.
Elements of the response message body include:

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>internet_connection</td>
<td>object</td>
<td>Indicates whether or not the SPS appliance can connect to the status endpoint (oneidentitycloud.statuspage.io).</td>
<td></td>
</tr>
<tr>
<td>internet_connection.error</td>
<td>string</td>
<td>The description of the error. If no error occurred, the value is N/A.</td>
<td></td>
</tr>
<tr>
<td>Elements</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>internet_connection.status</td>
<td>boolean</td>
<td>The value is true, if the SPS appliance could connect to the status page.</td>
<td>The value is false, if there is no Internet connection.</td>
</tr>
<tr>
<td>proxy</td>
<td>boolean</td>
<td>Indicates whether or not a proxy server is configured.</td>
<td></td>
</tr>
<tr>
<td>starling_status</td>
<td>enum</td>
<td>Enumeration of the different One Identity Starling-related services and their current status.</td>
<td>NOTE: starling_status lists all available One Identity Starling-related services, however, to Starling Join/Unjoin to work, only the status of the One Identity Starling service is relevant.</td>
</tr>
<tr>
<td>verdict</td>
<td>boolean</td>
<td>The value is true if SPS can connect to the status page, a proxy is enabled, and the relevant service(s) are operational.</td>
<td></td>
</tr>
</tbody>
</table>

**HTTP response codes**

Standard HTTP response codes for this request:

- 400 InvalidRequestBody
- 400 SyntacticError
- 400 SemanticError
- 401 Unauthenticated
- 403 Unauthorized

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.
User management and access control

User management and access control

The AAA endpoint contains the configuration endpoints for the authentication, authorization, and account (AAA) settings of the users who access SPS.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/aaa/
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the AAA configuration endpoints.
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/

Response

The following is a sample response received when listing AAA configuration endpoints.
For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "acls",
      "meta": {
        "href": "/api/configuration/aaa/acls"
      }
    },
    {
      "key": "ldap_servers",
      "meta": {
        "href": "/api/configuration/aaa/ldap_servers"
      }
    },
    {
      "key": "local_database",
      "meta": {
        "href": "/api/configuration/aaa/local_database"
      }
    },
    {
      "key": "login_methods",
      "meta": {
        "href": "/api/configuration/aaa/login_methods"
      }
    },
    {
      "key": "settings",
      "meta": {
        "href": "/api/configuration/aaa/settings"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa",
    "href": "/api/configuration/aaa",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/alerting",
  }
}
```
Element | Description
--- | ---
acls | Access control settings for usergroups.
ldap_servers | LDAP server configuration for authentication, authorization, and accounting.
local_database | Local users and usergroups.
login_methods | Multiple login method configuration for SPS.
settings | Authentication and user database settings.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Login settings

Use login settings to control the web login access of administrators and users to SPS. With the /aaa/settings endpoint you can configure the following three security enhancing measures:
- Protecting against brute-force attacks
- Authentication banner
- Web interface timeout

For more information, see the corresponding sections in One Identity Safeguard for Privileged Sessions Administration Guide.

**URL**

```plaintext
POST https://<IP-address-of-SPS>/api/configuration/aaa/settings
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Operations**

Operations with the /aaa/settings endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating login settings</td>
<td>GET</td>
<td>/api/configuration/aaa/settings</td>
<td></td>
</tr>
<tr>
<td>Retrieving login settings</td>
<td>POST</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Sample request**

The following command lists the login settings.

```plaintext
curl -X GET -b "${COOKIE_PATH}" https://<IP-address-of-SPS>/api/configuration/aaa/settings
```
Response

The following is a sample response received when listing login settings.
For details of the meta object, see Message format on page 10.

```json
{
    "key": "settings",
    "body": {
        "authentication_banner": "",
        "bruteforce_protection": {
            "attempt_limit": 20,
            "lockout_minutes": 10
        },
        "webinterface_timeout": 10
    }
}
```

Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication_banner</td>
<td>string</td>
<td>Displays a banner with a configurable text on the web and console login screen of SPS. Users will see the banner every time they try to log in to SPS. The login screen displays the banner text as plain text, with whitespaces preserved.</td>
<td>If you specify an empty string, then no authentication banner will show.</td>
</tr>
<tr>
<td>bruteforce_protection</td>
<td>JSON object</td>
<td>Protects the web login addresses of administrators and users against brute-force attacks. After the users reach the configured number of unsuccessful login attempts, SPS denies all following attempts for the configured time.</td>
<td></td>
</tr>
</tbody>
</table>
| bruteforce_protection.attempt_limit | number        | The number of unsuccessful login attempts before the user name or the IP address is locked out. If the number of subsequent unsuccessful login attempts exceeds this limit, the IP address or the user name will be blocked for a period, which is specified in bruteforce_protection.lockout_minutes. | Value range: 1-50 attempts  
Default value: 20 attempts |
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>bruteforce_protection.lockout_</td>
<td>number</td>
<td>The period of time for which the user or the IP address is locked out from using the SPS appliance. It is measured in minutes.</td>
<td>Value range: 1-720 minutes Default value: 10 minutes</td>
</tr>
<tr>
<td>minutes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>webinterface_timeout</td>
<td>number</td>
<td>The period of inactivity after which SPS terminates the web session of a user. It is measured in minutes.</td>
<td>Value range: 5-720 minutes Default value: 10 minutes</td>
</tr>
</tbody>
</table>

**HTTP response codes**

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

**Privileges of usergroups**

This endpoint lists the usergroups configured on SPS, and the privileges (ACLs) of each group.

Note that currently you cannot edit the privileges (ACLs) of the groups using the REST API. If you change the privileges of a usergroup on the SPS web interface, the changes will apply to the users when they authenticate again on SPS, the privileges of active sessions are not affected.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/aaa/acls

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. NOTE: This session ID refers to the connec-</td>
</tr>
<tr>
<td>Cookie name</td>
<td>Description</td>
<td>Required</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
</tr>
</tbody>
</table>

The request sent to the SPS REST API mentioned in the previous section between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the local users.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/acls
```

Response

The following is a sample response received when querying the endpoint. For details of the meta object, see Message format on page 10.

```
{
    "body": [
        {
            "group": "basic-view",
            "objects": [
                "/special/basic"
            ],
            "permission": "read"
        },
        {
            "group": "basic-write",
            "objects": [
                "/special/basic"
            ],
            "permission": "write"
        },
        {
            "group": "auth-view",
            "objects": [
                "/special/auth"
            ],
            "permission": "read"
        },
        {
            "group": "auth-write",
            "objects": [
                "/special/auth"
            ],
            "permission": "write"
        }
    ]
}
```


```json
{
  "group": "search",
  "objects": [
    "/special/searchmenu"
  ],
  "permission": "read"
},
{
  "group": "changelog",
  "objects": [
    "/special/changelog"
  ],
  "permission": "read"
},
{
  "group": "policies-view",
  "objects": [
    "/special/pol"
  ],
  "permission": "read"
},
{
  "group": "policies-write",
  "objects": [
    "/special/pol"
  ],
  "permission": "write"
},
{
  "group": "ssh-view",
  "objects": [
    "/special/ssh"
  ],
  "permission": "read"
},
{
  "group": "ssh-write",
  "objects": [
    "/special/ssh"
  ],
  "permission": "write"
},
{
  "group": "rdp-view",
  "objects": [
    "/special/rdp"
  ],
},
```
"permission": "read"
},
{
  "group": "rdp-write",
  "objects": [
    "/special/rdp"
  ],
  "permission": "write"
},
{
  "group": "telnet-view",
  "objects": [
    "/special/telnet"
  ],
  "permission": "read"
},
{
  "group": "telnet-write",
  "objects": [
    "/special/telnet"
  ],
  "permission": "write"
},
{
  "group": "vnc-view",
  "objects": [
    "/special/vnc"
  ],
  "permission": "read"
},
{
  "group": "vnc-write",
  "objects": [
    "/special/vnc"
  ],
  "permission": "write"
},
{
  "group": "indexing",
  "objects": [
    "/special/search/search",
    "/special/bap"
  ],
  "permission": "write"
},
{
  "group": "ica-view",
  "objects": [
"/special/ica",
  "permission": "read"
}]

"key": "acls",
"meta": {
  "first": "/api/configuration/aaa/acls",
  "href": "/api/configuration/aaa/acls",
  "last": "/api/configuration/aaa/settings",
  "next": "/api/configuration/aaa/local_database",
  "parent": "/api/configuration/aaa",
  "previous": null,
}
"transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>Top level element (JSON object)</td>
<td>Contains the properties of the user.</td>
</tr>
<tr>
<td>group</td>
<td>string</td>
<td>The name of the usergroup.</td>
</tr>
<tr>
<td>objects</td>
<td>list</td>
<td>The list of privileges that the group has access to.</td>
</tr>
<tr>
<td>permission</td>
<td>read</td>
<td>write</td>
</tr>
</tbody>
</table>

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Audit data access rules

This endpoint enables you to restrict the search and access privileges of usergroups to audit data.

**URL**

GET https://<IP-address-of-SPS>/api/acl/audit_data
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Sample request

The following command lists the available audit data access rules.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/acl/audit_data
```

Response

The following is a sample response received when querying the endpoint. For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "autogenerated-10211162955b9621d4eb244",
            "meta": {
                "href": "/api/acl/audit_data/autogenerated-10211162955b9621d4eb244"
            }
        }
    ],
    "meta": {
        "href": "/api/acl/audit_data",
        "parent": "/api/acl",
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
    }
}
```
### Query a specific audit data access rule

To find out the contents of a particular audit data access rule, complete the following steps:

**NOTE:** If you have an SPS user who has **Search > Search in all connections** privileges in **Users & Access Control > Appliance Access**, the autogenerated-all-data-access-id rule is automatically generated. Therefore, you can almost always query this audit data access rule.

1. **Query the** https://<IP-address-of-SPS>/api/acl/audit_data/<key-of-rule-to-be-queried> **endpoint.**

   ```
curl --cookie cookies https://<IP-address-of-SPS>/api/acl/audit_data/<key-of-rule-to-be-queried>
   ``

   The following is a sample response received.

   For details of the `meta` object, see [Message format on page 10](#).

   ```json
   {
     "body": {
       "name": "my_ssh_rule",
       "query": "psm.connection_policy:my_ssh_connection_policy",
       "groups": [
         "ssh-view",
         "ssh-write"
       ]
     },
     "key": "autogenerated-10211162955b9621d4eb244",
     "meta": {
       "href": "/api/acl/audit_data/autogenerated-10211162955b9621d4eb244",
   ```

   

---

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Top-level element (list of JSON objects)</td>
<td>List of endpoints (objects) available from the current endpoint.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>The ID of the endpoint.</td>
</tr>
<tr>
<td>meta</td>
<td>Top-level item (JSON object)</td>
<td>Contains the path to the endpoint.</td>
</tr>
<tr>
<td>href</td>
<td>string (relative path)</td>
<td>The path of the resource that returned the response.</td>
</tr>
</tbody>
</table>
"parent": "/api/acl/audit_data",
"remaining_seconds": 600,
"transaction": "/api/transaction"
}
}

### Elements

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>Top-level element (JSON object)</td>
<td>Contains the JSON object of the rule.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The human-readable name of the audit data access rule that you specified when you created the rule.</td>
</tr>
<tr>
<td>query</td>
<td>string</td>
<td>The query that members of the usergroup(s) are allowed to perform.</td>
</tr>
<tr>
<td>groups</td>
<td>list</td>
<td>The usergroup(s) whose access to audit data you want to restrict.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>SemanticError</td>
<td>The configuration contains semantic errors, inconsistencies or other problems that would put the system into an unreliable state if the configuration had been applied. The details section contains the errors that were found in the configuration.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Active sessions

The api/active-sessions endpoint has only one parameter and it only serves the DELETE request that closes the specified session.

URL

DELETE https://<IP-address-of-SPS>/api/active-sessions?id=<session_id>

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the Access Control Lists (ACLs):

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/acls
```

The user (in this example, user1) has to be a member of a group that has read and write/perform privileges for Active Sessions (/special/active_sessions). After authentication, user1 can close the active session determined by the session ID.

```bash
curl -k --user user1 --cookie-jar /tmp/cookie https://192.168.122.194/api/authentication

curl -k --cookie /tmp/cookie https://192.168.122.194/api/active-sessions?id=svc/rpokH8fD9kx6CaxNLznKx2/test:12 -X DELETE
```
Closing active sessions in a cluster environment

In a cluster environment, after authentication, user1 can close active sessions recorded on Search Minion nodes through the Search Master node's IP address.

```
curl -k --cookie /tmp/cookie https://<IP-address-of-Search-Master-SPS>/api/active-sessions?id=<session_id> -X DELETE
```

Active sessions recorded on the Search Local node can be closed only from the node itself.

```
curl -k --cookie /tmp/cookie https://<IP-address-of-Search-Local-SPS>/api/active-sessions?id=<session_id> -X DELETE
```

Active sessions recorded on the Search Minion node can be closed from the node itself, as well.

```
curl -k --cookie /tmp/cookie https://<IP-address-of-Search-Minion-SPS>/api/active-sessions?id=<session_id> -X DELETE
```

**NOTE:** The following scenarios are not supported:

- Closing an active session recorded on Search Local node from the Search Master node.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>SessionIdMissing</td>
<td>No session ID is given in the &quot;id&quot; query parameter.</td>
</tr>
<tr>
<td>404</td>
<td>SessionCouldNotBeFound</td>
<td>No session could be found for the given session ID. Select an ongoing session at the Active Connections page on the Web UI and give its session ID as &quot;id&quot; query parameter.</td>
</tr>
<tr>
<td>500</td>
<td>SessionTerminationFailed</td>
<td>The session could not be terminated due to internal errors.</td>
</tr>
<tr>
<td>500</td>
<td>RemoteNodeInfoMissing</td>
<td>The cluster node where the session is being recorded is missing from your primary node's configuration. For assistance, contact our Support Team.</td>
</tr>
<tr>
<td>503</td>
<td>SessionTerminationServiceUnavailable</td>
<td>Session termination service is unavailable.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------</td>
<td>------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>504</td>
<td>MinionUnavailable</td>
<td>The minion node that is recording the session is unavailable. To get more information about the missing node, navigate to /api/cluster-status.</td>
</tr>
</tbody>
</table>

## Manage users and usergroups locally on SPS

Contains the endpoints for managing users and usergroups locally on SPS.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/aaa/local_database

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists the endpoints of the local database.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/local_database

Response

The following is a sample response received when listing the endpoint.

For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "groups",
            "meta": {
                "href": "/api/configuration/aaa/local_database/groups"
            }
        },
        {
            "key": "users",
            "meta": {
                "href": "/api/configuration/aaa/local_database/users"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/aaa/acls",
        "href": "/api/configuration/aaa/local_database",
        "last": "/api/configuration/aaa/settings",
        "next": "/api/configuration/aaa/settings",
        "parent": "/api/configuration/aaa",
        "previous": "/api/configuration/aaa/acls",
        "transaction": "/api/transaction"
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groups</td>
<td>Endpoint that contains local usergroups.</td>
</tr>
<tr>
<td>users</td>
<td>Endpoint that contains local usernames.</td>
</tr>
</tbody>
</table>

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
Manage usergroups locally on SPS

Contains the local usergroups of SPS. You can use local groups to control the privileges of SPS local and LDAP users — who can view and configure what. You can edit the group memberships here as well.

Note that currently you cannot edit the privileges (ACLs) of the groups using the REST API. If you change the privileges of a usergroup on the SPS web interface, the changes will apply to the users when they authenticate again on SPS, the privileges of active sessions are not affected.

URL

GET https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions.
Sample request

The following command lists the local usergroups.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups

Response

The following is a sample response received when querying a particular usergroup endpoint.

For details of the meta object, see Message format on page 10.

```
{
  "body": {
    "members": [],
    "name": "http-write"
  },
  "key": "ca2dc85730ca082ee6b5c8",
  "meta": {
    "first": "/api/configuration/aaa/local_database/groups/224696054489c27f6c5710",
    "href": "/api/configuration/aaa/local_database/groups/ca2dc85730ca082ee6b5c8",
    "last": "/api/configuration/aaa/local_database/groups/ca2dc85730ca082ee6b5f8",
    "next": "/api/configuration/aaa/local_database/groups/b080b1ba546232548bb1f9",
    "parent": "/api/configuration/aaa/local_database/groups",
    "previous": "/api/configuration/aaa/local_database/groups/b080b1ba546232548bb1a9",
    "transaction": "/api/transaction"
  }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>Top level element (JSON object)</td>
<td>Contains the properties of the usergroup.</td>
</tr>
<tr>
<td>members</td>
<td>list</td>
<td>Lists the names of the users belonging to</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the group.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
<tr>
<td>409</td>
<td>NoTransaction</td>
<td>No open Transaction is available. You must open a transaction first (for details, see Open a transaction on page 29).</td>
</tr>
</tbody>
</table>

**Add new local usergroup**

To create a new local usergroup, you have to POST the name and members of the group as a JSON object to the https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups endpoint. For details, see Create a new object on page 45.

1. **Open a transaction.**
   
   For details, see Open a transaction on page 29.

2. **Create a new usergroup.**
   
   POST the name of the group and the list of member accounts as a JSON object to the https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups endpoint. The body of the POST request should be the following. Note that you must refer to existing user accounts, and use their reference IDs, not their usernames.
For example:

```bash
curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups --data '{"name": "new-usergroup", "members": ["46785097158061f46c63d0", "1362061674580df4e00620d"]}'
```

If the POST request is successful, the response includes a reference ID for the usergroup object.

3. **Commit your changes.**

   For details, see Commit a transaction on page 31.

### Delete usergroup

To delete a usergroup, you have to:

1. Open a transaction (for details, see Open a transaction on page 29).
2. DELETE the `https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups/<ID-of-the-group>` endpoint. For details, see Delete an object on page 43. If the DELETE request is successful, the response includes only the meta object, for example:

   ```json
   {
     "meta": {
       "href": "https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups/b080b1ba546232548bb1a9",
       "parent": "https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups"
     }
   }
   ```

3. Commit your changes to actually delete the object from SPS (for details, see Commit a transaction on page 31).

### Delete user from usergroup

To delete a user from a usergroup, you have to:

1. Open a transaction (for details, see Open a transaction on page 29).
2. Create an updated version of the usergroup object that does not include the user you want to delete.
3. PUT the updated usergroup object to the https://<IP-address-of-SPS>/api/configuration/aaa/local_database/groups/<ID-of-the-group> endpoint. For details, see Delete an object on page 43.

4. Commit your changes to actually delete the object from SPS (for details, see Commit a transaction on page 31).

Manage users locally on SPS

Contains the local users of SPS. You can use local users and groups to control the privileges of SPS local and LDAP users — who can view and configure what.

NOTE: The admin user is available by default and has all possible privileges. It is not possible to delete this user.

Local users cannot be managed when LDAP authentication is used. When LDAP authentication is enabled, the accounts of local users is disabled, but they are not deleted.

When using RADIUS authentication together with local users, the users are authenticated to the RADIUS server, only their group memberships must be managed locally on SPS. For details, see Login settings on page 208.

URL

GET https://<IP-address-of-SPS>/api/configuration/aaa/local_database/users

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists the local users.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/local_database/users

The following command displays the parameters of a specific user.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/local_database/users/<ID-of-the-user>

Response

The following is a sample response received when querying the list of users.
For details of the meta object, see Message format on page 10.

```json
{
  "items": [
    {
      "key": "103640099357f3b14f0529a",
      "meta": {
        "href": "/api/configuration/aaa/local_database/users/103640099357f3b14f0529a"
      }
    },
    {
      "key": "46785097158061f46c63d0",
      "meta": {
        "href": "/api/configuration/aaa/local_database/users/46785097158061f46c63d0"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa/local_database/groups",
    "href": "/api/configuration/aaa/local_database/users",
    "last": "/api/configuration/aaa/local_database/users",
    "next": null,
    "parent": "/api/configuration/aaa/local_database",
    "previous": "/api/configuration/aaa/local_database/groups",
    "transaction": "/api/transaction"
  }
}
```

The following is a sample response received when querying a specific user.
{  
  "body": {  
    "name": "testuser",  
    "password": {  
      "key": "8f84d7d1-9de1-429a-a7a7-c33a61cc7419",  
      "meta": {  
        "href": "/api/configuration/passwords/8f84d7d1-9de1-429a-a7a7-c33a61cc7419"
      }
    },  
    "password_created": 1476796261
  },  
  "key": "46785097158061f46c63d0",  
  "meta": {  
    "first": "/api/configuration/aaa/local_database/users/103640099357f3b14f0529a",  
    "href": "/api/configuration/aaa/local_database/users/46785097158061f46c63d0",  
    "last": "/api/configuration/aaa/local_database/users/46785097158061f46c63d0",  
    "next": null,  
    "parent": "/api/configuration/aaa/local_database/users",  
    "previous": "/api/configuration/aaa/local_database/users/103640099357f3b14f0529a",  
    "transaction": "/api/transaction"
  }
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>Top level element (JSON object)</td>
<td>Contains the properties of the user.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The username of the user account.</td>
</tr>
<tr>
<td>password</td>
<td>reference</td>
<td>A reference to a password object. To create or update passwords, see Passwords stored on SPS on page 262.</td>
</tr>
<tr>
<td>password_created</td>
<td>integer</td>
<td>The date when the password of the account was changed in UNIX timestamp format (for example, 1476796261).</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the user.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
### Code Description Notes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>SemanticError</td>
<td>You tried to reuse a password object. You can use a password object for only one purpose, that is, you cannot reference a password object twice.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
<tr>
<td>409</td>
<td>NoTransaction</td>
<td>No open Transaction is available. You must open a transaction first (for details, see Open a transaction on page 29).</td>
</tr>
</tbody>
</table>

## Configuring LDAP servers

Configure LDAP AD and LDAP POSIX servers for authentication, authorization, and accounting (AAA).

### URL

POST https://<IP-address-of-SPS>/api/configuration/aaa/ldap_servers

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS.
### Operations

Operations with the /aaa/ldap_servers endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating a new LDAP server</td>
<td>POST</td>
<td>/api/configuration/aaa/ldap_servers</td>
<td></td>
</tr>
<tr>
<td>Retrieving a LDAP server</td>
<td>GET</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Sample request

The following command creates a new LDAP server.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/ldap_servers
```

The following is a sample request.

```json
{
   "name": "ldap-server-name",
   "schema": {
      "selection": "posix",
      "username_attribute": "uid",
      "membership_check": {
         "enabled": true,
         "member_uid_attribute": "memberUid"
      },
      "memberof_check": {
         "enabled": true,
         "memberof_user_attribute": "memberOf",
         "memberof_group_objectclass": "groupOfNames"
      },
      "user_dn_in_groups": [
      {
         "object_class": "groupOfNames",
         "attribute": "member"
      },
      {
         "object_class": "groupOfUniqueNames",
         "attribute": "member"
      }
   }
}
```
The following is a sample response received when you retrieve LDAP server information. For details of the meta object, see Message format on page 10.

```json
{
  "items": [
    {
      "body": {
        "bind_dn": null,
        "bind_password": null,
        "encryption": {
          "selection": "disabled"
        },
        "group_base_dn": "ou=Groups,dc=example",
        "name": "ldap-server-name",
        "schema": {
          "memberof_check": {
            "enabled": true,
            "memberof_group_objectclass": "groupOfNames",
            "memberof_user_attribute": "memberOf"
          },
          "membership_check": {
            "enabled": true,
            "member_uid_attribute": "memberUid"
          }
        }
      }
    },
    {
      "host": {
        "selection": "ip",
        "value": "10.110.0.1"
      },
      "port": 389
    },
    "user_base_dn": "ou=People,dc=example",
    "group_base_dn": "ou=Groups,dc=example",
    "bind_dn": null,
    "bind_password": null,
    "encryption": {
      "selection": "disabled"
    }
  ]
}
```
Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>bind_dn</td>
<td>string</td>
<td>The Distinguished Name that SPS should use to bind to the LDAP directory.</td>
<td>NOTE: SPS accepts both pre Windows 2000-style and Windows 2003-style account names, or User Principal Names (UPNs). For example, <a href="mailto:administrator@example.com">administrator@example.com</a> is also accepted.</td>
</tr>
<tr>
<td>bind_password</td>
<td>null</td>
<td>References the password</td>
<td>NOTE: One Identity</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
|         |        | SPS uses to authenticate on the server. You can configure passwords at the /api/configuration/passwords/ endpoint. Must be used if the value of the selection element is set to ldap. To modify or add a password, use the value of the returned key as the value of the password element, and remove any child elements (including the key). | Safeguard for Privileged Sessions (SPS) accepts passwords that are not longer than 150 characters and supports the following characters:  
  - Letters A-Z, a-z  
  - Numbers 0-9  
  - The space character  
  - Special characters: !"#$%&'()\[\]|\^\`\{\}_\-|The displayed value type depends on the encryption.selection parameter. |
| encryption | union  | Configuration settings for encrypting the communication between SPS and the LDAP server.                                                                                                                                 |                                                                                                                                                                                                       |
| encryption.selection | enum   | Defines the type of encryption SPS uses to communicate with the LDAP server. Possible values are:  
  - disabled: The communication is not encrypted.  
  - ssl: TLS/SSL encryption. To use a TLS-encrypted with certificate verification to connect to the LDAP server, use the full domain name (for example ldap.example.com) as the server address, otherwise the certificate verification might fail. The name of the LDAP server | Example if the value is disabled  
```json
{
    "selection": "disabled"
}
```
Example if the value is ssl or starttls:  
```json
{
    "selection": "ssl | starttls",
    "trust_store": null,<trust-store-ref>,
    "client_authentication": null,<x509-ref>
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Top level element, the name of the object. This name is also displayed on</td>
<td>Example with LDAP AD server, where the schema selection is ad:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the SPS web interface. It cannot contain whitespace.</td>
<td></td>
</tr>
<tr>
<td>group_base_dn</td>
<td>string</td>
<td>Name of the DN to be used as the base of queries regarding groups.</td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>NOTE:</strong> You must fill in this field. It is OK to use the same value for</td>
<td>&quot;ou=users,ou=example&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>user_base_dn and group_base_dn.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>However, note that specifying a sufficiently narrow base for the LDAP</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>subtrees where users and groups are stored can speed up LDAP operations.</td>
<td></td>
</tr>
<tr>
<td>schema</td>
<td>union</td>
<td>AD and POSIX specific LDAP configuration.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>NOTE:</strong> TLS-encrypted connection to Microsoft Active Directory is supported</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>only on Windows 2003 Server and newer platforms. Windows 2000 Server is not</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>supported.</td>
<td></td>
</tr>
<tr>
<td>starttls</td>
<td></td>
<td>Opportunistic TLS.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>must appear in the Common Name of the certificate.</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** TLS-encrypted connection to Microsoft Active Directory is supported only on Windows 2003 Server and newer platforms. Windows 2000 Server is not supported.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ad</td>
<td></td>
<td>&quot;ad&quot;,                                                                                             &quot;memberof_check&quot;: ...,                                                                                             &quot;memberof_check&quot;: ...,                                                                                             &quot;user_dn_in_groups&quot;: ...                                                                                             }</td>
<td></td>
</tr>
<tr>
<td>memberof_check</td>
<td>object</td>
<td>The Enable checking for group DN in user objects setting allows checking a configurable attribute in the user object. This attribute contains a list of group DN the user is additionally a member of. This user attribute is usually memberOf.</td>
<td>Example with LDAP AD server, if memberof_check and memberof_user_attribute are enabled:</td>
</tr>
<tr>
<td>schema.memberof_check</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Example with LDAP POSIX server, if memberof_check is enabled:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;schema&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;memberof_check&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;memberof_group_objectclass&quot;: &quot;groupOfNames&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;memberof_user_attribute&quot;: &quot;memberOf&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>To disable memberof_check for both LDAP AD and LDAP POSIX servers, set the enabled parameter to false.</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;schema&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;memberof_check&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;enabled&quot;: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

- schema.memberof_check.enabled  boolean  To enable memberof_check, set it to true.
- schema.memberof_check.memberof_group_objectclass  string  The attribute holding the members of the LDAP group.  This field is case-sensitive.  Default value: groupOfNames
- schema.memberof_  string  Must be used if the  This attribute is the same
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>check.memberof_user_attribute</td>
<td>memberof_check is set to true. The name of the user attribute (for example, memberof) that contains the group DNs.</td>
<td>for both LDAP AD and LDAP POSIX schema.</td>
<td></td>
</tr>
<tr>
<td>schema.membership_check</td>
<td>object</td>
<td>• POSIX: POSIX primary and supplementary group membership checking.</td>
<td>• Example with LDAP AD server, if membership_check and nested_groups are enabled:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AD: Active Directory specific non-primary group membership checking.</td>
<td>&quot;membership_check&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;nested_groups&quot;: true</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Example with LDAP POSIX server, if membership_check and member_uid_attribute are enabled:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;membership_check&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;member_uid_attribute&quot;: null</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• To disable membership_check for both LDAP AD and LDAP POSIX servers, set the enabled parameter to false.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>schema.membership_check.enabled</td>
<td>boolean</td>
<td>POSIX: Enables POSIX primary and supplementary group membership checking. AD: Enables Active Directory specific non-primary group membership checking.</td>
<td></td>
</tr>
<tr>
<td>schema.membership_check.member_uid_attribute</td>
<td>string</td>
<td>Must be used if the value of the selection element is set to posix. The POSIX group membership attribute name is the name of the attribute in a posixGroup group object, which lists the plain usernames that are members of the group. These groups are usually referred to as supplementary groups of the referred user. Can be null.</td>
<td>Default value: memberUid</td>
</tr>
<tr>
<td>schema.selection</td>
<td>string</td>
<td>Configures which LDAP schema to use: AD or POSIX. Possible values are:</td>
<td>Default value: ad, posix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ad: Microsoft Active Directory server. For details and examples, see Configuring LDAP servers.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• posix: The server uses the POSIX LDAP scheme.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>member_uid_attribute and username_attribute elements. For details and examples, see Configuring LDAP servers.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>schema.user_dn_in_groups</code></td>
<td>array</td>
<td>Add object_class / attribute pairs. SPS will search for the user DN in the group's attribute defined here. If it finds the user DN there, SPS considers the user the member of that group.</td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

```json
"user_dn_in_groups": [
    
    
    "attribute":
    "member",

    "object_class":
    "groupOfNames"

    

    
    "attribute":
    "uniqueMember",

    "object_class":
    "groupOfUniqueNames"

]```

The array can return empty:

```json
"user_dn_in_groups": []```

**User_dn_in_groups** can serve as additional validation. At least one out of membership_check, memberof_check or user_dn_in_groups must be filled for validation.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>schema.user_dn_in_groups.attribute</code></td>
<td>string</td>
<td>Name of the group attribute which contains the user DN.</td>
<td></td>
</tr>
</tbody>
</table>

Default value: member
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>schema.user_dn_in_groups.object_class</td>
<td>string</td>
<td>Consider groups of this objectClass.</td>
<td>Possible values: groupOfNames, groupOfUniqueNames</td>
</tr>
<tr>
<td>schema.username_attribute</td>
<td>string</td>
<td>The login attribute that uniquely identifies a single user record.</td>
<td>Default value: uid</td>
</tr>
<tr>
<td>servers</td>
<td>array</td>
<td>Contains the addresses and ports of the LDAP servers.</td>
<td>The displayed value type depends on the servers.host.selection parameter. Possible values are: ip, fqdn</td>
</tr>
</tbody>
</table>

- Example if the host selection is ip:

```json
"servers": [
{
"host":{
  "selection": "ip",
  "value": "1.2.3.4"
},
"port": 123
}
]
```

- Example, if the host selection is fqdn:

```json
"servers": [
{
"host": {
}
}
]"
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>servers.host</td>
<td>object</td>
<td>Contains the address of the LDAP server.</td>
<td></td>
</tr>
<tr>
<td>servers.host.selection</td>
<td>string</td>
<td>Defines the address type (IP or domain name).</td>
<td>Possible values are:</td>
</tr>
<tr>
<td>servers.host.value</td>
<td>string</td>
<td>The address of the LDAP server.</td>
<td></td>
</tr>
<tr>
<td>servers.port</td>
<td>int</td>
<td>The port of the LDAP server.</td>
<td></td>
</tr>
<tr>
<td>user_base_dn</td>
<td>string</td>
<td>Name of the DN to be used as the base of queries regarding users.</td>
<td>For example:</td>
</tr>
</tbody>
</table>

NOTE: You must fill in this field. It is OK to use the same value for user_base_dn and group_base_dn.
However, note that specifying a sufficiently narrow base for the LDAP subtrees where users and groups are stored can speed up LDAP operations.

Configure LDAP servers

To configure an LDAP server, you have to:

1. Open a transaction.
   For more information, see Open a transaction on page 29.
2. Create the JSON object for the new LDAP server configuration.
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/aaa/ldap_servers endpoint. You can find a detailed description of the available parameters listed in Element.
3. Commit your changes.
   For more information, see Commit a transaction on page 31.

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

Configuring SPS login methods

Use the /aaa/login_methods endpoint to configure multiple login methods for SPS. Possible login methods are the following:

- Local login
- LDAP login (AD or POSIX)
- X509 login
- RADIUS login

URL

POST https://<IP-address-of-SPS>/api/configuration/aaa/login_methods
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d03e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Operations

Operations with the /aaa/login_methods endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring the Local login method</td>
<td>POST</td>
<td>/api/configuration/aaa/login_methods</td>
<td>NOTE: Changing your password is possible only with Local login. Before you can configure the X509 login method, you must upload your X509 certificate to a trust store at the /api/configuration/trust_stores endpoint. Before you can configure the LDAP login method, you must create a new LDAP server at the /api/configuration/aaa/ldap_servers endpoint.</td>
</tr>
<tr>
<td>Configuring the X509 login method</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuring the password-based LDAP login method (AD or POSIX)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuring the RADIUS login method</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Operation Table

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reordering login</td>
<td>PUT</td>
<td>/api/configuration/aaa/login_methods/@order</td>
<td>By default, the Local login method button appears as the first login method on the SPS web interface. To reorder the login method buttons, use the /@order endpoint.</td>
</tr>
</tbody>
</table>

#### Sample request

The following command creates a new login method.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/login_methods
```

The following parameters are applicable to all login methods - Local, LDAP, X509, and RADIUS.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>enum</td>
<td>The login method type used for authentication.</td>
<td>Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• local</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Use a local user database for authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• x509</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Use a X509 certificate for authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• ldap</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Use an LDAP server (AD or POSIX) for authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• radius</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Use a RADIUS server for authentication.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>A unique identifier of the login method.</td>
<td></td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>The title that appears above the login method button on the SPS web</td>
<td>In the case of X509 login method, there is</td>
</tr>
</tbody>
</table>
Example: Sample request for configuring Local login

```
{
    "type": "local",
    "name": "inactive-local",
    "title": "Local login",
    "active": false,
    "cracklib_enabled": true,
    "expiration_days": 30,
    "remember_previous_passwords": 10,
    "minimum_password_strength": "good",
    "minimum_password_length": 1
}
```

Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>cracklib_enabled</td>
<td>boolean</td>
<td>Password setting. Must be used if the value of the selection element is set to local. Set to true to test the strength of user passwords with simple dictionary attacks before they are committed. Set to false if a RADIUS server or X.509 certificate is used for authentication.</td>
<td>NOTE: The strength of a password is determined by its length and complexity: the variety of numbers, letters, capital letters, and special characters used. To run simple dictionary-based attacks to find weak passwords, enable Cracklib (eg. dictionary) protection.</td>
</tr>
<tr>
<td>expiration_days</td>
<td>integer</td>
<td>Password setting. Configures the number of days the user passwords</td>
<td>Set to 0 if a RADIUS server or X.509 certificate is used for authentication.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>are considered valid.</td>
<td></td>
<td>Expired passwords must be changed upon login.</td>
<td>The 0 value means the passwords do not expire. The highest value you can configure is 365. Must be used if the value of the selection element is set to local.</td>
</tr>
<tr>
<td>remember_previous_passwords</td>
<td>integer</td>
<td>Password setting. Configures the number of previous passwords to retain to prevent password reuse.</td>
<td>Set to 0 if a RADIUS server or X.509 certificate is used for authentication. The 0 value means passwords can be reused. Must be used if the value of the selection element is set to local.</td>
</tr>
<tr>
<td>minimum_password_strength</td>
<td>string</td>
<td>Password setting. Configures the required password strength for new passwords.</td>
<td>Set to disabled if a RADIUS server or X.509 certificate is used for authentication. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• disabled                                                                                 Any password is accepted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• good                                                                                   Weak passwords are not accepted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• strong                                                                                 Only strong passwords are accepted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Must be used if the value of the selection element is set to local.</td>
</tr>
<tr>
<td>minimum_password_length</td>
<td>number</td>
<td>The minimum number of characters the password must consist of.</td>
<td></td>
</tr>
</tbody>
</table>
Example: Sample request for configuring LDAP login

```json
{
  "type": "ldap",
  "name": "ldap",
  "title": "LDAP login",
  "active": true,
  "ldap_server": "5ed1b422-395a-4d5e-abf0-b43cd6c752f8"
}
```

Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap_server</td>
<td>string</td>
<td>The identifier of the previously created LDAP server.</td>
<td></td>
</tr>
</tbody>
</table>

Example: Sample request for configuring X509 login

```json
{
  "type": "x509",
  "name": "x509",
  "title": "X509 login",
  "active": true,
  "trust_store": "5ed1b422-395a-4d5e-abf0-b43cd6c752f8",
  "username_attribute": "commonName",
  "groups": {
    "type": "local"
  }
}
```

Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>trust_store</td>
<td>string</td>
<td>The identifier of the X509 certificate that was previously uploaded into</td>
<td>Trust stores serve as local certificate storages where users can store the</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Trust stores.</td>
<td></td>
<td>certificate chains of trusted Certificate Authorities (CAs).</td>
<td></td>
</tr>
</tbody>
</table>
| username_attribute    | enum          | The login attribute that uniquely identifies a single user record. You can choose which type of username you want to use. | Possible values are:  
  - commonName  
  - emailAddress  
  - userId  
  - userPrincipalName |
| groups                | object        | The location of your group from which you will authorize access.             |                                                                                                 |
| groups.type           | enum          | The type of the group.                                                      | Possible values are:  
  - local  
  - ldap  
  If the group type is ldap, the identifier of the previously created LDAP server (the value of the ldap_server field) will appear. |

**Example: Configuring X509 login with LDAP groups**

*To configure a X509 login with LDAP groups*

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. Create a new LDAP server at the `/api/configuration/aaa/ldap_servers` endpoint.

   ```javascript
   {
     "name": "ldap_server",
     "schema": {
       "selection": "posix",
       "username_attribute": "uid",
     },
   }
   ```
3. Upload a X509 certificate at the /api/configuration/trust_stores endpoint.

```json
{
    "name": "X509_Trust_Store",
    "authorities": [
        "-----BEGIN CERTIFICATE-----
```
4. Create a new LDAP login method at the `/api/configuration/aaa/login_methods` endpoint.

```json
{
  "type": "x509",
  "name": "x509",
  "title": "X509 login",
  "active": true,
  "trust_store": "5ed1b422-395a-4d5e-abf0-b43cd6c752f8",
  "username_attribute": "commonName",
  "groups": {
    "type": "ldap",
    "ldap_server": "5ed1b422-395a-4d5e-abf0-b43cd6c752f8"
  }
}
```

5. Commit your changes.

For more information, see Commit a transaction on page 31.
Example: Configuring RADIUS login with local groups

To configure a RADIUS login with local groups

1. Open a transaction.
   
   For more information, see Open a transaction on page 29.

2. Create a new secret for your RADIUS server at the /api/configuration/passwords/ endpoint. For example, any.secret.

3. Create a new RADIUS login method at the /api/configuration/aaa/login_methods endpoint.

```
{
   "type": "radius",
   "name": "radius",
   "title": "Radius login",
   "active": true,
   "servers": [
      {
         "address": {
            "selection": "ip",
            "value": "4.5.6.7"
         },
         "port": 1812,
         "shared_secret": "<key' from the response of the penultimate creation>"
      },
      {
         "address": {
            "selection": "fqdn",
            "value": "radius.example"
         },
         "port": 18120,
         "shared_secret": "<key' from the response of the last creation>"
      }
   ],
   "authentication_protocol": "pap",
   "groups": {
      "type": "local"
   }
}
```
4. Commit your changes.

For more information, see Commit a transaction on page 31.

Example: Configuring RADIUS login with LDAP groups (AD or POSIX)

To configure a RADIUS login with LDAP groups

1. Open a transaction.

   For more information, see Open a transaction on page 29.

2. Create a new LDAP server at the /api/configuration/aaa/ldap_servers endpoint.

3. Create a new secret for your RADIUS server at the /api/configuration/passwords/ endpoint. For example, any.secret.

4. Create a new LDAP server login method at the /api/configuration/aaa/login_methods endpoint.

   ```json
   {
     "type": "radius",
     "name": "radius",
     "title": "Radius login",
     "active": true,
     "servers": [
       {
         "address": {
           "selection": "fqdn",
           "value": "radius.balabit"
         },
         "port": 18120,
         "shared_secret": "<\'key\' from the response of the last creation>"
       }
     ],
     "authentication_protocol": "pap",
     "groups": {
   ```
5. **Commit your changes.**

For more information, see [Commit a transaction](#) on page 31.

**Response**

For details of the meta object, see [Message format](#) on page 10.

**HTTP response codes**

For more information and a complete list of standard HTTP response codes, see [Application level error codes](#) on page 37.
Managing SPS

Troubleshooting options

Configures debug logging and the retention time of core dump files.

- Debug logging increases the log level of the non-network-related events, adding the commands executed by the SPS web interface to the log.
- SPS automatically generates core dump files if an important software component of the system crashes. These core dump files can be of great help to the One Identity Support Team to identify problems. To download the generated core dump files, navigate to Basic Settings > Troubleshooting > Core files on the web interface of SPS.

URL

GET https://<IP-address-of-SPS>/api/configuration/troubleshooting

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command queries the troubleshooting settings.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/troubleshooting
```

Response

The following is a sample response received.

For details of the meta object, see Message format on page 10.

```json
{
   "body": {
      "core_files": {
         "retention_days": 14
      },
      "debug_logging": {
         "enabled": true
      }
   },
   "key": "troubleshooting",
   "meta": {
      "first": "/api/configuration/aaa",
      "href": "/api/configuration/troubleshooting",
      "last": "/api/configuration/x509",
      "next": "/api/configuration/vnc",
      "parent": "/api/configuration",
      "previous": "/api/configuration/telnet",
      "transaction": "/api/transaction"
   }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>Contains the troubleshooting settings.</td>
</tr>
<tr>
<td></td>
<td>element</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(string)</td>
<td></td>
</tr>
<tr>
<td>core_files</td>
<td>Top level</td>
<td>Contains the settings for core dump file retention.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>retention_days</td>
<td>int</td>
<td>Retention time for core files, in days.</td>
</tr>
<tr>
<td>debug</td>
<td>Top level</td>
<td>Settings for debug logging.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Element

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>logging</td>
<td>item</td>
<td>Set to true to increase the log level of the non-network-related events, adding the commands executed by the SPS web interface to the log.</td>
</tr>
</tbody>
</table>

### Modify troubleshooting settings

To modify troubleshooting settings, you have to:

1. **Open a transaction.**
   - For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the troubleshooting options.**
   - PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/troubleshooting endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   - For more information, see Commit a transaction on page 31.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Internal certificates

This endpoint references the certificates of SPS's internal Certificate Authority, Timestamping Authority, and the SSL certificate of the web and REST interface.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/management/certificates
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following request command lists the internal certificates of SPS.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/certificates
```

**Response**

The following is a sample response received when listing the internal certificates of SPS.

For details of the *meta* object, see **Message format** on page 10.

```
{
   "body": {
      "ca": {
         "selection": "identity",
         "x509_identity": {
```
"key": "fbd684e1-e1ac-4f34-ad25-86c560c51e24",
"meta": {
    "href": "/api/configuration/x509/fbd684e1-e1ac-4f34-ad25-86c560c51e24"
}
},
"server": {
    "key": "fd1c73e8-bcb8-4d13-991f-722f492dc074",
    "meta": {
        "href": "/api/configuration/x509/fd1c73e8-bcb8-4d13-991f-722f492dc074"
    }
},
"tsa": {
    "key": "20e72ede-78ef-460a-b843-68a35d994142",
    "meta": {
        "href": "/api/configuration/x509/20e72ede-78ef-460a-b843-68a35d994142"
    }
},
"key": "certificates",
"meta": {
    "first": "/api/configuration/management/certificates",
    "href": "/api/configuration/management/certificates",
    "last": "/api/configuration/management/webinterface",
    "next": "/api/configuration/management/disk_fillup_prevention",
    "parent": "/api/configuration/management",
    "previous": null,
    "transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>The ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the internal certificates of SPS.</td>
</tr>
<tr>
<td>ca</td>
<td>Top level item</td>
<td>Contains the certificate of SPS's internal Certificate Authority.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Must be set to identity.</td>
</tr>
<tr>
<td>x509_identity</td>
<td>string</td>
<td>References the certificate of SPS's internal Certificate Authority. You can configure certificates at the /api/configuration/x509/ endpoint. To modify or add an X.509 certificate, use the</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>server</td>
<td>string</td>
<td>References the SSL certificate of SPS’s web interface. You can configure certificates at the /api/configuration/x509/ endpoint. To modify or add an X.509 certificate, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key). For details, see Certificates stored on SPS on page 287.</td>
</tr>
<tr>
<td>tsa</td>
<td>string</td>
<td>References the certificate of SPS’s internal Timestamping Authority. You can configure certificates at the /api/configuration/x509/ endpoint. To modify or add an X.509 certificate, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key). For details, see Certificates stored on SPS on page 287.</td>
</tr>
</tbody>
</table>

Modify a certificate

To modify a certificate, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create a CA**
   
   Have the value of the key element of a valid X.509 CA certificate stored on SPS.

3. **Modify the JSON object of the endpoint.**
   
   Use the X.509 certificate’s key as the value of the ca element. You can find a detailed description of the available parameters listed in Element. PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/management/certificates endpoint.

4. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Passwords stored on SPS

To create a new password, you have to POST the password or its hash as a JSON object to the https://<IP-address-of-SPS>/api/passwords endpoint. For details, see Create a new object on page 45. The body of the POST request must contain a JSON object with the parameters listed in Password parameters. The response to a successful POST message is a JSON object that includes the reference ID of the created password in its key attribute. You can reference this ID in other parts of the configuration, for example, to set the password of a user account. Note that you can use a password object for only one purpose, that is, you cannot reference a password object twice.

### URL

**POST** https://<IP-address-of-SPS>/api/configuration/passwords

- Note that the GET method is not permitted on this endpoint, you cannot list the existing passwords. However, if you know the reference ID of a password, you can display its properties:

  **GET** https://<IP-address-of-SPS>/api/configuration/passwords/<reference-ID-of-the-password>;

- You cannot directly delete or modify a password, the DELETE and PUT methods are not permitted on password objects. To update a password, create a new one, then update the object that uses the old password to reference the new password.
Table 3: Headers

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the type of the data sent. SPS uses the JSON format</td>
<td>Required</td>
<td>application/json</td>
</tr>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API.</td>
</tr>
</tbody>
</table>

Sample request

The following command creates a new password object.

curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/passwords --data '{"plain": "newpassword"}'

If you do not want to include the actual password in the request, the SHA-256 hash of the password is enough:

curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/passwords --data '{"hash": "$6$rounds=5000$If20/EFyQ4dw3dg/$xrECLfXgZ1C2Xr1s257E2aZen42fM7R.sOGG9pkPy1x5ORTx6j03oPWexViB3f5wnaZOQCBF.Nj1Dgyg2WEe."}"

Table 4: Password parameters

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hash</td>
<td>string</td>
<td>Must contain the SHA-256 hash of the password to be created, for example, &quot;hash&quot;: &quot;ddec437ebe1da25a146a24c432d1165bc646daa7fecc6aa14c636265c83ca a14&quot;.</td>
</tr>
<tr>
<td>nthash</td>
<td>string</td>
<td>Optional. Contains the NT-HASH of the password to be created, for example, &quot;nthash&quot;: &quot;2c01a73ad9e597f6eab0d072ed74616c&quot;</td>
</tr>
<tr>
<td>plain</td>
<td>string</td>
<td>Contains the password in plain-text format, for example, &quot;plain&quot;: &quot;mypassword&quot;.</td>
</tr>
</tbody>
</table>

When choosing the format of your password, the request must contain one of these formats:
- The password format is only in plain-text
- The password format is only a hash
- The password format is only a nthash
- The password format can be both a hash and a nthash.

**Response**

The response to a successful POST message is a JSON object that includes the reference ID of the created password in its key attribute.

For details of the meta object, see Message format on page 10.

```
{
    "key": "faa96916-c85e-46ff-8697-f4cc5e596e7f",
    "meta": {
        "href": "/api/configuration/passwords/faa96916-c85e-46ff-8697-f4cc5e596e7f",
        "parent": "/api/configuration/passwords",
        "transaction": "/api/transaction"
    }
}
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>The method &lt;method&gt; is not allowed for this node.</td>
</tr>
</tbody>
</table>
Modify or delete password

You cannot directly delete or modify a password, the DELETE and PUT methods are not permitted on password objects. To update a password, create a new one, then update the object that uses the old password to reference the new password. After you commit the transaction, SPS will automatically delete the old password. For details, see Change the admin password.

Change the admin password

To change the password of the admin user, complete the following steps.

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create a new password object**
   
   POST a JSON object containing the password or the hash of the password to the https://<IP-address-of-SPS>/api/passwords endpoint. For details, see Password parameters. For example:

   ```
curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/passwords --data '{"plain": "mypassword"}'
   ```

   If the operation is successful, the response includes a reference key to the new password object.

3. **Reference the key of the password in the user configuration.**
   
   Modify the JSON object of the user to reference the key of the new password object, and PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/aaa/local_database/users/<key-of-the-user> endpoint. For example:

   ```
curl -X PUT -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/local_database/users/14322374245a7de542bbb04 --data '{"name": "admin", "password": "<key-of-the-new-password>"}''
   ```

4. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

Change the root password

To change the password of the root user, complete the following steps.
1. **Open a transaction.**
   For more information, see [Open a transaction](#) on page 29.

2. **Create a new password object**
   POST a JSON object containing the password or the hash of the password to the https://<IP-address-of-SPS>/api/passwords endpoint. For details, see [Password parameters](#). For example:
   ```
   curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/passwords --data '{"plain": "mypassword"}"
   ```
   If the operation is successful, the response includes a reference key to the new password object.

3. **Configure SPS to use this password for the root user configuration.**
   PUT the reference key of the new password object to the https://<IP-address-of-SPS>/api/configuration/management/root_password endpoint. For example:
   ```
   ```
   Note that you must PUT the reference key as a JSON string, enclosed in double-quotes.

4. Alternatively, instead of performing the previous two steps, you can replace an existing password in a single step, PUT the following JSON object to the https://<IP-address-of-SPS>/api/configuration/management/root_password endpoint:
   ```
   {
       "plain": "new_password"
   }
   ```

5. **Commit your changes.**
   For more information, see [Commit a transaction](#) on page 31.

**Change the user password**

Logged in users can change their own passwords by completing the following steps.

1. **Open a transaction.**
   For more information, see [Open a transaction](#) on page 29.
2. **Create a new password object**

POST a JSON object containing the password or the hash of the password to the https://<IP-address-of-SPS>/api/passwords endpoint. For details, see Password parameters. For example:

```bash
curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/passwords --data '{"plain": "mypassword"}'
```

If the operation is successful, the response includes a reference key to the new password object.

3. **Change the password of the user.**

PUT a JSON object that includes the current password in plain text and the key of the new password object to the https://<IP-address-of-SPS>/api/user/password endpoint. For example:

```bash
curl -X PUT -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/user/password --data '{"current_password_in_plaintext": "<old-password>", "new_password_reference": "<key-of-the-new-password>"}"
```

4. Alternatively, instead of performing the previous two steps, you can replace an existing password in a single step, PUT the following JSON object to the https://<IP-address-of-SPS>/api/configuration/management/root_password endpoint:

```json
{
    "current_password_in_plaintext": "<current_password_in_plaintext>",
    "new_password_reference": {
        "plain": "newpassword"
    }
}
```

5. **Commit your changes.**

For more information, see Commit a transaction on page 31.

### Private keys stored on SPS

To create a new private key, you have to POST the private key as a JSON object to the https://<IP-address-of-SPS>/api/private_keys endpoint. For details, see Create a new object on page 45. The body of the POST request must contain a JSON object with the parameters listed in Element. The response to a successful POST message is a JSON object that includes the reference ID of the created private key in its key attribute. You can...
reference this ID in other parts of the configuration. Note that you can use a private-key object for only one purpose, that is, you cannot reference one object twice.

**URL**

| POST  https://<IP-address-of-SPS>/api/configuration/private_keys |

- Note that the GET method is not permitted on this endpoint, you cannot list the existing private keys. However, if you know the reference ID of a private key, you can display its properties:

  | GET  https://<IP-address-of-SPS>/api/configuration/private_keys/<reference-ID-of-the-private-key> |

- You cannot directly delete or modify a private key, the DELETE and PUT methods are not permitted on private key objects. To update a private key, create a new one, then update the object that uses the old private key to reference the new private key.

**Table 5: Headers**

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the type of the data sent. SPS uses the JSON format</td>
<td>Required</td>
<td>application/json</td>
</tr>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API.</td>
</tr>
</tbody>
</table>

**Sample request**

The following command creates a new private key object. Note the following requirements:

- The key must be in PKCS-1 or PKCS-8 PEM format.
- Encrypted private keys are not supported.
- The body of the POST message must be the private key as a single line, enclosed in double-quotes.
- Replace line-breaks in the PEM file with \n
curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/private_keys --data "-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE25RwVKY+ALnzPZF4fUoJy
.....
I2SchDibk/Xj/ZvuE23LvxayW0VvVf0H3JZX3SU4Sa0vpaeC+3oddVTwQOWrQ0

Qbn5W3xKz4vXDDQHEsEvDQ9A7+uCEuHp04s33IK9KEa0Zdp745A05DGXN4HFzc

-----END RSA PRIVATE KEY------"

Querying a specific key returns the following information about the key:

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/private_keys/<reference-ID-of-the-private-key>;

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public-key-fingerprint</td>
<td>string</td>
<td>The fingerprint of the public key that matches the private key.</td>
</tr>
<tr>
<td>digest</td>
<td>string</td>
<td>The fingerprint of the key, for example 2048 SHA256:JPkdFkT6wU9c11bbqX53hovDo7KbIB80REfumUWD h9f no comment (RSA)</td>
</tr>
<tr>
<td>hash_algorithm</td>
<td>string</td>
<td>The type of the private key. Must be rsa</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The hash algorithm used to create the fingerprint, for example, sha256.</td>
</tr>
</tbody>
</table>

Response

The response to a successful POST message is a JSON object that includes the reference ID of the created public key in its key attribute.

For details of the meta object, see Message format on page 10.

```json
{
    "key": "faa96916-c85e-46ff-8697-f4cc5e596e7f",
    "meta": {
        "href": "/api/configuration/private_keys/faa96916-c85e-46ff-8697-f4cc5e596e7f",
        "parent": "/api/configuration/private_keys",
        "transaction": "/api/transaction"
    }
}
```
The response to querying a specific key is a JSON object that includes the parameters of the key, for example:

```json
{
    "body": {
        "public-key-fingerprint": {
            "digest": "2048 SHA256:JPKdfkT6wU9c11bbqX53hovDo7KbIB8OREfumUWdh9f no comment (RSA)",
            "hash_algorithm": "sha256"
        },
        "type": "rsa",
        "key": "6c4d1116-d79d-475b-bb37-9f844f085c14",
        "meta": {
            "first": "/api/configuration/private_keys/e5d13d18-07c5-43fa-89f4-c3d2ece17c71",
            "href": "/api/configuration/private_keys/6c4d1116-d79d-475b-bb37-9f844f085c14",
            "last": "/api/configuration/private_keys/6c4d1116-d79d-475b-bb37-9f844f085c14",
            "next": null,
            "parent": "/api/configuration/private_keys",
            "previous": "/api/configuration/private_keys/e5d13d18-07c5-43fa-89f4-c3d2ece17c71",
            "transaction": "/api/transaction"
        }
    }
}
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see *Application level error codes* on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>SyntacticError</td>
<td>Syntax error: Could not load PEM key: Unsupported private key format, only PKCS-1 and PKCS-8 is supported. Encrypted private keys are not supported.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-----------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>The method <code>&lt;method&gt;</code> is not allowed for this node.</td>
</tr>
</tbody>
</table>

**Modify or delete private key**

You cannot directly delete or modify a private key, the DELETE and PUT methods are not permitted on private key objects. To update a private key, create a new one, then update the object that uses the old private key to reference the new private key. After you commit the transaction, SPS will automatically delete the old private key.

**Private keys generated on SPS**

In some security contexts it might be a requirement to generate private keys on the appliance so that you can avoid any kind of eavesdropping during the transfer of the unencrypted key. Safeguard for Privileged Sessions supports generating Elliptic Curve (secp256r1) private keys on its REST API. You must use the REST API to use the generated key in the configuration. SPS supports the on-box generated private keys to be used for the following purposes:

- for the web server, timestamping authority or CA
  (/api/configuration/management/certificates, see Internal certificates on page 259)
- SMTP client authentication (/api/configuration/management/email, see Mail settings on page 144)
- Syslog client authentication (/api/configuration/management/syslog, see Syslog server settings on page 135)
- LDAP client authentication (for policies: /api/configuration/policies/ldap_servers, see LDAP servers on page 393, and for authentication: /api/configuration/aaa/ldap_servers, see Configuring LDAP servers on page 231)

Overview of the steps required to use on-box generated private keys in the configuration:

1. Generate a private key and a certificate signing request (CSR).
2. Obtain the CSR and send it to a certificate authority (CA). The required steps for performing the validation are mandated by the CA.
3. Once the CA signs the certificate, upload it to SPS.
4. Change the relevant REST configuration element to refer to the freshly generated 'X.509 identifier' (which is a reference to a private key and the associated certificate chain).
5. (Optional): You might want to delete the private key if you want to prevent the key to be used for a different purpose on the SPS.

**NOTE**: In this case, whenever the certificate expires, you must generate a fresh private key and CSR.

**Perequisites**: A certificate authority must be configured in Trust stores on page 187.

**URL**

https://<IP-address-of-SPS>/api/pki/certificate

**Table 6: Headers**

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the type of the data sent. SPS uses the JSON format</td>
<td>Required</td>
<td>application/json</td>
</tr>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API.</td>
</tr>
</tbody>
</table>

**Operations**

Operations with the /api/pki/certificate endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generating a new CSR</td>
<td>POST</td>
<td>/api/pki/certificate/requests</td>
<td></td>
</tr>
<tr>
<td>Adding an X.509 certificate chain to a CSR to create a X.509 identifier</td>
<td>PUT</td>
<td>/api/pki/certificate/requests/&lt;ID-of-the-CSR&gt;</td>
<td></td>
</tr>
<tr>
<td>Operation</td>
<td>HTTP method</td>
<td>URL</td>
<td>Notes</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-------------</td>
<td>------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Setting or replacing a certificate chain for a CSR without knowing the CSR identifier</td>
<td>POST</td>
<td>/api/pki/certificate</td>
<td>X.509 identifier that have been referenced in the configuration will not be updated automatically, when you replace a certificate chain for a CSR. If you want to use the newly created X.509 identifier, you must set or update the reference to it in the configuration.</td>
</tr>
<tr>
<td>Querying existing CSRs</td>
<td>GET</td>
<td>/api/pki/certificate/requests</td>
<td></td>
</tr>
<tr>
<td>Querying a single CSR</td>
<td>GET</td>
<td>/api/pki/certificate/requests/&lt;ID-of-the-CSR&gt;</td>
<td></td>
</tr>
<tr>
<td>Deleting a CSR</td>
<td>DELETE</td>
<td>/api/pki/certificate/requests/&lt;ID-of-the-CSR&gt;</td>
<td>Deleting a CSR does not remove the corresponding X.509 identifier from the configuration, that is, the existing private key and certificate chain pair remains in use until you update the reference. Unreferenced X.509 identifier are removed automatically.</td>
</tr>
</tbody>
</table>

**Example: Generating a new CSR**

The following command creates a new CSR.
curl --cookie cookies https://<IP-address-of-SPS>/api/pki/certificate/requests

```json
{
    "subject": [
        {
            "name": "countryName", "value": "US"},
        {
            "name": "stateOrProvinceName", "value": "CA"},
        {
            "name": "streetAddress", "value": "Example Street"},
        {
            "name": "organizationName", "value": "Example Organization"},
        {
            "name": "commonName", "value": "first.example.com"},
        {
            "name": "emailAddress", "value": "info@example.com"
        }
    ],
    "extensions": [
        {
            "name": "basicConstraints", "value": "CA:FALSE", "critical": true},
        {
            "name": "keyUsage", "value": "digitalSignature,keyAgreement", "critical": true},
        {
            "name": "extendedKeyUsage", "value": "clientAuth", "critical": false},
        {
            "name": "subjectAltName", "value": "IP:123.123.123.123,DNS:example2.organization.com", "critical": false}
    ]
}
```

Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>subject.name</td>
<td>string</td>
<td>The subject name must be an object identifier (OID), or a name that can be translated to an OID. Example values are:</td>
<td>• countryName • stateOrProvinceName • streetAddress • organizationName • commonName • emailAddress</td>
</tr>
<tr>
<td>subject.value</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>extensions</td>
<td>object</td>
<td>The list of extensions. If you do not want to specify</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------</td>
<td>--------</td>
<td>------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>any extensions</td>
<td></td>
<td></td>
<td>any extensions in the request, use an empty list.</td>
</tr>
<tr>
<td>extensions.name</td>
<td>enum</td>
<td>The name of the extension.</td>
<td>Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- basicConstraints</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- keyUsage</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- extendedKeyUsage</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- subjectAltName</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- subjectInfoAccess</td>
</tr>
<tr>
<td>extensions.value</td>
<td>string</td>
<td>The value of the extension.</td>
<td></td>
</tr>
<tr>
<td>extensions.critical</td>
<td>boolean</td>
<td>Indicates whether the extension should be marked as critical in the request.</td>
<td></td>
</tr>
</tbody>
</table>

**Response**

The following is a sample response received when a new CSR is created.
For details of the meta object, see [Message format](#) on page 10.

```json
{
    "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "meta": {
        "href": "/api/pki/certificate/requests/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
        "parent": "/api/pki/certificate/requests"
    }
}
```

**Setting or updating a certificate chain for a CSR**

There are two ways to set or update a certificate chain for a CSR:

- Use a **PUT request**, if you know the CSR identifier.
- Use a **POST request**, if you do not know the CSR identifier. In this case the REST API automatically selects the CSR based on the information in the first certificate in the chain.
Example with PUT request: Replacing a web server certificate

To replace a web server certificate, you have to

1. Generate a new CSR. You can find a detailed description of the available parameters listed in the request message body table of **Example: Generating a new CSR**. The result will be the identifier of the new CSR.

2. Send a GET request to the https://<IP-address-of-SPS>/api/pki/certificate/requests/<ID-of-the-CSR> endpoint. Obtain the PEM value of the CSR.

3. Send the CSR to the trusted certificate authority (CA).

   The CA validates your request for using the stored certificate. If the validation is successful, it will respond with a signed X.509 certificate to be used by SPS. The chain might contain CA certificates in the hierarchy.

4. Open a transaction.

   For more information, see **Open a transaction** on page 29.

5. Send a PUT request to the https://<IP-address-of-SPS>/api/pki/certificate/requests/<ID-of-the-CSR> endpoint. The request must include:
   - the X.509 certificate chain
   - the identifier of the trust store that is used to validate the certificate chain, or null, if you want to disable validation

```json
{
    "certificate_chain": "-----BEGIN CERTIFICATE-----
MIID+zzCAEvOAwIBAgIBATANBgkqhkiG9w0BAQsFADAXMRUwEwYDVQQDDAxFe
GFt\ncGx1IENBIDIwHhcNMjAwODMxMTEyMjYyMzQyNDEwMDA5FeGtcGx1IEN
pdHkgMjef\nMB0GA1UECgweRXhhbXBsZSBPcmdhbml6YXRpb24gMjEf
MB0GA1UECwwMRXhhbXBsZSBPcmdhbml6YXRpb24gMjEvMSAaMB0GA1UEBhMCVVMx
CzAJBgNVBAgMMAkNBMRcwFQYDVQQHDAF0aGlyZC5leGFtcGxlLmNvbTATBGCg
wCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
gwCAqhkjOAPB0gDgcwgcqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBBggqhkjOPQIBB
-----END CERTIFICATE-----
```

SPS 6.13.0 REST API Reference Guide
Managing SPS

ONE IDENTITY
by Quest

276
uNjIbrfzMAdavYUv4dtFCi49gByjHshJbGyDPQ/nbr1Zzky8/20IvarmelEigp
8gn3XWqkJ0juQO6q1M6bnyjcr3RFXyNo3Ed8FI4S4/I
/n/H9ZCPFKQBXS560fGy6x
ftbrc3fij3/B/c0Ev8lzAc8trT1lVRDAbgVizn4/nYmporo17HjqkU7Oau
Bq99eDDUziczFVMU3V4U+IuCepoB1f7a1YRp/kS1p:Xp0+o
/n/Zh+9SA4IIF7cb
PwD4M5J5kmpCG9xHqj7UKnvcJ4f0v34fjp/GwJpr2Z5TkyQPQNYJFud/dn
8N4zNvK1kw2Hlg2bg06ARAT10s9kR0v3RKFrNZb9nXVYkedNeXFA
/n4i5yf9G
KNF9C0SYZbp5zaNZBZn9re5+PKo1iccBUKS0209j6D2ZJ7Fu3oq3fIabuVNYK
Zrala:jXFErDr0qS5/TxtJUmcmQCXITljps0ndyGN2317w/vImQNo6cTeoKF1li
/y+t+zC8nNfcJp6snYfUT1RZw2ros+84ARY0wzd8S0v8x9xu+CfotWR4aOqCd9s
/nqno6yMjNwUreti1+IbHim
/n-END CERTIFICATE-----

"trust_store" : "2222"
Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificate_chain</td>
<td>string</td>
<td>The certificate chain can be specified as a string of individual certificates separated by a newline, or as a list of strings containing the certificates. The certificates must be specified in PEM format.</td>
<td></td>
</tr>
<tr>
<td>trust_store</td>
<td>string</td>
<td>The identifier of the trust store that is used to validate the certificate chain, or null, if you want to disable validation.</td>
<td></td>
</tr>
</tbody>
</table>

The result will be the X.509 identifier referring to the private key + certificate chain pair.

| NOTE: The X.509 identifier can only be used in REST configuration. |

6. Use the X.509 identifier to replace your web server certificate. For more information, see Internal certificates on page 259.

7. **Commit your changes.**

For more information, see Commit a transaction on page 31.

---

**Example with POST request: Replacing a web server certificate**

*To replace a web server certificate without knowing the CSR identifier, you have to*

1. Generate a new CSR. You can find a detailed description of the available parameters listed in the request message body table of Example: Generating a new CSR. The result will be the identifier of the new CSR.

2. Send a GET request to the https://<IP-address-of-SPS>/api/pki/certificate/requests/<ID-of-the-CSR> endpoint. Obtain the PEM value of the CSR.

3. Send the CSR to the trusted certificate authority (CA).

   The CA validates your request for using the stored certificate. If the validation is successful, it will respond with a signed X.509 certificate chain. The first
element of this certificate chain must be the certificate to be used by SPS. The chain might contain CA certificates in the hierarchy.

4. Open a transaction.

For more information, see Open a transaction on page 29.

5. Send a POST request to the https://<IP-address-of-SPS>/api/pki/certificate endpoint. The request must include:
   - the X.509 certificate chain
   - the identifier of the trust store that is used to validate the certificate chain, or null, if you want to disable validation

```json
{
    "certificate_chain": "-----BEGIN CERTIFICATE-----
MIID+zCAoIGAIBAgIBDANBgqhkjigKG9w0BAQfADAxMRUwEwYDVQQDDAFAZEx
GFtncGxIENBIDWhhNjAwOQDyMTIyMDUwWhcNMzAxE4MTIyMDU2WjCBj
ELMAkGcEAIUEBMBVVMxO0EBjAgNBBGwAgMBAAE=
-----BEGIN CERTIFICATE-----
MIID+zCCAgIBAgIBDANBgqhkjigKG9w0BAQfADAxMRUwEwYDVQQDDAFAZEx
GFtncGxIENBIDWhhNjAwOQDyMTIyMDUwWhcNMzAxE4MTIyMDU2WjCBj
ELMAkGcEAIUEBMBVVMxO0EBjAgNBBGwAgMBAAE=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIID+zCCAgIBAgIBDANBgqhkjigKG9w0BAQfADAxMRUwEwYDVQQDDAFAZEx
GFtncGxIENBIDWhhNjAwOQDyMTIyMDUwWhcNMzAxE4MTIyMDU2WjCBj
ELMAkGcEAIUEBMBVVMxO0EBjAgNBBGwAgMBAAE=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIID+zCCAgIBAgIBDANBgqhkjigKG9w0BAQfADAxMRUwEwYDVQQDDAFAZEx
GFtncGxIENBIDWhhNjAwOQDyMTIyMDUwWhcNMzAxE4MTIyMDU2WjCBj
ELMAkGcEAIUEBMBVVMxO0EBjAgNBBGwAgMBAAE=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIID+zCCAgIBAgIBDANBgqhkjigKG9w0BAQfADAxMRUwEwYDVQQDDAFAZEx
GFtncGxIENBIDWhhNjAwOQDyMTIyMDUwWhcNMzAxE4MTIyMDU2WjCBj
ELMAkGcEAIUEBMBVVMxO0EBjAgNBBGwAgMBAAE=
-----END CERTIFICATE-----

SPS 6.13.0 REST API Reference Guide
Managing SPS

279
Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificate_chain</td>
<td>string</td>
<td>The certificate chain can be specified as a string of individual certificates separated by a newline, or as a list of strings containing the certificates. The certificates must be specified in PEM format.</td>
<td></td>
</tr>
<tr>
<td>trust_store</td>
<td>string</td>
<td>The identifier of the trust store that is used to validate the certificate chain, or null, if you want to disable validation.</td>
<td></td>
</tr>
</tbody>
</table>
The result will be the X.509 identifier referring to the private key + certificate chain pair.

| NOTE: The X.509 identifier can only be used in REST configuration.

6. Use the X.509 identifier to replace your web server certificate. For more information, see Internal certificates on page 259.

I. Commit your changes.

For more information, see Commit a transaction on page 31.

Example: Querying existing CSRs

The following is a sample response received when existing CSRs are queried.

For details of the meta object, see Message format on page 10.

```json
{
  "meta": {
    "href": "/api/pki/certificate/requests",
    "parent": "/api/pki/certificate"
  },
  "items": [
    {
      "meta": {
        "href": "/api/pki/certificate/requests/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"},
      "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
      "body": {
        "certificate_chain": null,
        "fingerprint": {
          "hash_algorithm": "sha256"
        },
        "subject": [
          {"name": "countryName", "value": "US"},
          {"name": "stateOrProvinceName", "value": "CA"},
          {"name": "streetAddress", "value": "Example Street"},
          {"name": "organizationName", "value": "Example Organization"},
          {"name": "commonName", "value": "first.example.com"},
          {"name": "emailAddress", "value": "info@example.com"}
        ]
    }
  ]
}
```
"extensions": [
  {
    "name": "basicConstraints",
    "value": "CA:FALSE",
    "critical": true,
  },
  {
    "name": "keyUsage",
    "value": "digitalSignature,keyAgreement",
    "critical": true,
  },
  {
    "name": "extendedKeyUsage",
    "value": "clientAuth",
    "critical": false,
  },
  {
    "name": "subjectAltName",
    "value": "IP:123.123.123.123,DNS:second.example.com",
    "critical": false
  }
],
"pem": "-----BEGIN CERTIFICATE REQUEST-----
MIICPzCCAgAwgCwAoGBgNVBAkMB04wMBcwEwYDVQQDDE5FeGFtcGxlIFZlZCBleGFtcGxlLm5ldCBhY2VydGl2ZSB0aGlzIHRvY3VtZW50IHRvY3VtZW50IGNvbXBvcnQgaXMgaHR0cDovL3d3dy5saWJlcy5jb20vdGlezIzLTMEQIjIwMjAyMi0xMjUzLTM5Ni0xMDExOFoiLOMZQ01UZ3JzUmVxdWVzdF9pbnN0YW5jZTJFbm90b20=
-----END CERTIFICATE REQUEST-----

"public_key": "-----BEGIN PUBLIC KEY-----
MIID+CzCCAgAwgCwAoGBgNVBAkMB04wMBcwEwYDVQQDDE5FeGFtcGxlIFZlZCBleGFtcGxlLm5ldCBhY2VydGl2ZSB0aGlzIHRvY3VtZW50IHRvY3VtZW50IGNvbXBvcnQgaXMgaHR0cDovL3d3dy5saWJlcy5jb20vdGlezIzLTMEQIjIwMjAyMi0xMjUzLTM5Ni0xMDExOFoiLOMZQ01UZ3JzUmVxdWVzdF9pbnN0YW5jZTJFbm90b20=
-----END PUBLIC KEY-----
"
"hash_algorithm": "sha256"
},
"subject": [
  {"name": "countryName", "value": "US"},
  {"name": "stateOrProvinceName", "value": "CA"},
  {"name": "streetAddress", "value": "Example Street 2"},
  {"name": "organizationName", "value": "Example Organization 2"},
  {"name": "commonName", "value": "third.example.com"},
  {"name": "emailAddress", "value": "info2@example.com"}
],
"extensions": [
  {"name": "basicConstraints", "value": "CA:FALSE", "critical": true},
  {"name": "keyUsage", "value": "digitalSignature,keyAgreement", "critical": true},
  {"name": "extendedKeyUsage", "value": "clientAuth", "critical": false},
  {"name": "subjectAltName", "value": "DNS:fourth.example.com", "critical": false}
],
"pem": "-----BEGIN CERTIFICATE REQUEST-----
MIICIzCCAkCAQAwgdQxCzAJBgNVBAYTA1VTMQwDAYDVQQHRDAsUm9vdHN0b3Jr
nMIIdwNMQ4wDAYDVQQLDBNMb3JtYXMMykGA1UdDgQgMCMGAgEwMIXA
nMIIDPwIBADANBgkqhkiG9w0BAQsFAAOCAQEAcJ870w14RvEuJbXX7
nMIIDPwIBADANBgkqhkiG9w0BAQsFAAOCAQEAaqV3sAlh5B5JvJ7V0
-----END CERTIFICATE REQUEST-----
","public_key": "-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZiZj0DAcQDgAEmAY3OymmcLjhZSOGAD8bQM
-----END PUBLIC KEY-----
"}
Example: Querying a single CSR

The following is a sample response received when a single CSR is queried.
For details of the meta object, see Message format on page 10.

```json
{
    "meta": {
        "href": "/api/pki/certificate/requests/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
        "parent": "/api/pki/certificate/requests"
    },
    "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "body": {
        "certificate_chain": null,
        "fingerprint": {
            "hash_algorithm": "sha256"
        },
        "subject": [
            {"name": "countryName", "value": "US"},
            {"name": "stateOrProvinceName", "value": "CA"},
            {"name": "streetAddress", "value": "Example Street"},
            {"name": "organizationName", "value": "Example Organization"},
            {"name": "commonName", "value": "first.example.com"},
            {"name": "emailAddress", "value": "info@example.com"}
        ],
        "extensions": [
            {"name": "basicConstraints", "value": "CA:FALSE", "critical": true},
            {"name": "keyUsage", "value": "digitalSignature,keyAgreement", "critical": true},
            {"name": "extendedKeyUsage", "value": "clientAuth", "critical": false},
            {"name": "subjectAltName", "value": "IP:123.123.123.123,DNS:second.example.com", "critical": false}
        ],
        "pem": "-----BEGIN CERTIFICATE REQUEST-----
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
MIICPzCCAcICAQAwgZgCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG
A1UECAwCQ0ExFTATBgNVBAcMDEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQK
DA5FeGFtcGxlIFVuaXQgMT
"}
```

SPS 6.13.0 REST API Reference Guide
Managing SPS

285
Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificate_chain</td>
<td>string</td>
<td>The certificate chain received from the trusted CA.</td>
<td></td>
</tr>
<tr>
<td>fingerprint</td>
<td>object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fingerprint.digest</td>
<td>string</td>
<td>The fingerprint of the certificate, for example ef:d3:8e:d0:81:4f:a2:8f:3b:8b:0c:dd:c7:8f:8c:7e.</td>
<td></td>
</tr>
<tr>
<td>fingerprint.hash_algorithm</td>
<td>string</td>
<td>The hash algorithm used to create the fingerprint, for example, sha256.</td>
<td></td>
</tr>
<tr>
<td>subject</td>
<td>object</td>
<td>The subject string of the certificate.</td>
<td></td>
</tr>
<tr>
<td>extensions</td>
<td>object</td>
<td>The list of extensions.</td>
<td></td>
</tr>
<tr>
<td>pem</td>
<td>string</td>
<td>The certificate signing request in PEM format.</td>
<td></td>
</tr>
<tr>
<td>public_key</td>
<td>string</td>
<td>The public key in PEM format.</td>
<td></td>
</tr>
</tbody>
</table>

**Example: Deleting a CSR**

The following is a sample response received when a CSR is deleted.  
For details of the meta object, see Message format on page 10.
HTTP response codes

HTTP response codes comprise of standard or endpoint-specific HTTP status and error codes. The following table lists the endpoint-specific HTTP response codes for this request.

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status/Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>SyntacticError</td>
<td>Syntax error: Could not load PEM certificate; error=&quot;[('PEM routines', 'get_name', 'no start line')]&quot;</td>
</tr>
<tr>
<td>400</td>
<td>CertChainValidationException</td>
<td>You have attempted to store a certificate chain, which could not be validated with the specified Trust Store.</td>
</tr>
<tr>
<td>400</td>
<td>OnlyOnCentralNode</td>
<td>Certificate signing requests can only be created or updated on the Central management node of the cluster.</td>
</tr>
<tr>
<td>404</td>
<td>NoMatchingCsrFound</td>
<td>You have attempted to store a certificate chain which belongs to a private key for which no certificate signing requests can be found. Make sure to only send certificates which belong to a private key for which a certificate signing request exists.</td>
</tr>
</tbody>
</table>

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.

Certificates stored on SPS

To create a new certificate, you have to POST the certificate and its private key as a JSON object to the https://<IP-address-of-SPS>/api/x509 endpoint. For details, see Create a
new object on page 45. The body of the POST request must contain a JSON object with the parameters listed in Element. The response to a successful POST message is a JSON object that includes the reference ID of the created certificate in its key attribute. You can reference this ID in other parts of the configuration. Note that you can use a certificate object for only one purpose, that is, you cannot reference one object twice.

**URL**

POST https://<IP-address-of-SPS>/api/configuration/x509

- Note that the GET method is not permitted on this endpoint, you cannot list the existing certificates. However, if you know the reference ID of a certificate, you can display its properties:

  GET https://<IP-address-of-SPS>/api/configuration/x509/<reference-ID-of-the-private-key;>

- You cannot directly delete or modify a certificate, the DELETE and PUT methods are not permitted on certificate objects. To update a certificate, create a new one, then update the object that uses the old certificate to reference the new certificate.

**Table 7: Headers**

<table>
<thead>
<tr>
<th>Header name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Specifies the type of the data sent. SPS uses the JSON format</td>
<td>Required</td>
<td>application/json</td>
</tr>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API.</td>
</tr>
</tbody>
</table>

**Sample request**

The following command creates a new certificate object. Note the following requirements:
The key must be in PKCS-1 PEM format.
You need the certificate and the private key as well.
Encrypted private keys are not supported.
The attributes of the POST message that contain the certificate and the private key must be a single line, enclosed in double-quotes.
Replace line-breaks in the PEM certificate with \n
The certificate and the certificate chain must be valid, SPS will reject invalid certificates and invalid certificate chains.

curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-SPS>/api/configuration/x509 --data '{"private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIeIAIBAAKCAQEpOYvZMhqe2oQMNfVQ5N1dVRE2SR0VY+ALnzPZ4fUoJy\n.....\nI2SchDibk/Xj/ZvuEq23lvzayW0VVuVHTh3JXX3SU4Sa0vpaeC+3oddVTwQOWRq0\n".

Qbn5W3xKz4vXDDQHEbEsVsDQ9A7+uCEuHp04s33IK9KEa0Zdp745AU0DSGNNH4Fzc\n-----END RSA PRIVATE KEY-----"

"certificate": "-----BEGIN CERTIFICATE-----
\nMIIeIAIBAAKCAQEpOYvZMhqe2oQMNfVQ5N1dVRE2SR0VY+ALnzPZ4fUoJy\n.....\nI2SchDibk/Xj/ZvuEq23lvzayW0VVuVHTh3JXX3SU4Sa0vpaeC+3oddVTwQOWRq0\n".

Qbn5W3xKz4vXDDQHEbEsVsDQ9A7+uCEuHp04s33IK9KEa0Zdp745AU0DSGNNH4Fzc\n-----END CERTIFICATE-----",

"private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIeIAIBAAKCAQEpOYvZMhqe2oQMNfVQ5N1dVRE2SR0VY+ALnzPZ4fUoJy\n.....\nI2SchDibk/Xj/ZvuEq23lvzayW0VVuVHTh3JXX3SU4Sa0vpaeC+3oddVTwQOWRq0\n".

Qbn5W3xKz4vXDDQHEbEsVsDQ9A7+uCEuHp04s33IK9KEa0Zdp745AU0DSGNNH4Fzc\n-----END RSA PRIVATE KEY-----",

"issuer_chain": []
}'

The body should be:

```json
{
    "certificate": "-----BEGIN CERTIFICATE-----
\nMIIeIAIBAAKCAQEpOYvZMhqe2oQMNfVQ5N1dVRE2SR0VY+ALnzPZ4fUoJy\n.....\nI2SchDibk/Xj/ZvuEq23lvzayW0VVuVHTh3JXX3SU4Sa0vpaeC+3oddVTwQOWRq0\n".

Qbn5W3xKz4vXDDQHEbEsVsDQ9A7+uCEuHp04s33IK9KEa0Zdp745AU0DSGNNH4Fzc\n-----END CERTIFICATE-----",

"private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIeIAIBAAKCAQEpOYvZMhqe2oQMNfVQ5N1dVRE2SR0VY+ALnzPZ4fUoJy\n.....\nI2SchDibk/Xj/ZvuEq23lvzayW0VVuVHTh3JXX3SU4Sa0vpaeC+3oddVTwQOWRq0\n".

Qbn5W3xKz4vXDDQHEbEsVsDQ9A7+uCEuHp04s33IK9KEa0Zdp745AU0DSGNNH4Fzc\n-----END RSA PRIVATE KEY-----",

"issuer_chain": []
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificate</td>
<td>The certificate in PKCS-1 PEM format (replace line-breaks with \n). For example:</td>
</tr>
<tr>
<td>private key</td>
<td>The private key of the certificate, without encryption or password protection (replace line-breaks with \n). For example:</td>
</tr>
</tbody>
</table>
Querying a specific key returns the following information about the key:

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/x509/<reference-ID-of-the-private-key>;
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fingerprint</td>
<td>string</td>
<td>The fingerprint of the certificate.</td>
</tr>
<tr>
<td>digest</td>
<td>string</td>
<td>The fingerprint of the certificate, for example ef:d3:8e:d0:81:4f:a2:8f:3b:8b:0c:dd:c7:8f:8c:7e</td>
</tr>
<tr>
<td>hash_algorithm</td>
<td>string</td>
<td>The hash algorithm used to create the fingerprint, for example, sha256.</td>
</tr>
<tr>
<td>subject</td>
<td>string</td>
<td>The subject string of the certificate.</td>
</tr>
</tbody>
</table>

Response

The response to a successful POST message is a JSON object that includes the reference ID of the created certificate in its key attribute.

For details of the meta object, see Message format on page 10.

```
{
    "key": "faa96916-c85e-46ff-8697-f4c5e596e7f",
    "meta": {
```
The response to querying a specific certificate is a JSON object that includes the parameters of the certificate, for example:

```json
{
    "body": {
        "fingerprint": {
            "digest": "ef:d3:8e:d0:81:4f:a2:8f:3b:8b:0c:dd:c7:8f:8c:7e",
            "hash_algorithm": "md5"
        },
        "subject": "C=RO/ST=State/L=Locality/O=Organization/OU=OrganizationalUnit/CN=example.com/emailAddress=root@example.com",
        "key": "6c4d1116-d79d-475b-bb37-9f844f085c14",
        "meta": {
            "first": "/api/configuration/x509/e5d13d18-07c5-43fa-89f4-c3d2ece17c71",
            "href": "/api/configuration/x509/6c4d1116-d79d-475b-bb37-9f844f085c14",
            "last": "/api/configuration/x509/6c4d1116-d79d-475b-bb37-9f844f085c14",
            "next": null,
            "parent": "/api/configuration/x509",
            "previous": "/api/configuration/x509/e5d13d18-07c5-43fa-89f4-c3d2ece17c71",
            "transaction": "/api/transaction"
        }
    }
}
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>SyntacticError</td>
<td>Syntax error: Could not load PEM key: Unsupported private key format, only PKCS-1 is supported. Encrypted private keys are not supported.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------</td>
<td>----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
<tr>
<td>405</td>
<td>MethodNotAllowed</td>
<td>The method &lt;method&gt; is not allowed for this node.</td>
</tr>
</tbody>
</table>

**Modify or delete certificate**

You cannot directly delete or modify a certificate, the DELETE and PUT methods are not permitted on certificate objects. To update a certificate, create a new one, then update the object that uses the old certificate to reference the new certificate. After you commit the transaction, SPS will automatically delete the old certificate.

**Local services: enabling SSH access to the SPS host**

Exclusively for troubleshooting purposes, you can access the SPS host using SSH. Completing the Welcome Wizard automatically disables SSH access to SPS. Re-enabling it allows you to connect remotely to the SPS host and login using the root user. The password of the root user is the one you provided in the Welcome Wizard.

⚠️ **CAUTION:**

Accessing the One Identity Safeguard for Privileged Sessions (SPS) host directly using SSH is not recommended or supported, except for troubleshooting purposes. In such case, the One Identity Support Team will give you exact instructions on what to do to solve the problem.

For security reasons, disable SSH access to SPS when it is not needed. For details, see "Enabling SSH access to the One Identity Safeguard for Privileged Sessions (SPS) host" in the Administration Guide.

The following encryption algorithms are configured on the local SSH service of SPS:

- Key exchange (KEX) algorithms:

  ```
  diffie-hellman-group-exchange-sha256
  ```
• Ciphers:
  aes256-ctr, aes128-ctr

• Message authentication codes:
  hmac-sha2-512, hmac-sha2-256

URL
GET https://<IP-address-of-SPS>/api/configuration/local_services/ssh

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Sample request

The following command lists the configuration options.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/local_services/ssh

Response

The following is a sample response received when listing the configuration options. For details of the meta object, see **Message format** on page 10.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration options of the SSH server.</td>
</tr>
<tr>
<td>access_restriction</td>
<td>JSON object</td>
<td>Enables and configures limitations on the clients that can access the web interface, based on the IP address of the clients.</td>
</tr>
<tr>
<td>allowed_from</td>
<td>list</td>
<td>The list of IP networks from where the administrators are permitted to access this management interface. To specify the IP addresses or networks, use the IPv4-Address/prefix format, for example, 10.40.0.0/16.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set it to true to restrict access to the specified client addresses.</td>
</tr>
<tr>
<td>allow_password_auth</td>
<td>boolean</td>
<td>Enables password-based authentication, so administrators can remotely login to SPS. If this option is set to False, SPS ignores every other option of this endpoint.</td>
</tr>
<tr>
<td>bruteforce_protection</td>
<td>boolean</td>
<td>Enables protection against brute-force attacks by denying access after failed login attempts for increasingly longer period. Enabled by default.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Enables the SSH server, so administrators can remotely login to SPS. If this option is set to False, SPS ignores every other option of this endpoint.</td>
</tr>
<tr>
<td>listen</td>
<td>list</td>
<td>Selects the network interface, IP address, and port where the clients can access the web interface.</td>
</tr>
<tr>
<td>address</td>
<td>JSON object</td>
<td>A reference to a configured network interface and IP address where this local service accepts connections. For example, if querying the interface /api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/ returns the following response:</td>
</tr>
</tbody>
</table>

```json
{
  "body": {
```
Then the listening address of the local service is the following.

```
nic1.interfaces.ff7574025754b3df1647001.addresses.1
```

This is the format you have to use when configuring the address of the local service using REST:

```
"address": "nic1.interfaces.ff7574025754b3df1647001.addresses.1"
```
When querying a local services endpoint, the response will contain a reference to the IP address of the interface in the following format:

```
"address": {
  "key": "nic1.interfaces.ff754025754b3df1647001.addresses.1",
  "meta": {
    "href": "/api-configuration/network/communications/nic1#interfaces/ff7574025754b3df1647001/addresses/1"
  }
},
```

The port number where this local service accepts connections.

**TIP:** One Identity recommends using 2048-bit RSA keys (or stronger).
### Elements of public keys

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>comment</td>
<td>string Comments of the public key.</td>
</tr>
<tr>
<td>key</td>
<td>JSON object Contains the type of the key and the key itself. For example:</td>
</tr>
<tr>
<td></td>
<td>&quot;key&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;selection&quot;: &quot;rsa&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;ASFDAB3Nz-aC1yc2EAAAABIwAAASdfASF/EuQh9zc2umxX...dU=&quot;</td>
</tr>
<tr>
<td>selection</td>
<td>rsa The type of the public key. Must be rsa.</td>
</tr>
<tr>
<td>value</td>
<td>string The public key itself.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Manage the SPS license

You can display information about the currently used SPS license from the https://<IP-address-of-SPS>/api/configuration/management/license endpoint.
CAUTION:

Accessing the One Identity Safeguard for Privileged Sessions (SPS) host directly using SSH is not recommended or supported, except for troubleshooting purposes. In such case, the One Identity Support Team will give you exact instructions on what to do to solve the problem.

For security reasons, disable SSH access to SPS when it is not needed. For details, see "Enabling SSH access to the One Identity Safeguard for Privileged Sessions (SPS) host" in the Administration Guide.

The following encryption algorithms are configured on the local SSH service of SPS:

- Key exchange (KEX) algorithms:
  
  ```diffie-hellman-group-exchange-sha256```

- Ciphers:
  
  ```aes256-ctr,aes128-ctr```

- Message authentication codes:
  
  ```hmac-sha2-512,hmac-sha2-256```

URL

GET `https://<IP-address-of-SPS>/api/configuration/management/license`

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, <code>a1f71d030e657634730b9e887cb59a5e56162860</code>. For details on authentication, see <a href="#">Authenticate to the SPS REST API on page 19</a>.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists the configuration options.

```
curl --cookie cookies -H "Content-Type: application/json"
https://10.30.255.28/api/configuration/management/license
```

Response

The following is a sample response received.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "customer": "Example",
        "limit": 5000,
        "limit_type": "host",
        "serial": "b937d212-db7d-0f2f-4c87-295e3c57024a",
        "valid_not_after": "2018-11-07",
        "valid_not_before": "2017-11-06"
    },
    "key": "license",
    "meta": {
        "first": "/api/configuration/management/certificates",
        "href": "/api/configuration/management/license",
        "last": "/api/configuration/management/webinterface",
        "next": "/api/configuration/management/root_password",
        "parent": "/api/configuration/management",
        "previous": "/api/configuration/management/health_monitoring",
        "remaining_seconds": 600,
        "transaction": "/api/transaction",
        "upload": "/api/upload/license"
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the parameters of the license.</td>
</tr>
<tr>
<td>customer</td>
<td>string</td>
<td>The company permitted to use the license (for example, Example Ltd.).</td>
</tr>
<tr>
<td>limit</td>
<td>integer</td>
<td>The actual value of the session or host limit (see limit_type).</td>
</tr>
</tbody>
</table>
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Upload a new license

To upload a new license file, complete the following steps.
1. Download your license file from support portal.

2. **Open a transaction.**
   For more information, see Open a transaction on page 29.

3. **Upload the license file.**
   Upload the file to the https://<IP-address-of-SPS>/api/upload/license endpoint. For example:
   
   ```
   curl --cookie cookies -F 'data=@/path/license.txt' -H "Expect:" --insecure https://<IP-address-of-SPS>/api/upload/license
   ```

4. **Restart the traffic on SPS.**
   SPS will not use the new license to ongoing sessions. For the new license to take full effect, you must restart all traffic on the Basic Settings > System > Traffic control page of the SPS web interface.
   
   ```
   curl --cookie cookies -F 'data=@/path/license.txt' -H "Expect:" --insecure https://<IP-address-of-SPS>/api/upload/license
   ```

5. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

### Change contact information

The About page on the SPS web interface and the /api/info endpoint contains various contact information. You can change this to a custom email address or URL.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/management/support_info

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the...</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication</td>
</tr>
<tr>
<td><strong>Cookie name</strong></td>
<td><strong>Description</strong></td>
<td><strong>Required</strong></td>
<td><strong>Values</strong></td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
<td>--------------</td>
<td>------------</td>
</tr>
<tr>
<td>token</td>
<td>token of the user</td>
<td>response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the contact information.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/support_info
```

**Response**

The following is a sample response received when querying the endpoint.

For details of the meta object, see Message format on page 10.

```
{
  "body": {
    "uri": null
  },
  "key": "support_info",
  "meta": {
    "first": "/api/configuration/management/certificates",
    "href": "/api/configuration/management/support_info",
    "last": "/api/configuration/management/webinterface",
    "next": "/api/configuration/management/syslog",
    "parent": "/api/configuration/management",
    "previous": "/api/configuration/management/splunk_forwarder",
    "remaining_seconds": 600,
    "transaction": "/api/transaction"
  }
}
```

**Change the support link**

To change the support link, complete the following steps.
1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **PUT a JSON object containing the new support link.**
   PUT a JSON object containing the new support link to the `https://<IP-address-of-SPS>/api/configuration/management/support_info` endpoint. For example:

   ```bash
   curl -X PUT -d '{"uri": { "selection": "mailto", "value": "mailto:support@example.com" } }' -H "Content-Type: application/json" --cookie cookies "https://<IP-address-of-SPS>/api/configuration/management/support_info"
   ```
   To use an HTTP or HTTPS link as contact info, use the following JSON object:

   ```json
   {
     "uri": {
       "selection": "url",
       "value": "http://example.com"
     }
   }
   ```
   To use a email address as contact info, use the following JSON object:

   ```json
   {
     "uri": {
       "selection": "mailto",
       "value": "mailto:support@example.com"
     }
   }
   ```

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

**Splunk integration**

SPS can forward session data to Splunk near real-time. Using the One Identity Safeguard for Privileged Sessions App for Splunk you can integrate this data with your other sources, and access all your data related to privileged user activities from a single interface. To configure SPS to forward session data to Splunk, complete the following steps.
Prerequisites and restrictions:

- SPS version 5 F5 or later
- Splunk version 6.5 or later
- SPS does not send historical data to Splunk, only data from the sessions started after you complete this procedure.

URL

GET https://<IP-address-of-SPS>/api/configuration/management/splunk_forwarder

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the endpoints for SNMP configuration settings.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/splunk_forwarder
```

Response

The following is a sample response received when querying the endpoint.

For details of the meta object, see Message format on page 10.
{  
    "body": {  
      "enabled": true,  
      "flush_interval": 600,  
      "host": {  
        "selection": "fqdn",  
        "value": "splunk.example.com"  
      },  
      "pam_address": {  
        "selection": "fqdn",  
        "value": "scb.example.com"  
      },  
      "port": 8088,  
      "ssl": {  
        "selection": "insecure"  
      },  
      "token": "2134356431"  
    }  
  }

Elements of remote_desktop_gateway

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>JSON object Top-level element</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean Set to true and configure the other options as needed for your environment to forward session data from SPS to Splunk.</td>
</tr>
<tr>
<td>flush_interval</td>
<td>integer [seconds] If the Splunk server becomes unaccessible, SPS will try to resend the data when this period expires.</td>
</tr>
<tr>
<td>host</td>
<td>JSON object Contains the hostname or the IPv4 address of the Splunk server.</td>
</tr>
</tbody>
</table>

```
"host": {  
  "selection": "fqdn",  
  "value": "splunk.example.com"  
},

"host": {  
  "selection": "ip",  
  "value": "192.168.1.1"  
},
```

**selection** fqdn | ip Defines the address type (IP or domain name). Possible values are:

- **fqdn**: The server address is provided as a fully qualified domain name.
- **ip**: The server address is provided as an IPv4 address.
## Elements of remote_desktop_gateway

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>string</td>
<td>The address of the server.</td>
</tr>
<tr>
<td>port</td>
<td>integer</td>
<td>The port number where your Splunk HTTP Event Collector is accepting connections. By default, Splunk uses port 8088.</td>
</tr>
<tr>
<td>ssl</td>
<td>JSON object</td>
<td>Determines if encryption is used between SPS and Splunk.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Determines if encryption is used between SPS and Splunk. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- disabled: Use this option if your Splunk HTTP Event Collector accepts only unencrypted HTTP connections. Since the data forwarded to Splunk contains sensitive information, One Identity recommends to use HTTPS encryption between SPS and Splunk.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ssl&quot;: { &quot;selection&quot;: &quot;disabled&quot; },</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- insecure: Use HTTPS encryption between SPS and Splunk.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ssl&quot;: { &quot;selection&quot;: &quot;insecure&quot; },</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- secure: Use HTTPS encryption between SPS and Splunk and also verify the identity of the Splunk server. If you use this option, you must include the certificate of the Splunk server, or the certificate of the CA that issued the certificate of the Splunk server in the certificate option.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ssl&quot;: { &quot;selection&quot;: &quot;secure&quot; },</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>token</td>
<td>string</td>
<td>The HTTP Event Collector authentication token you have generated for SPS.</td>
</tr>
</tbody>
</table>
Configure Splunk forwarder

1. Install the One Identity Safeguard for Privileged Sessions App for Splunk to your Splunk installation. This will automatically enable and configure the HTTP Event Collector (HEC) in your Splunk installation, and create an HTTP Event Collector authentication token ("HEC token") that SPS will use.

   To help identify the source of the received data, the following settings are configured automatically in the One Identity Safeguard for Privileged Sessions App for Splunk:
   - **index**: The One Identity Safeguard for Privileged Sessions App for Splunk creates the index automatically, with the name balabit_events.
   - **sourcetype**: The source type of the events the SPS forwards is balabit:event.

2. On your Splunk interface, navigate to Settings > Data inputs > HTTP Event Collector. Copy the **Token Value** from the Balabit_HEC field. This is the HTTP Event Collector authentication token and you will need it when configuring SPS.

3. **Create the JSON object that configures SPS to forward session data to Splunk.**

   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/management/splunk_forwarder` endpoint. You can find a detailed description of the available parameters listed in Elements of remote_desktop_gateway. For example,

   ```json
   {
     "enabled": true,
     "flush_interval": 600,
     "host": {
       "selection": "fqdn", "value": "splunk.example.com"
     },
     "pam_address": {
       "selection": "fqdn", "value": "psm.example.com"
     },
     "port": 8088,
     "ssl": {
       "selection": "insecure"
     },
     "token": "2134356431"
   }
   ```

4. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

5. Splunk will display the data received from SPS as it was received from the host set in the `pam_address` field. By default, this is the hostname and domain name of the SPS appliance as set on the `/api/configuration/network/naming` endpoint. Adjust this field as needed for your environment.

6. Start a session that SPS will audit to test your configuration, and verify that the data of the session appears in Splunk.
**Splunk integration**

The universal SIEM forwarder can automatically send data about the audited sessions to Splunk, ArcSight, or other third-party systems. The messages are standard syslog messages in RFC3164 format (also called legacy-syslog or BSD-syslog format). The body of the syslog message (the MESSAGE part) can be formatted as JavaScript Object Notation (JSON), Common Event Format (CEF), or JSON-CIM format. For information about the details of the messages that the universal SIEM forwarder sends to the external SIEM network elements, see "Message format forwarded to SIEMs" in the Administration Guide.

One of the main advantages of the universal SIEM forwarder is that it has a lower impact on network and performance.

Each message contains the minimal information relevant to the event. Use the built-in correlation feature of the SIEM to combine events by session ID and view all information in one place.

**Prerequisites and restrictions**

- SPS version 5 F9 or later
- Splunk version 6.5 or later
- The CEF format is supported on all currently supported versions of ArcSight ESM, IBM QRadar and Microsoft Azure Sentinel.
- SPS does not send historical data, only data from the sessions started after you complete this procedure.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/management/universal_siem_forwarder

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format)
Sample request

The following command lists the endpoints for SNMP configuration settings.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/management/universal_siem_forwarder

Response

The following is a sample response received when querying the endpoint.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "enabled": true,
        "prefix": "myprefix",
        "targets": [
            {
                "format": "json_cim",
                "name": "siem_target",
                "protocol": {
                    "selection": "syslog",
                    "value": {
                        "address": {
                            "selection": "ip",
                            "value": "192.168.1.1"
                        },
                        "port": 5555,
                        "tls": {
                            "selection": "disabled"
                        }
                    }
                }
            }
        ]
    }
}
```

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>JSON object</td>
<td>Top-level element</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true and configure the other options as needed for your environment to forward session data from SPS to an external SIEM device.</td>
</tr>
<tr>
<td>prefix</td>
<td>string</td>
<td>A prefix to make the data more readable.</td>
</tr>
</tbody>
</table>
### Elements

<table>
<thead>
<tr>
<th>Elements of protocol</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
<td>Must be syslog</td>
</tr>
<tr>
<td>value</td>
<td>JSON object</td>
<td>Contains the address of the SIEM and the TLS settings of the connection.</td>
</tr>
</tbody>
</table>

### Type

The prefix is added to each JSON key. For example, if you use `sps_` as a prefix, in the forwarded JSON message the {"protocol": "ssh"} key changes to {"sps_protocol": "ssh"}, which allows you to identify the forwarded data more easily.

Other formats ignore the Prefix option.

### Description

<table>
<thead>
<tr>
<th>targets</th>
<th>JSON object</th>
<th>Specifies the details of the target SIEM device.</th>
</tr>
</thead>
<tbody>
<tr>
<td>format</td>
<td>cef</td>
<td>json</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the SIEM forwarder policy.</td>
</tr>
<tr>
<td>protocol</td>
<td>JSON object</td>
<td>Specifies connection details to the target SIEM device. For example:</td>
</tr>
</tbody>
</table>

```json
"protocol": {
  "selection": "syslog",
  "value": {
    "address": {
      "selection": "ip",
      "value": "192.168.1.1"
    },
    "port": 5555,
    "tls": {
      "selection": "secure",
      "trusted_ca_list_ref": "1241814345d074efd1ded7"
    }
  }
}
```
<table>
<thead>
<tr>
<th><strong>Elements of protocol</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>JSON object</td>
<td>Contains the type and the value of the address. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;address&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;ip&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;value&quot;: &quot;192.168.1.1&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;address&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;fqdn&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;value&quot;: &quot;my-siem.example.com&quot;</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the address type (IP or domain name). Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fqdn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The server address is provided as a fully qualified domain name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ip</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The server address is provided as an IP address.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The address of the server, corresponding to the format set in the selection field.</td>
</tr>
<tr>
<td>port</td>
<td>integer</td>
<td>The port number of the server.</td>
</tr>
<tr>
<td>tls</td>
<td>JSON object</td>
<td>The security settings of the</td>
</tr>
<tr>
<td>Elements of protocol</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>connection. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>tls</code>: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;secure&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;trusted_ca_list_ref&quot;: &quot;1241814345d074ef-d1ded7&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>`tls&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;disabled&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>selection</th>
<th>disabled</th>
<th>insecure</th>
<th>secure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>disabled: Use an unencrypted connection. Since the data forwarded contains sensitive information, One Identity recommends to use TLS encryption between SPS and your SIEM.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>insecure: Use TLS encryption, but do not validate the certificate of the SIEM.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>secure: Use TLS encryption and validate the certificate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elements of protocol</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Description of the SIEM. If you use this option, you must also set the trusted_ca_list_ref field.</td>
<td></td>
</tr>
</tbody>
</table>

- **trusted_ca_list_ref**: string
  - The key of the trusted CA list used to validate the certificate of the SIEM. This option is required if you set "selection": "secure". For details on creating trusted CA lists, see Trusted Certificate Authorities.

**Configure universal SIEM forwarder**

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. If you want to send the messages in an encrypted connection to the SIEM and also validate the certificate of the SIEM, upload the certificate of the CA that signed the certificate of the SIEM to a trusted CA list. For details on creating trusted CA lists, see Trusted Certificate Authorities.

3. **Create the JSON object that configures SPS to forward session data to your SIEM.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/management/universal_siem_forwarder endpoint. You can find a detailed description of the available parameters listed in Splunk integration. For example,

   ```json
   {
     "enabled": true,
     "prefix": "myprefix",
     "targets": [
       
   ```
4. Commit your changes.

For more information, see Commit a transaction on page 31.

Manage SPS clusters

When you have a set of two or more One Identity Safeguard for Privileged Sessions (SPS) instances in your deployment, you can join them into a cluster. This has several advantages. You can:

- Manage the nodes from one central location.
- Monitor their status and update their configuration centrally.
- Search all session data recorded by all nodes in the cluster on a single node.
- Scale the performance of the cluster by adding new nodes and joining them to the cluster easily.
- Extend auditing to other networks by adding new nodes to the cluster and joining them to the cluster.

This is achieved by assigning roles to the individual nodes in your cluster: you can set one of your SPS nodes to be the Central management node and the rest of the nodes are managed from this central node.

| NOTE: All nodes in a cluster must run the same version of SPS. |

| NOTE: To configure the /api/cluster/ endpoint, your usergroup must have "read and write/perform" privileges assigned to the Basic Settings > Cluster management object. |
You can configure this on the **Users & Access Control > Appliance Access** page of SPS’s web interface.
For details, see "Managing user rights and usergroups" in the Administration Guide.

**URL**

GET https://<IP-address-of-any-node-in-cluster>/api/cluster

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the endpoints available under the **cluster** endpoint.

```shell
curl --cookie cookies https://<IP-address-of-any-node-in-cluster>/api/cluster
```

**Response**

The following is a sample response received.
For details of the meta object, see [Message format](#) on page 10.

```json
{
  "items": [
    {
      "key": "configuration_sync",
      "meta": {
        "href": "/api/cluster/configuration_sync"
      }
    },
  ]
}
```
```json
{
    "key": "join_request",
    "meta": {
        "href": "/api/cluster/join_request"
    }
},
{
    "key": "nodes",
    "meta": {
        "href": "/api/cluster/nodes"
    }
},
{
    "key": "promote",
    "meta": {
        "href": "/api/cluster/promote"
    }
},
{
    "key": "status",
    "meta": {
        "href": "/api/cluster/status"
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Top-level element (list of JSON objects)</td>
<td>List of endpoints (objects) available from the current endpoint.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>The ID of the endpoint.</td>
</tr>
<tr>
<td>meta</td>
<td>Top-level item (JSON object)</td>
<td>Contains the path to the endpoint.</td>
</tr>
<tr>
<td>href</td>
<td>string (relative path)</td>
<td>The path of the resource that returned the response.</td>
</tr>
</tbody>
</table>
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Promote a SPS node to be the Central Management node in a new cluster

You can build a cluster by promoting a SPS node to the role of the Central Management node, and then join other nodes to your cluster.

To promote a node to be the Central Management node, complete the following steps:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the Central Management node.**
   
   POST an empty request to the `https://<IP-address-of-node-to-become-Central-Management-node>/api/cluster/promote` endpoint.
   
   The following is a sample response received.
   
   For details of the `meta` object, see Message format on page 10.
3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

### Join node(s) to the cluster

Once you have a Central Management SPS node in place, then you can join other nodes to your cluster.

To join nodes to your cluster, complete the following steps for each node that you want to join to the cluster:

1. **Open a transaction.**

   For more information, see [Open a transaction](#) on page 29.

2. **Create a join request.**

   POST the IP address of the Central Management node as a JSON object to the https://<IP-address-of-node-to-join-to-cluster>/api/cluster/join_request endpoint. The body of the POST request should be the following:
For example:

curl -X POST -H "Content-Type: application/json" --cookie cookies 
https://<IP-address-of-node-to-join-to-cluster>/api/cluster/join_request 
-data '{"central_management_address": "<IP-address-of-Central-Management-
node>"}'}

The following is a sample response received.

For details of the meta object, see Message format on page 10.

By default, no role is assigned to a non-management node, that is why the "roles" array is empty.

```
{
    "body": {
        "address": "<IP-address-of-node-joined-to-cluster>",
        "node_id": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
        "psk": "Ler7HZDFmZCxnLLgHNRfZYf0Rh1Z59919vEvR5UKtJEr1d4WeaHcBmQJLs4VDWi"
    },
    "meta": {
        "href": "/api/cluster/join_request",
        "parent": "/api/cluster",
        "remaining_seconds": 600
    }
}
```

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>Top-level element (JSON object)</td>
<td>Contains the JSON object of the node.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>The IP address of the node.</td>
</tr>
<tr>
<td>node_id</td>
<td>string</td>
<td>A reference ID for the node.</td>
</tr>
<tr>
<td>psk</td>
<td>string</td>
<td>The pre-shared key of the node used for authentication.</td>
</tr>
<tr>
<td>roles</td>
<td>string</td>
<td>The role of the node.</td>
</tr>
</tbody>
</table>
3. **Join the node to the cluster.**

POST the "body" object of the response to the https://<IP-address-of-Central-Management-node>/api/cluster/nodes endpoint as a JSON object. The body of the POST request should be the following:

```json
{
   "address": "<IP-address-of-node-joined-to-cluster>",
   "node_id": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
   "psk": "Ler7HZDFmZCxnLlGHNRFZyf0Rh1ZS9919vEvR5UKtJeb1d4WeahcBmQJJs4VDWiN",
   "roles": []
}
```

For example:

```bash
POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-Central-Management-node>/api/cluster/nodes --data '{"address": "<IP-address-of-node-joined-to-cluster>", "node_id": "46f97a58-4028-467d-9a22-9cfe78ae3e1c", "psk": "Ler7HZDFmZCxnLlGHNRFZyf0Rh1ZS9919vEvR5UKtJeb1d4WeahcBmQJJs4VDWiN", "roles": []}'
```

If the POST request is successful, the response includes:

```json
{
   "body": {
      "address": "<IP-address-of-node-joined-to-cluster>",
      "roles": []
   },
   "key": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
   "meta": {
      "href": "/api/cluster/nodes/46f97a58-4028-467d-9a22-9cfe78ae3e1c",
      "parent": "/api/cluster/nodes",
      "remaining_seconds": 28800
   }
}
```

4. **Commit your changes on both the Central Management node and the node you joined to the cluster.**

For details, see Commit a transaction on page 31.

### Query join status

To find out whether a node has been joined to a cluster, complete the following steps.
1. Query the /api/cluster/join_request endpoint on the node whose join status you want to figure out.

```
curl GET --cookie cookies https://<IP-address-of-node-to-be-queried>/api/cluster/join_request
```

The following is a sample response received.

For details of the meta object, see Message format on page 10.

```
"details": {
    "central_management_address": "<IP-address-of-Central-Management-node>",
},
"meta": {
    "href": "/api/cluster/join_request",
    "parent": "/api/cluster",
    "remaining_seconds": 600
},
"status": "in cluster"
```

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>details</td>
<td>Top-level element</td>
<td>Contains the IP address of the Central Management node of the cluster.</td>
</tr>
<tr>
<td>central_management_address</td>
<td>string</td>
<td>The IP address of the Central Management node.</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
<td>Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* not configured: Displayed when no cluster has been set up yet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* in progress: Displayed when the join action is in progress.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* in cluster: Displayed when the node is already in the cluster.</td>
</tr>
</tbody>
</table>
Assign a role to a node

By default, nodes do not have any roles assigned to them. The only exception is the Central management node, which you specifically promoted to fulfill that role. To assign a role to a node in the cluster, complete the following steps.

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Update the JSON object of the node.**
   
   PUT the role you want to assign to the node and the node's IP address as a JSON object to the https://<IP-address-of-Central-Management-node>/api/cluster/nodes/<node-id-of-node-to-be-updated> endpoint.

   You can assign the following roles to a node:

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>managed-host</td>
<td>There can be several nodes with this role. Nodes with the Managed Host role synchronize their entire configuration from the Central Management node, not only those elements of the configuration that are related to the cluster.</td>
</tr>
<tr>
<td>search-master</td>
<td>There can be only one node with this role. The Search Master node is the one node in the cluster on which you can search all the session data recorded by other nodes in the cluster, provided that the other nodes have been assigned the Search Minion role.</td>
</tr>
<tr>
<td>search-minion</td>
<td>There can be several nodes with this role. Nodes with the Search Minion role send session data that they recorded to the Search Master for central search purposes. The session data recorded by a Search Minion node is not searchable on the node itself, only on the Search Master.</td>
</tr>
<tr>
<td>search-local</td>
<td>There can be several nodes with this role. Nodes with the Search Local role keep the session data that they recorded for local searching. The session data recorded by a Search Local node is searchable on the node itself, but not on the Search Master. This is the only backward-compatible search role.</td>
</tr>
</tbody>
</table>
For further details on roles, see "Cluster roles" in the Administration Guide.

The body of the PUT request should be the following:

```
{
    "roles": ["<role-to-assign>"],
    "address": "<IP-address-of-node-to-be-updated>"
}
```

For example:

```
curl -H "Content-Type: application/json" --cookie cookies -X PUT https://<IP-address-of-Central-Management-node>/api/cluster/nodes/46f97a58-4028-467d-9a22-9cfe78ae3e1c --data '{"roles": ["managed-host"], "address": "<IP-address-of-node-to-be-updated>"}'
```

3. **Commit your changes.**

For more information, see Commit a transaction on page 31.

### Query nodes

To list the nodes available in a cluster, complete the following steps.

1. **Query the /api/cluster/nodes endpoint on the Central Management node.**

   ```
curl --cookie cookies https://<IP-address-of-Central-Management-node>/api/cluster/nodes
   ```

   The following is a sample response received.

   For details of the meta object, see Message format on page 10.

   ```json
   {
       "items": [
           {
               "key": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
               "meta": {
                   "href": "https://<IP-address-of-Central-Management-node>/api/cluster/nodes/46f97a58-4028-467d-9a22-9cfe78ae3e1c",
                   "status": "https://<IP-address-of-Central-Management-node>/api/cluster/status/46f97a58-4028-467d-9a22-9cfe78ae3e1c"
               }
           }
       ]
   }
   ```
"key": "b35c54da-b556-4f91-ade5-d26283d68277",
"meta": {
  "href": "/api/cluster/nodes/b35c54da-b556-4f91-ade5-d26283d68277",
  "status": "/api/cluster/status/b35c54da-b556-4f91-ade5-d26283d68277"
}
],
"meta": {
  "href": "/api/cluster/nodes",
  "parent": "/api/cluster",
  "remaining_seconds": 28800,
  "self": "/api/cluster/nodes/b35c54da-b556-4f91-ade5-d26283d68277",
  "status": "/api/cluster/status"
}

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Top-level element</td>
<td>List of endpoints (objects) available from the current endpoint.</td>
</tr>
<tr>
<td></td>
<td>(list of JSON objects)</td>
<td></td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>The ID of the node.</td>
</tr>
<tr>
<td>meta</td>
<td>Top-level item (JSON</td>
<td>Contains links to different parts of the REST service.</td>
</tr>
<tr>
<td></td>
<td>object)</td>
<td></td>
</tr>
<tr>
<td>href</td>
<td>string (relative path)</td>
<td>The path of the node that returned the response.</td>
</tr>
<tr>
<td>status</td>
<td>string (relative path)</td>
<td>The path to the status of the node that returned the response.</td>
</tr>
</tbody>
</table>

**Query one particular node**

To query one particular node, complete the following steps.

1. **Query the /api/cluster/nodes/<node-id-of-node-to-be-queried> endpoint on the node that you want to query.**

   ```
curl --cookie cookies https://<IP-address-of-node-to-be-queried>/api/cluster/nodes/<node-id-of-node-to-be-queried>
   ```

   The following is a sample response received.

   For details of the meta object, see [Message format](#) on page 10.
We refer to the SPS 6.13.0 REST API Reference Guide on page 326 for details of the meta object.
Elements | Type | Description
--- | --- | ---
items | Top-level element (list of JSON objects) | List of endpoints (objects) available from the current endpoint.
key | string | The ID of the node.
fqdn | string | The address of the node as a fully qualified domain name.
health_status | null or object | The health status of a node. If the node is down, the value is null. Otherwise, the health-related data is listed.
memory | floating point integer (percent) | Memory use
disk | floating point integer (percent) | Hard disk use
swap | floating point integer (percent) | Swap use
<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| load1    | floating point integer | The average system load during the last one minute. The values mean the following:  
|          |               | - < 1: low system load  
|          |               | - 1-5: high system load  
|          |               | - > 5: extremely high system load |
| load5    | floating point integer | The average system load during the last five-minute period. The values mean the following:  
|          |               | - < 1: low system load  
|          |               | - 1-5: high system load  
|          |               | - > 5: extremely high system load |
| load15   | floating point integer | The average system load during the last fifteen-minute period. The values mean the following:  
|          |               | - < 1: low system load  
|          |               | - 1-5: high system load  
|          |               | - > 5: extremely high system load |
| sessions | string        | The protocol type and the number of ongoing sessions. For example:  
|          |               | "sessions": {  
|          |               | "ssh": 3,  
|          |               | "rdp": 4  
<p>|          |               | } |</p>
<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>total_sessions</td>
<td>integer</td>
<td>The total number of ongoing sessions.</td>
</tr>
<tr>
<td>sync_status</td>
<td>string</td>
<td>Indicates the status of configuration synchronization. It has the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- up-to-date: The node has fetched the latest configuration from the Central Management node, and has applied it. It is in sync with the Central Management node.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- pending: There has been a configuration change on the Central Management node, and the change has not been synchronized yet to the node.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- outdated: There has been some error on the node and therefore it is running an old configuration.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- not-fetched: The node has not fetched any configuration yet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- n/a: The node is the Central Management node.</td>
</tr>
<tr>
<td>Elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>meta</td>
<td>Top-level item (JSON object)</td>
<td>Contains links to different parts of the REST service.</td>
</tr>
<tr>
<td>configuration</td>
<td>string (relative path)</td>
<td>The path to the configuration of the node that returned the response.</td>
</tr>
<tr>
<td>href</td>
<td>string (relative path)</td>
<td>The path to the node that returned the response.</td>
</tr>
<tr>
<td>last_seen</td>
<td>string</td>
<td>The last time the node sent status information to the Central Management node, in ISO 8601 format.</td>
</tr>
<tr>
<td>configuration_sync</td>
<td>Top-level item (JSON object)</td>
<td></td>
</tr>
<tr>
<td>downloaded_xml_hash</td>
<td>string</td>
<td>The hash of the latest downloaded configuration file (used for configuration synchronization). If no configuration file has been downloaded yet, it says null.</td>
</tr>
<tr>
<td>last_updated</td>
<td>string</td>
<td>The last time the node's configuration was synchronized, in ISO 8601 format.</td>
</tr>
<tr>
<td>last_checked</td>
<td>string</td>
<td>The last time the node attempted to fetch a new configuration, in ISO 8601 format.</td>
</tr>
<tr>
<td>issues</td>
<td>Top-level</td>
<td>The issues that...</td>
</tr>
</tbody>
</table>

node, so it is not fetching its configuration from any other node.
To query the status of one particular node in your cluster, complete the following steps.

1. **Query the /api/cluster/status/<node-id-of-node-to-be-queried> endpoint on the Central Management node.**

   ```
curl --cookie cookies https://<IP-address-of-Central-Management-node>/api/cluster/status/<node-id-of-node-to-be-queried>
   ``

   The following is a sample response received.

   For details of the meta object, see Message format on page 10.

   For details of the other objects, see tables Cluster status details and "issues" object details.
{  
  "fqdn": "managed-host.cluster",
  "key": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
  "configuration_sync": {  
    "downloaded_xml_hash": "2853830f4aa0a90a63e75bab1b22e513",
    "last_updated": "2018-02-08T09:59:30Z",
    "last_checked": "2018-02-08T09:59:30Z",
    "issues": {}  
  },
  "health_status": {  
    "memory": 62.5,
    "disk": 1.9,
    "swap": 0,
    "load1": 0.53,
    "load5": 0.68,
    "load15": 0.37,
    "sessions": {  
      "ssh": 3,
      "rdp": 4
    },
    "total_sessions": 7
  },
  "sync_status": "up-to-date",
  "last_seen": "2018-02-08T10:00:00Z",
  "meta": {  
    "configuration": "/api/cluster/nodes/46f97a58-4028-467d-9a22-9cfe78ae3e1c",
    "href": "/api/cluster/status/46f97a58-4028-467d-9a22-9cfe78ae3e1c"
  }
}

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Top-level element (list of JSON objects)</td>
<td>List of endpoints (objects) available from the current endpoint.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>The ID of the node.</td>
</tr>
<tr>
<td>fqdn</td>
<td>string</td>
<td>The address of the node as a fully qualified domain name.</td>
</tr>
<tr>
<td>Elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>health_status</td>
<td>null or object</td>
<td>The health status of a node. If the node is down, the value is null. Otherwise, the health-related data is listed.</td>
</tr>
<tr>
<td>memory</td>
<td>floating point integer (percent)</td>
<td>Memory use</td>
</tr>
<tr>
<td>disk</td>
<td>floating point integer (percent)</td>
<td>Hard disk use</td>
</tr>
<tr>
<td>swap</td>
<td>floating point integer (percent)</td>
<td>Swap use</td>
</tr>
<tr>
<td>load1</td>
<td>floating point integer</td>
<td>The average system load during the last one minute. The values mean the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt; 1: low system load</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1-5: high system load</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt; 5: extremely high system load</td>
</tr>
<tr>
<td>load5</td>
<td>floating point integer</td>
<td>The average system load during the last five-minute period. The values mean the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt; 1: low system load</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1-5: high system load</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt; 5: extremely high system load</td>
</tr>
<tr>
<td>load15</td>
<td>floating</td>
<td>The average system</td>
</tr>
<tr>
<td>Elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>point</td>
<td>integer</td>
<td>load during the last fifteen-minute period. The values mean the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt; 1: low system load</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1-5: high system load</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt; 5: extremely high system load</td>
</tr>
<tr>
<td>sessions</td>
<td>string</td>
<td>The protocol type and the number of ongoing sessions. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sessions&quot;: {</td>
</tr>
<tr>
<td>total_sessions</td>
<td>integer (number of)</td>
<td>The total number of ongoing sessions.</td>
</tr>
<tr>
<td>sync_status</td>
<td>string</td>
<td>Indicates the status of configuration synchronization. It has the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• up-to-date: The node has fetched the latest configuration from the Central Management node, and has applied it. It is in sync with the Central Management node.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• pending: There has been a configuration change on the Central Management node, and the</td>
</tr>
<tr>
<td>Elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>change has not been synchronized yet to the node.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>outdated: There has been some error on the node and therefore it is running an old configuration.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>not-fetched: The node has not fetched any configuration yet.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>n/a: The node is the Central Management node, so it is not fetching its configuration from any other node.</td>
<td></td>
</tr>
<tr>
<td>meta</td>
<td>Top-level item (JSON object)</td>
<td></td>
</tr>
<tr>
<td>configuration</td>
<td>string (relative path)</td>
<td></td>
</tr>
<tr>
<td>href</td>
<td>string (relative path)</td>
<td></td>
</tr>
<tr>
<td>last_seen</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>configuration_sync</td>
<td>Top-level item (JSON object)</td>
<td></td>
</tr>
<tr>
<td>Elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>downloaded_</td>
<td>string</td>
<td>The hash of the latest downloaded configuration file (used for configuration</td>
</tr>
<tr>
<td>xml_hash</td>
<td></td>
<td>synchronization). If no configuration file has been downloaded yet, it says</td>
</tr>
<tr>
<td></td>
<td></td>
<td>null.</td>
</tr>
<tr>
<td>last_updated</td>
<td>string</td>
<td>The last time the node's configuration was synchronized, in ISO 8601</td>
</tr>
<tr>
<td></td>
<td></td>
<td>format.</td>
</tr>
<tr>
<td>last_checked</td>
<td>string</td>
<td>The last time the node attempted to fetch a new configuration, in ISO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8601 format.</td>
</tr>
<tr>
<td>issues</td>
<td>top-level</td>
<td>The issues that occurred during configuration synchronization.</td>
</tr>
<tr>
<td></td>
<td>item (JSON</td>
<td>Object)</td>
</tr>
<tr>
<td>issues_element</td>
<td>type</td>
<td>Description</td>
</tr>
<tr>
<td>warning</td>
<td>top-level</td>
<td>Top-level item (JSON object)</td>
</tr>
<tr>
<td></td>
<td>item (JSON</td>
<td>Object)</td>
</tr>
<tr>
<td></td>
<td>object)</td>
<td></td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>Human-readable text explaining why the warning occurred.</td>
</tr>
<tr>
<td>details</td>
<td>array</td>
<td>List of additional information about the warning (for example, the path</td>
</tr>
<tr>
<td></td>
<td></td>
<td>where the warning occurred).</td>
</tr>
<tr>
<td>error</td>
<td>top-level</td>
<td>Top-level item (JSON object)</td>
</tr>
<tr>
<td></td>
<td>item (JSON</td>
<td>Object)</td>
</tr>
<tr>
<td></td>
<td>object)</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The type of the error.</td>
</tr>
<tr>
<td>message</td>
<td>string</td>
<td>Human-readable text explaining why the error occurred.</td>
</tr>
<tr>
<td>details</td>
<td>JSON object</td>
<td>List of additional information about the error (for example, the path where</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the error occurred).</td>
</tr>
</tbody>
</table>

SPS 6.13.0 REST API Reference Guide
Managing SPS
Upload and enable a configuration synchronization plugin

Nodes fetch their configuration from the Central management node, and merge it into their own configuration. Depending on their role, nodes may merge the whole configuration into their own (Managed host nodes), or only the cluster-specific parts (nodes with no roles assigned). Whenever a configuration change is made on the Central management node and the change is committed, it is synchronized to all nodes in the cluster as soon as the nodes fetch the latest configuration from the Central management node.

When synchronizing the central configuration across nodes, you may want to:

- Keep certain parts in the configuration of individual nodes unchanged.
- Customize certain parts of the central configuration to specific needs of individual nodes in the cluster (for example, your nodes may access external services through different network addresses).

You can achieve all of these by using a configuration synchronization plugin that contains transformations for the problematic parts. The plugin only runs on nodes that have the Managed host role.

Customizing certain parts or features of a node using a configuration synchronization plugin has the same limitations as configuring One Identity Safeguard for Privileged Sessions (SPS) through the REST API. In other words, whatever you can configure through the REST API, you can configure the exact same settings using the plugin. One notable difference between the REST API and the plugin is that using the REST API, you can only read certain types of data (such as keys and passwords), while using the configuration synchronization plugin, you can write these types of data as well.

Data structures in the plugin are represented as nested JSON objects. For object references, the plugin uses keys.

The plugin works with the following key parameters:

- local_config: The current configuration of a Managed Host node (those parts that can be configured through the REST API).
- merged_config: The configuration of the Central management node that is about to be synced to the Managed host node (those parts that can be configured through the REST API), with settings related to networking, local services, management, and the license of SPS whitelisted. These settings are never overwritten by configuration synchronization.
- node_id: The unique ID of the Managed host node in the cluster (you can retrieve this identifier by querying the /api/cluster/nodes endpoint through the REST API).
- plugin_config: The configuration of the plugin provided as free-form text. Specifying the configuration of the plugin is optional. It enables you to run configuration synchronization on each cluster with different parameters if you have multiple clusters.
Example: Customizing an IP address in a connection policy

For example, an RDP connection policy on a Managed host node specifies the following client and target addresses:

```bash
$ curl ... https://<url-of-Central-Management-node>/api/configuration/rdp/connections/<id-of-the-connection-policy>
```

```json
{
   "body": {
      "network": {
         "clients": ["0.0.0.0/0"],
         "ports": [3389],
         "targets": ["10.30.255.28/24"]
      }
   },
   ...
}
```

In the following example, an RDP connection policy is configured with the following details on the Central management node:

```bash
$ curl ... https://<url-of-Managed-Node>/api/configuration/rdp/connections/<id-of-the-connection-policy>
```

```json
{
   "body": {
      "network": {
         "clients": ["0.0.0.0/0"],
         "ports": [3389],
         "targets": ["10.30.255.8/24"]
      }
   },
   ...
}
```
To ensure that the details of the connection policy on the Managed host node are kept as-is after configuration synchronization, add the following lines to the plugin main.py file:

```python
$ cat main.py
def merge(local_config: dict, merged_config: dict, node_id: str, plugin_config: str, **kwargs):
    merged_config['rdp']['connections'][<id-of-the-connection-policy>]['network']['targets'][0] = "10.30.255.8/24"
    return merged_config
```

Due to possible new (as yet undefined) parameters, it is good practice to close the parameter list of the merge function with **kwargs.

In case you need assistance with writing customized transformations, contact our Professional Services Team, and a One Identity Service Delivery Engineer will be able to help you.

**NOTE:** Configuration settings related to networking (/api/configuration/network), local services (/api/configuration/local_services), and the management of SPS (/api/configuration/management) are not overwritten on the nodes by configuration synchronization even when not using a plugin.

To upload a configuration synchronization plugin to the Central Management node, complete the following steps.

1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Upload the plugin file.**
   POST the plugin as a zip file (application/zip) to the https://<IP-address-of-Central-Management-node>/api/upload/plugins endpoint, for example:

   ```bash
curl -X POST -H "Content-Type: application/zip" --cookie cookies
   https://<IP-address-of-Central-Management-node>/api/upload/plugins --data-binary @<path-to-plugin.zip>
   ```

   The following is a sample response received.
   For details of the meta object, see Message format on page 10.
```json
{
    "body": {
        "api": "1.0",
        "default_configuration": "",
        "description": "Whitelist the list of paths when merging the configuration",
        "name": "whitelist",
        "path": "/opt/scb/var/plugins/configuration_sync/whitelist",
        "scb_max_version": "",
        "scb_min_version": "",
        "version": "1.0"
    },
    "key": "794a5e17-b8be-4426-8596-0dfc129c06ef",
    "meta": {
        "href": "/api/configuration/plugins/configuration_sync/794a5e17-b8be-4426-8596-0dfc129c06ef",
        "parent": "/api/configuration/plugins/configuration_sync",
        "remaining_seconds": 599
    }
}
```

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>Top-level element (JSON object)</td>
<td></td>
</tr>
<tr>
<td>api</td>
<td>string</td>
<td>Always &quot;1.0&quot;.</td>
</tr>
<tr>
<td>default_configuration</td>
<td>string</td>
<td>Contains the default configuration of the plugin if there is one.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of what the plugin does.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the plugin.</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>The path to the plugin.</td>
</tr>
<tr>
<td>scb_max_version</td>
<td>string</td>
<td>The plugin is compatible with SPS versions not later than this one.</td>
</tr>
<tr>
<td>scb_min_version</td>
<td>string</td>
<td>The plugin is compatible with SPS versions not earlier than this one.</td>
</tr>
<tr>
<td>version</td>
<td>string</td>
<td>The version number of the plugin.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>The ID of the plugin.</td>
</tr>
</tbody>
</table>

SPS 6.13.0 REST API Reference Guide
Managing SPS

ONE IDENTITY by Quest

Page 341
3. **To enable the plugin**

Replace /api/cluster/configuration_sync_plugin with:

```json
{
   "enabled": true,
   "plugin": "<key from-response-of-last-creation>",
   "configuration": ""
}
```

For example:

```bash
curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-Central-Management-node>/api/cluster/configuration_sync_plugin --data '{"enabled": true, "plugin": "794a5e17-b8be-4426-8596-0dfc129c06ef", "configuration": ""}"
```

The following is a sample response received:

```json
{
   "plugin": {
      "key": "794a5e17-b8be-4426-8596-0dfc129c06ef",
      "meta": {
         "href": "https://<IP-address-of-Central-Management-node>/api/configuration/plugins/configuration_sync/794a5e17-b8be-4426-8596-0dfc129c06ef"
      }
   }
}
```

4. **Commit your changes.**

For more information, see [Commit a transaction](#) on page 31.

**Disable a configuration synchronization plugin**

To disable a configuration synchronization plugin on the Central Management node, complete the following steps.

1. **Open a transaction.**

   For more information, see [Open a transaction](#) on page 29.
2. To disable the plugin, replace `/api/cluster/configuration_sync_plugin` with:

```json
{
   "enabled": false
}
```

For example:

```
curl -X POST -H "Content-Type: application/json" --cookie cookies https://<IP-address-of-Central-Management-node>/api/cluster/configuration_sync_plugin --data '{"enabled": false}'
```

The following is a sample response received:

```json
{
   "plugin": {
      "key": null,
      "meta": {}
   }
}
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

### Configuration tools in SPS

A list of tools that can help with the configuration of SPS.

#### URL

**GET** `https://<IP-address-of-SPS>/api/tools/`

#### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, <code>a1f71d030e657634730b9e887cb59a5e56162860</code>.</td>
</tr>
</tbody>
</table>
**Cookie name**    **Description**    **Required**    **Values**

For details on authentication, see Authenticate to the SPS REST API on page 19.

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists available configuration tools in SPS.

```
curl -X GET -b "${COOKIE_PATH}" https://<IP-address-of-SPS>/api/tools/
```

**Response**

The following is a sample response received when the available configuration tools are listed.

For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "hosts-by-name",
            "meta": {
                "href": "/api/tools/hosts-by-name"
            }
        },
        {
            "key": "ldaptest",
            "meta": {
                "href": "/api/tools/ldaptest"
            }
        }
    ],
}
```

**Item**    **Description**

| Resolving hostnames to IP addresses on page 345 | Resolve the hostname of a computer or server to IP addresses. |
| Testing LDAP server connection on page 347 | Test LDAP server connection. |
HTTP response codes

For more information and a list of standard HTTP response codes, see Application level error codes on page 37.

Resolving hostnames to IP addresses

SPS configuration requires you to set IP addresses as values. Resolve the hostname of a computer or server with the /hosts-by-name endpoint to receive the list of all related IP addresses that you can use for configuration.

| NOTE: The protocol parameter can only take the following two values: TCP and UDP. Anything else will return an error message.

URL

POST https://<IP-address-of-SPS>/api/tools/hosts-by-name

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

| NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format). |

Sample request

The following command resolves the hostname to IP addresses.

```
curl -X POST -b "${COOKIE_PATH}" https://<IP-address-of-SPS>/api/tools/hosts-by-name
```
Elements of the request message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Required</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostname</td>
<td>string</td>
<td>Required</td>
<td>The unique identifier that serves as the name of the computer or server whose IP address you want to resolve.</td>
<td></td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>Required</td>
<td>The type of Internet Protocol used to address and route packets of data.</td>
<td>Possible values are: TCP, UDP</td>
</tr>
</tbody>
</table>

**Response**

When resolving a hostname to IP addresses, the response is the following.

For details of the meta object, see Message format on page 10.

```
{
    "ipv4": [
        "93.184.216.34",
        "93.184.216.35"
    ],
    "ipv6": [
    ]
}
```

Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipv4</td>
<td>string array</td>
<td>The Internet Protocol type of the IP addresses is version 4.</td>
<td></td>
</tr>
<tr>
<td>ipv6</td>
<td>string array</td>
<td>The Internet Protocol type of the IP addresses is version 6.</td>
<td></td>
</tr>
</tbody>
</table>
HTTP response codes

HTTP response codes comprise of standard or endpoint-specific HTTP status and error codes. The following table lists the endpoint-specific HTTP response codes for this request.

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status/Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Syntactic Error</td>
<td>The protocol you provided is not valid. Use TCP or UDP as value instead.</td>
</tr>
<tr>
<td>400</td>
<td>HostnameCannotBeResolved</td>
<td>The hostname you provided cannot be resolved. Check the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The hostname you provided is valid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The hostname is available on the Internet.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>Unauthenticated users cannot query the IP addresses of a host.</td>
</tr>
</tbody>
</table>

For more information and a list of standard HTTP response codes, see Application level error codes on page 37.

Testing LDAP server connection

Use the /ldaptest endpoint to test whether your LDAP server configuration was successful and the connection between the LDAP server and SPS can be established. You can identify connection issues based on the specific error messages received (for example, unsuccessful authorization due to incomplete credentials).

URL

GET https://<IP-address-of-SPS>/api/tools/ldaptest

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication</td>
</tr>
<tr>
<td>Cookie name</td>
<td>Description</td>
<td>Required</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td>token</td>
<td>token of the user</td>
<td>response, for example, a1f71d030e657634730b9e887cb59a5e56162860.</td>
<td>For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists endpoints for testing LDAP server availability.

```
curl -k -X GET -b "${COOKIE_PATH}" https://<IP-address-of-SPS>/api/tools/ldaptest
```

**Response**

The following is a sample response received when LDAP test endpoints are listed.

For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "aaa",
            "meta": {
                "href": "/api/tools/ldaptest/aaa"
            }
        },
        {
            "key": "policies",
            "meta": {
                "href": "/api/tools/ldaptest/policies"
            }
        }
    ],
    "meta": {
        "aaa": "/api/tools/ldaptest/aaa",
        "href": "/api/tools/ldaptest",
```
Operations

Operations with the /ldaptest endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>List endpoints for testing LDAP server availability</td>
<td>GET</td>
<td>/api/tools/ldaptest</td>
<td></td>
</tr>
<tr>
<td>Testing a LDAP server</td>
<td>POST</td>
<td>/api/tools/ldaptest/policies</td>
<td></td>
</tr>
<tr>
<td>Testing an AAA LDAP server</td>
<td>POST</td>
<td>/api/tools/ldaptest/aaa</td>
<td></td>
</tr>
</tbody>
</table>

Example: Testing LDAP server connection

To test your connection to a LDAP server, you have to

1. Authenticate to SPS. For more information, see Authenticate to the SPS REST API on page 19.
2. List the available LDAP servers that you have previously configured.
   
   ```
   curl -k -X GET -b "$\{COOKIE\_PATH\}" https://<IP-address-of-SPS>/api/configuration/policies/ldap_servers
   ```
3. Select the key of the LDAP server that you want to test.
4. Use the key to test whether you can connect to the LDAP server.
   
   ```
   curl -k -X POST -H "Content-Type: application/json" -b "$\{COOKIE\_PATH\}" https://<IP-address-of-SPS>/api/tools/ldaptest/policies --data '{"key": "200796612861e564724534a"}'}
   ```

Response

The following is a sample response received when you test your connection to a LDAP server.
For details of the meta object, see Message format on page 10.

```json
{
    "meta": {
        "href": "/api/tools/ldaptest/policies",
        "ldap_server": "/api/configuration/aaa/ldap_servers/200796612861e564724534a",
        "parent": "/api/tools/ldaptest",
        "remaining_seconds": 600
    },
    "success": true
}
```

Example: Testing AAA LDAP server connection

To test your connection to an AAA LDAP server, you have to

1. Authenticate to SPS. For more information, see Authenticate to the SPS REST API on page 19.
2. List the available AAA LDAP servers that you have previously configured.

   ```bash
curl -k -X GET -b "${COOKIE_PATH}" https://<IP-address-of-SPS>/api/configuration/aaa/ldap_servers
   
   curl -k -X POST -H "Content-Type: application/json" -b "${COOKIE_PATH}" https://<IP-address-of-SPS>/api/tools/ldaptest/aaa --data "{"key": "200796612861e564724534a"}'
```

Response

The following is a sample response received when you test your connection to an AAA LDAP server.

For details of the meta object, see Message format on page 10.

```json
{
    "meta": {
        "href": "/api/tools/ldaptest/policies",
        "ldap_server": "/api/configuration/aaa/ldap_servers/200796612861e564724534a",
    }
}```
Elements of the response message body include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>success</td>
<td>boolean</td>
<td>Indicates that the connection to the provided LDAP server could be established.</td>
<td>Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• true - the test was successful</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• false - the test was not successful</td>
</tr>
<tr>
<td>meta.href</td>
<td>string (relative path)</td>
<td>Path of the resource that returned the response. When creating a new object, this is the URL of the created object.</td>
<td></td>
</tr>
<tr>
<td>meta.ldap_server</td>
<td>string (relative path)</td>
<td>Identifier of the LDAP server that was tested.</td>
<td></td>
</tr>
<tr>
<td>meta.parent</td>
<td>string (relative path)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>meta.remaining_seconds</td>
<td>integer</td>
<td>Time left until the session times out in seconds.</td>
<td>SPS closes idle sessions after a period of inactivity. This value shows the number of seconds left until the timeout. For details on setting the session timeout, see Web interface.</td>
</tr>
</tbody>
</table>

**HTTP response codes**

HTTP response codes comprise of standard or endpoint-specific HTTP status and error codes. The following table lists the endpoint-specific HTTP response codes for this request.
<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status/Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>TransactionMustBeClosed</td>
<td>The user attempted to test a LDAP server while a transaction was open (for example, there were pending configuration changes while the test was attempted). Commit or roll back the transaction before testing the LDAP server.</td>
</tr>
<tr>
<td>400</td>
<td>NoSuchLDAPServer</td>
<td>There are no LDAP servers configured with the identifier specified in the request:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-data '{&quot;key&quot;: &quot;&lt;invalid value&gt;&quot;}'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Make sure that you use a valid key from the list of configured LDAP servers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This can also occur if you attempt to test an AAA LDAP server using the /ldaptest/policies endpoint or vice versa.</td>
</tr>
<tr>
<td>400</td>
<td>LDAPServerWithNoBindPassword</td>
<td>You can connect to a LDAP server using a bind DN (bind_dn) and a bind password (bind_password). These credentials are used for authentication. Normally, you need both credentials to make a successful connection, so that access to the LDAP server can be controlled. Some LDAP servers might be configured to allow so-called anonymous connections, meaning that there is no access control whatsoever and anybody can connect to that server. To test such a connection, there must be no credentials configured for that LDAP server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- valid: a bind_dn and a bind_password are both configured, or there is no bind_dn and no bind_password.</td>
</tr>
<tr>
<td>HTTP response code</td>
<td>Status/Error</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>password</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• invalid: there is a bind_dn, but no bind_password, or there is a bind_password, but no bind_dn</td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>LDAPServerWithNoBindDn</td>
<td>See LDAPServer-WithNoBindPassword.</td>
</tr>
<tr>
<td>500</td>
<td>LDAPConnectionFailure</td>
<td>Something went wrong during testing the connection. The most common reason would be an incorrect server address/port, DNS resolution failure, firewalls, or a damaged networking cable. Make sure that the configured address/port is reachable from SPS.</td>
</tr>
</tbody>
</table>

For more information and a list of standard HTTP response codes, see Application level error codes on page 37.
General connection settings

Channel policy

The channel policy lists the channels (for example, terminal session and SCP in SSH, Drawing, Clipboard in RDP) that can be used in a connection. The channel policy can further restrict access to each channel based on the IP address of the client or the server, a user list, user group, or a time policy. For example, all clients may access the servers defined in a connection via SSH terminal, but the channel policy may restrict SCP access only to a single client. The policies set in the channel policy are checked when the user attempts to open a particular channel type in the connection.

Channel policies are protocol specific. To list the available Channel policies for a protocol, use the following command.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/<http|ica|rdp|ssh|telnet|vnc>/channel_policies
```

The following sections detail the properties of Channel policy objects.

URL

```
GET https://<IP-address-of-SPS>/api/configuration/<http|ica|rdp|ssh|telnet|vnc>/channel_policies/<object-id>
```

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860.</td>
</tr>
<tr>
<td>Cookie name</td>
<td>Description</td>
<td>Required</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
</tr>
</tbody>
</table>

For details on authentication, see Authenticate to the SPS REST API on page 19.

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the properties of a specific RDP Channel policy object.

```bash
curl --cookie cookies -https:<IP-address-of-SPS>/api/configuration/<rdp>/channel_policies/<object-id>
```

**Response**

The following is a sample response received, showing the properties of Channel policy objects.

For details of the **meta** object, see **Message format** on page 10.

```
{
    "body": {
        "name": "terminal-only",
        "rules": [
            {
                "actions": {
                    "audit": true,
                    "content_policy": null,
                    "four_eyes": false,
                    "ids": false
                },
                "allowed_for": {
                    "clients": [],
                    "gateway_groups": [],
                    "remote_groups": [],
                    "servers": [],
                    "time_policy": {
                        "key": "-100",
                        "meta": {
                            "href": "/api/configuration/policies/time_policies/-100"
                        }
                    }
                }
            }
        ]
    }
}```
Element | Type | Description
--- | --- | ---
name | string | Top level element, the name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.

rules | list of JSON objects | Top level element, contains the configuration properties of the object.

actions | JSON object | The actions that SPS performs for the channel, for example, recording the traffic into an audit trail.

allowed_for | JSON object | Specifies the access control rules of the channel, for example, permitted target IP addresses or usergroups.

channel | string | The type of the channel. Note that channels are protocol specific, and different type of channels can have different parameters.

- For details on HTTP-specific channels, see [HTTP channels](#) on page 452.
### Element Type Description

- For details on Citrix ICA-specific channels, see [ICA channels on page 484](#).
- For details on RDP-specific channels, see [RDP channels on page 565](#).
- For details on SSH-specific channels, see [SSH channels on page 611](#).
- For details on Telnet-specific channels, see [Telnet channels on page 668](#).

For example:

```
"channel": "#drawing",
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actions</td>
<td>JSON object</td>
<td>The list of actions to perform when the Content policy matches the analyzed traffic. All actions are boolean values (true or false)</td>
</tr>
<tr>
<td>audit</td>
<td>boolean</td>
<td>Set to true to record the activities of the channel into audit trails. Possible values: true or false</td>
</tr>
<tr>
<td>content_policy</td>
<td>JSON object</td>
<td>Specifies the Content policy to use in the channel, otherwise its value is null (which is the default). For details on Content policies, see <a href="#">Real-time content monitoring with Content Policies</a>. For example:</td>
</tr>
<tr>
<td>four_eyes</td>
<td>boolean</td>
<td>Set to true to require four-eyes authorization to access the channel. For details, see &quot;<a href="#">Configuring four-eyes authorization</a>&quot; in the Administration Guide. Possible values: true or false</td>
</tr>
<tr>
<td>allowed_for</td>
<td>JSON object</td>
<td>Specifies the access control rules of the channel.</td>
</tr>
<tr>
<td>clients</td>
<td>list</td>
<td>To restrict the availability of the channel only to certain clients, list the IP address or network of the clients allowed to use this the channel. For IPv6 addresses, use the canonized format of the address. For example:</td>
</tr>
</tbody>
</table>
### Element | Type | Description
--- | --- | ---
"clients": [ "192.168.1.1/24", "2001:db8:85a3::8a2e:0:0/32"

Alternatively, you can also enter a hostname instead. One Identity Safeguard for Privileged Sessions (SPS) saves the hostname and resolves it when opening channels, therefore SPS can trace dynamic IP addresses.

**NOTE:** Note the following limitations:

- The Domain Name Servers you set must be able to resolve the hostnames you enter into the clients and servers fields, otherwise this function (and, therefore, the sessions using this Channel Policy) will not work.

- SPS Channel Policies support wildcard characters in the *.example.com format. If the channel opening request contains an IP address, SPS uses a reverse lookup method to resolve this IP address into a hostname for a match.

- SPS uses the Domain Name Servers set in the /api/configuration/network/dns endpoint to resolve the hostnames.

| gateway_groups | list | You can control channel access during gateway authentication with blacklists or whitelists of user groups. You can use local user lists on SPS, or LDAP groups. To use this option, you must also configure web gateway authentication in the connection policy, or client-side gateway authentication back-end in the authentication policy. For example:

"gateway_groups": ["group1", "group2"],

To configure local user lists, see User lists on page 425.

| remote_groups | list | You can control channel access during authentication to the remote server with blacklists or whitelists of...
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user groups. You can use local user lists on SPS, or LDAP groups. For example:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;remote_groups&quot;: [&quot;group1&quot;, &quot;group2&quot;],</td>
</tr>
<tr>
<td>To configure local user lists, see User lists on page 425.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>servers</td>
<td>list</td>
<td>To restrict the availability of the channel only to certain servers, list the IP address or network of the servers that your clients are allowed to access using this the channel. For IPv6 addresses, use the canonized format of the address. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;servers&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;192.168.1.1/24&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;2001:db8:85a3::8a2e:0:0/32&quot;</td>
</tr>
<tr>
<td>Alternatively, you can also enter a hostname instead. One Identity Safeguard for Privileged Sessions (SPS) saves the hostname and resolves it when opening channels, therefore SPS can trace dynamic IP addresses.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOTE: Note the following limitations:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The Domain Name Servers you set must be able to resolve the hostnames you enter into the clients and servers fields, otherwise this function (and, therefore, the sessions using this Channel Policy) will not work.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPS Channel Policies support wildcard characters in the *.example.com format. If the channel opening request contains an IP address, SPS uses a reverse lookup method to resolve this IP address into a hostname for a match.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPS uses the Domain Name Servers set in the /api/configuration/network/dns endpoint to resolve the hostnames.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alternatively, you can configure a custom DNS server to be used for target selection custom_dns field of the Connection Policy.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>time_policy</td>
<td>JSON object</td>
<td>Specifies the Time policy to use in the channel. If you do not want to restrict access, use the default 7x24 policy-100. For details on Time policies, see Time policy on page 410. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;time_policy&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;key&quot;: &quot;-100&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

**Policies**

List of endpoints for configuring policies and settings that can be referenced when configuring connections.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/policies

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d03e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the available endpoints.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies
Response

The following is a sample response received when listing the available configuration endpoints.
For details of the meta object, see Message format on page 10.

```json
{
  "items": [
    {
      "key": "aa_plugin_instances",
      "meta": {
        "href": "/api/configuration/policies/aa_plugin_instances"
      }
    },
    {
      "key": "analytics",
      "meta": {
        "href": "/api/configuration/policies/analytics"
      }
    },
    {
      "key": "archive_cleanup_policies",
      "meta": {
        "href": "/api/configuration/policies/archive_cleanup_policies"
      }
    },
    {
      "key": "audit_policies",
      "meta": {
        "href": "/api/configuration/policies/audit_policies"
      }
    },
    {
      "key": "backup_policies",
      "meta": {
        "href": "/api/configuration/policies/backup_policies"
      }
    },
    {
      "key": "content_policies",
      "meta": {
        "href": "/api/configuration/policies/content_policies"
      }
    },
    {
      "key": "credentialstores",
      "meta": {
        "href": "/api/configuration/policies/credentialstores"
      }
    }
  ]
}```
null,
<table>
<thead>
<tr>
<th>Endpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa_plugin_instances</td>
<td>Authentication and Authorization plugin policies</td>
</tr>
<tr>
<td>analytics</td>
<td>Analytics</td>
</tr>
<tr>
<td>archive_cleanup_policies</td>
<td>Archive/Cleanup policies.</td>
</tr>
<tr>
<td>audit_policies</td>
<td>Audit trail encryption, timestamping, and signing.</td>
</tr>
<tr>
<td>backup_policies</td>
<td>Backup policies</td>
</tr>
<tr>
<td>content_policies</td>
<td>Actions for detected commands, screen content, credit card information, and window titles.</td>
</tr>
<tr>
<td>credentialstores</td>
<td>Local and external credential stores.</td>
</tr>
<tr>
<td>indexing</td>
<td>Languages for Optical Character Recognition (OCR).</td>
</tr>
<tr>
<td>ldap_servers</td>
<td>LDAP servers</td>
</tr>
<tr>
<td>signing_cas</td>
<td>Signing CAs for generating the server-side certificates on the fly. You can use such CAs in SSL-encrypted RDP sessions, RDP sessions that use Network Level Authentication (CredSSP), or SSH connections that use X.509-based authentication. To configure signing for audit trails, use the audit_policies endpoint.</td>
</tr>
<tr>
<td>time_policies</td>
<td>Time policies</td>
</tr>
<tr>
<td>trusted_ca_lists</td>
<td>Trusted Certificate Authorities (CAs), and options for restricting the accepted certificates.</td>
</tr>
<tr>
<td>user_databases</td>
<td>Local User Databases are available for RDP, SSH and Telnet protocols, and can be used to authenticate the clients to credentials (passwords, public keys, and certificates) that are locally available on SPS.</td>
</tr>
<tr>
<td>Endpoint</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>userlists</td>
<td>Local white- or blacklists of usernames that allow fine-control over who can access a connection or a channel.</td>
</tr>
<tr>
<td>usermapping_policy</td>
<td>Usermapping policies describe who can use a specific username to access the remote server.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Archive/Cleanup policy

Archiving transfers data from SPS to an external storage solution, cleanup removes (deletes) old files. Archived data can be accessed and searched, but cannot be restored (moved back) to the SPS appliance. Only those closed audit-trail files are archived where the retention time has already elapsed. To list the available Archive policies, use the following command.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/archive_cleanup_policies/```

The following sections detail the properties of Archive/Cleanup policy objects.

### URL

GET https://<IP-address-of-SPS>/api/configuration/policies/archive_cleanup_policies/<object-id>
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following request command lists the properties of a specific Archive/Cleanup policy object.

```
curl --cookie cookies -https:<IP-address-of-SPS>/api/configuration/policies/archive_cleanu_policies/<object-id>
```

Response

The following is a sample response received, showing the properties of Archive/Cleanup policy objects.

For details of the meta object, see **Message format** on page 10.

```
{
  "key": "99375192754364c2b1bd01",
  "body": {
    "name": "archive_all_with_filelist",
    "include_node_id_in_path": false,
    "notification_event": {
      "type": "all",
      "send_filelist": true,
      "file_count_limit": 123456
    },
    "target": {
      "type": "nfs",
      "server": {
        "selection": "ip",
        "value": "1.2.3.5"
      }
    }
  }
```

SPS 6.13.0 REST API Reference Guide
General connection settings
```json
{
  "path": "/data/backup",
  "start_times": [
    "10:10"
  ],
  "template": "PROTOCOL/CONNECTION/ARCHIVEDATE/",
  "retention_days": 30
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string (smb</td>
<td>nfs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>server</td>
<td>Top level</td>
<td>element</td>
</tr>
<tr>
<td>domain</td>
<td>string</td>
<td>Only if type is set to smb.</td>
</tr>
<tr>
<td>protocol_version</td>
<td>string</td>
<td>Only if type is set to smb.</td>
</tr>
<tr>
<td>share</td>
<td>string</td>
<td>Only if type is set to smb.</td>
</tr>
<tr>
<td>authentication</td>
<td>Top level</td>
<td>element</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>The path to the archive directory on the target server</td>
</tr>
<tr>
<td>start_times</td>
<td>list of strings</td>
<td>The time when the archive process starts in H:MM or HH:MM format.</td>
</tr>
<tr>
<td>template</td>
<td>string</td>
<td>SPS organizes the audit trails into directories based on the date or the protocol. The subdirectories are created directly into the archive directory. The following subdirectory structures are</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>retention_days</td>
<td>integer (days)</td>
<td>Data older than this value is archived to the external server. The archived data is deleted from SPS.</td>
</tr>
</tbody>
</table>

**Elements of server**

<table>
<thead>
<tr>
<th>server</th>
<th>Top level element</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string (ip</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>string</td>
</tr>
</tbody>
</table>

**Elements of authentication**

<table>
<thead>
<tr>
<th>authentication</th>
<th>Top level element</th>
<th>Only if type is set to smb.</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string (password</td>
<td>anonymous)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>anonymous: To log on anony- mously.</td>
</tr>
<tr>
<td>username</td>
<td>string</td>
<td>Only if selection is set to password. The username used to log on to the remote server</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>Only if selection is set to password. The password corresponding to the username</td>
</tr>
</tbody>
</table>
Audit policies

The list of audit policies. An audit policy contains settings for encrypting, timestamping, and signing audit trails. To enable auditing for a connection, select an audit policy when configuring connections, and enable auditing for the appropriate protocol channels in the connection's channel policy.

**NOTE:** The default audit policy is pre-selected when creating connection policies. Modify that audit policy with care.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/policies/audit_policies
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the audit policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/audit_policies
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/policies/audit_policies/<policy-id>
```
Response

The following is a sample response received when listing audit policies.

For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "78101850949e47437dd91d",
      "meta": {
        "href": "/api/configuration/policies/audit_policies/78101850949e47437dd91d"
      }
    },
    {
      "key": "9161063345713f11489305",
      "meta": {
        "href": "/api/configuration/policies/audit_policies/9161063345713f11489305"
      }
    },
    {
      "key": "1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
      "meta": {
        "href": "/api/configuration/policies/audit_policies/1e089e2a-76b4-4079-94e3-c83ebc74dc2e"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/policies/audit_policies",
    "href": "/api/configuration/policies/audit_policies",
    "last": "/api/configuration/policies/usermapping_policies",
    "next": "/api/configuration/policies/content_policies",
    "parent": "/api/configuration/policies",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

When retrieving the endpoint of a specific audit policy, the response is the following.

```
{
  "body": {
    "encryption": {
      "certificates": [
        {
          "certificate": "<cert1>",
          "four_eyes_certificate": "<cert2>"
        }
      ]
    }
  }
}
```
"different_certificates_for_upstream": {
  "certificates": [
    {
      "certificate": "<cert3>",
      "four_eyes_certificate": "<cert4>
    }]
  },
  "enabled": true
},
  "enabled": true
},
  "name": "<policy-name>",
  "signing": {
    "enabled": true,
    "x509_identity": {
      "key": "ec0b6604-37f6-4df6-bd2f-d7879a75b324",
      "meta": {
        "href": "/api/configuration/x509/ec0b6604-37f6-4df6-bd2f-d7879a75b324"
      }
    },
    "timestamping_enabled": true
  },
  "key": "1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
  "meta": {
    "first": "/api/configuration/policies/audit_policies/78101850947e47437dd91d",
    "href": "/api/configuration/policies/audit_policies/1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
    "last": "/api/configuration/policies/audit_policies/1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
    "next": null,
    "parent": "/api/configuration/policies/audit_policies",
    "previous": "/api/configuration/policies/audit_policies/916106334573f11489305",
    "transaction": "/api/transaction"
  }
}
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encryption</td>
<td>Top level element</td>
<td>Audit trail encryption settings.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the policy. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>signing</td>
<td>Top level element</td>
<td>Audit trail signing settings.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable audit trail signing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If signing is enabled, the x509_identity element is also required.</td>
</tr>
<tr>
<td>x509_identity</td>
<td>string</td>
<td>Required for signing audit trails.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>References the identifier of the X.509 certificate stored on SPS. You can configure certificates at the /api/configuration/x509/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an X.509 host certificate, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>timestamping</td>
<td>boolean</td>
<td>Set to true to timestamp the audit trail.</td>
</tr>
</tbody>
</table>

**Elements of encryption**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificates</td>
<td>Top level list</td>
<td>Contains the encrypting certificates.</td>
</tr>
<tr>
<td>certificate</td>
<td>string</td>
<td>The encrypting certificate. You can replay an encrypted audit trail with the private key of the encrypting certificate.</td>
</tr>
<tr>
<td>four_eyes_certificate</td>
<td>string</td>
<td>Additional certificate for joint (4-eyes) encryption. You can only replay a jointly encrypted audit trail with the private keys of both certificates.</td>
</tr>
</tbody>
</table>
| different_certificates| Top level item      | Configures encrypting upstream traffic separ-
<table>
<thead>
<tr>
<th>Elements of encryption</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>for_upstream</td>
<td></td>
<td></td>
</tr>
<tr>
<td>certificates</td>
<td>Top level list</td>
<td>The certificates for encrypting upstream traffic.</td>
</tr>
<tr>
<td>certificate</td>
<td>string</td>
<td>The encrypting certificate. You can replay an encrypted upstream with the private key of the encrypting certificate.</td>
</tr>
<tr>
<td>four_eyes_certificate</td>
<td>string</td>
<td>Additional certificate for joint (4-eyes) encryption. You can only replay a jointly encrypted upstream with the private keys of both certificates.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to encrypt the upstream traffic with separate certificate(s). If upstream encryption is enabled, the certificates element is required.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable encrypting audit trails. If encryption is enabled, the certificates and different_certificates_for_upstream elements are required.</td>
</tr>
</tbody>
</table>

**Examples:**

Disable encryption, signing, and timestamping.

```json
{
  "encryption": {
    "enabled": false
  },
  "name": "default",
  "signing": {
    "enabled": false
  },
  "timestamping_enabled": false
}
```

Encrypt upstream traffic only (single certificate).
Enable signing and timestamping, no traffic encryption.

Enable signing and timestamping, and encrypt traffic with a single certificate (no separate upstream encryption).
Encrypting certificates

Encrypting certificates must not contain any metadata. SPS uses only the key part of the certificate, no other data (expiry, etc.) are relevant for encryption.

To use a certificate with the SPS API, remove all metadata, and substitute line breaks with \n.

The following is an example certificate, as used on the SPS web interface:

```plaintext
-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCCQCQcS36ObStIPQTANBgkqkhiG9w0BAQUFADCBjzELMAKGA1UEBhMCQ0ExEDA0BgNVBAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoTBBJhbGFiaXQxEDAOBgNVBAsTDURvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFiaXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYm10Lmh1MB4XDTE1MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAYGQVAFIBTQXG1uwPbnR⩽mYVUwRMAwDgYDVQQHEwVodWF2tElYwYwFAAYDVQQLEw1Elb2N1bWVudGFydmFyU015MTAvMQSwQgYJKoZIhvcNDAQBFHFyjXRhaAwAYMswYJpc5odTCCASISDEQBBQADgEPADCCAQoCggEBAAg9I2jmVldVWEi/ky7ahTeyaIjK52FQUxG8ok0SD+nV7ZKFUuiS51X+20w1aDqVGrDMgPNSyVPxYuDVUA0Ij4Jw4rAIoxDY6vDU9/4v9dIqFEPaluw0qNRJPSMLzjS0QDSKqPdkivk56HKZeX3+TFq3oxO+vIrFr9zF9p9T+eDg2o5obPc3mV2zvVd61CIxbeVzAdArD16wnsRyzyxYHWEhFwZepNXFD9Y5SNzKody7HncX5Kv1v6+26bhgF7/7hWhtysJdWnuwrl8By70VPM6wdA83k3Fy2gYNYk7Rc08bRFbQTXhJVFhzuSUWHVhIFqtAb4d1KU5qvqepFMCMAwEATANTBgkqkh1I9w08BAQFAOCAQEA65DJoWHSwK0GkiI3CH2VMxP2rR?qTnne6E1+DFlgOQDx1+aVnq4T4dnPnFVAPB1kqbmC4hJAsjMLU2b1ne6m+SLmzhrMxcA6x+fnYve5QT51bRdq2E/4oJGeyuy
-----END CERTIFICATE-----
```
Add an audit policy

To add an audit policy, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new audit policy.**

   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/policies/audit_policies` endpoint. You can find a detailed description of the available parameters listed in Element.

   If the POST request is successful, the response includes the key of the new audit policy. For example:

   ```
   \n   \n   \n   \n   \n   \n   ```
3. **Commit your changes.**

For more information, see Commit a transaction on page 31.

### Modify an audit policy

To modify an audit policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the audit policy.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/policies/audit_policies/<policy-key>` endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>--------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

## Backup policy

Backup policies define the address of the backup server, which protocol to use to access it, and other parameters. To list the available Backup policies, use the following command.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/backup_policies/
```

The following sections detail the properties of Backup policy objects.

### URL

GET https:<IP-address-of-SPS>/api/configuration/policies/backup_policies/<object-id>

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists the properties of a specific Backup policy object.
curl --cookie cookies -https:<IP-address-of-SPS>/api/configuration/policies/backup_policies<object-id>

Response

The following is a sample response received, showing the properties of Backup policy objects.
For details of the meta object, see Message format on page 10.

```
{
    "key": "99275192754364c2b1bd01",
    "body": {
        "name": "backup_all_with_filelist",
        "include_node_id_in_path": false,
        "notification_event": {
            "type": "all",
            "send_filelist": true,
            "file_count_limit": 123456
        },
        "target": {
            "type": "nfs",
            "server": {
                "selection": "ip",
                "value": "1.2.3.5"
            },
            "path": "/data/backup"
        },
        "start_times": [
            "10:10"
        ]
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Top level element, the name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>include_node_id_in_path</td>
<td>boolean</td>
<td>Include the Cluster Node ID in the path. Recommended to set to True if the SPS instance is a node in a cluster. This ensures that the ID of the node is included in the path of the relevant directory, which is required to prevent cluster nodes from backing up data to</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>the same location, and so overwriting each other's data and resulting in data loss.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>notification_event</td>
<td>Top level element</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>• all: Sends notification emails on all backup-related events.</td>
</tr>
<tr>
<td></td>
<td>(all</td>
<td>errors-only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• errors-only: Sends notification emails only on backup-related errors.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• none: Sends no backup-related notification emails.</td>
</tr>
<tr>
<td>send_filelist</td>
<td>boolean</td>
<td>This is meaningful only if notification_event is set to all.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>True if the list of files are included in the notification e-mail.</td>
</tr>
<tr>
<td>file_count_limit</td>
<td>integer</td>
<td>This is meaningful only if notification_event is set to all and send_filelist is set to True.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The maximum number of files that are included in the notification e-mail.</td>
</tr>
<tr>
<td>target</td>
<td>Top level element</td>
<td>Defines the address of the backup server, which protocol to use to access it, and other parameters. SPS can be configured to use the Rsync, SMB/CIFS, and NFS protocols to access the backup server.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>• rsync: Rsync over SSH</td>
</tr>
<tr>
<td></td>
<td>(rsync</td>
<td>smb</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• smb: Copy data to a remote server using SMB/CIFS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• nfs: Copy data to a remote server using NFS</td>
</tr>
<tr>
<td>server</td>
<td>Top level element</td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>string</td>
<td>Only if type is set to smb.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The domain name of the target server</td>
</tr>
<tr>
<td>protocol_version</td>
<td>string</td>
<td>Only if type is set to smb.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>share</td>
<td>string</td>
<td>Only if type is set to smb. The name and directory path of the share in the following format: share_name/path/to/directory</td>
</tr>
<tr>
<td>authentication</td>
<td>Top level element</td>
<td>Only if type is set to smb.</td>
</tr>
<tr>
<td>username</td>
<td>string</td>
<td>Only if type is set to rsync. The username used to log on to the remote server</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>The path to the backup directory on the target server</td>
</tr>
<tr>
<td>auth_key</td>
<td>JSON object</td>
<td>Only if type is set to rsync. This key will be used to authenticate SPS on the remote server. The public key of this keypair must be imported to the remote server. For details on private keys, see Private keys stored on SPS on page 267. For example:</td>
</tr>
<tr>
<td>host_key</td>
<td>Top level element or string</td>
<td>Only if type is set to rsync.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>port</td>
<td>integer</td>
<td>Only if type is set to rsync. The port number of the SSH server running on the remote machine.</td>
</tr>
<tr>
<td>start_times</td>
<td>list of strings</td>
<td>The time when the archive process starts in H:MM or HH:MM format.</td>
</tr>
</tbody>
</table>

### Elements of server

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>Top level element</td>
<td></td>
</tr>
<tr>
<td>selection</td>
<td>string (ip</td>
<td>fqdn)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fqdn: Hostname</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The IP address or the hostname of the remote server</td>
</tr>
</tbody>
</table>

### Elements of authentication

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>Top level element</td>
<td>Only if type is set to smb.</td>
</tr>
<tr>
<td>selection</td>
<td>string (password</td>
<td>anonymous)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>anonymous: To log on anonymously.</td>
</tr>
<tr>
<td>username</td>
<td>string</td>
<td>Only if selection is set to password. The username used to log on to the remote server</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>Only if selection is set to password. The password corresponding to the username</td>
</tr>
<tr>
<td>Description</td>
<td>T-Only if type is set to rsync.</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------</td>
<td></td>
</tr>
<tr>
<td>host_key</td>
<td>When editing this policy, for usability purposes, you can enter the public key of the host in the host_key element without using the selection and value elements. For example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;host_key&quot;: &quot;ssh-rsa AAAAB3Nz-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>aC1yc2EAAAADAQABAAABAQDmIDa1PuJFzgvZvPs9hzgvMd/9WI4J7RBfu0769g/OgTvCRT-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>grF8/IM/0iN0YzcUM3IGyPnJ1OLLE2Gb6CxVvEcjp6pme73roAwo039wQHR3Rx11KoEmC+0EO-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ImQycIdAS7-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>grWNwD2VB2S7iyFerZhqRx-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>hGJPkB/R/kF31Q3d6t-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>t3pr4+R6wnU91Z7RSETfB+N09FE4f5Nqy+VEShg-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>dc66E1FRXXVilmiTnIMAy-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>im3T7UVNgRdZIYIAZ79tyk7p6I+DZ7k7BG9TYwdBj-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wr0eVL56ILxpXy1pxwONwMhLXkLsL42NfmeagjVUD1CJVoRaGjCVGEeS3iQs6GVVxe78n&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When querying, the public key of the host will always be displayed in the selection and value elements.</td>
<td></td>
</tr>
<tr>
<td>s-</td>
<td>The algorithm the key is based on.</td>
<td></td>
</tr>
</tbody>
</table>

SPS 6.13.0 REST API Reference Guide
General connection settings
Example: querying an Rsync backup policy

When the query is the following:
curl --cookie cookies "https://<IP-address-of-SPS>/api/configuration/policies/backup_policies/99275192754364c2b1bd04"

The response is the following:

```json
{
   "key": "99275192754364c2b1bd04",
   "body": {
      "name": "backup_rsync",
      "include_node_id_in_path": true,
      "notification_event": {
         "type": "none",
         "send_filelist": true,
         "file_count_limit": 10240
      },
      "target": {
         "type": "rsync",
         "server": {
            "selection": "ip",
            "value": "192.168.122.1"
         },
         "username": "user1",
         "path": "/data/backup",
         "auth_key": {
            "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "meta": {
               "href": "/api/configuration/private_keys/XXXXXXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
            }
         },
         "host_key": {
            "selection": "rsa",
            "value": "AAAAB3NzaC1yc2EAAAADAQABAAAAYQCsU80IBrJb0qlqCi03qZK+FtgS783VKE1TVZBtDQ1sXJ9FXu6KnBvqvSajcULy+W+izqnpP14UVRYYvOd7WwLIWUoTKHfPqv3bdjwM4Bhd26POwSFyDf46yYzvMWgc="
         },
         "port": 1122
      },
      "start_times": [
         "8:00"
      ]
   }
}
```
Real-time content monitoring with Content Policies

You can monitor the traffic of certain connections in real time, and execute various actions if a certain pattern (for example, a particular command or text) appears in the command line or on the screen, or if a window with a particular title appears in a graphical protocol. Since content-monitoring is performed real-time, One Identity Safeguard for Privileged Sessions (SPS) can prevent harmful commands from being executed on your servers. SPS can also detect numbers that might be credit card numbers. The patterns to find can be defined as regular expressions. In case of ICA, RDP, and VNC connections, SPS can detect window title content.

The following actions can be performed:

- Log the event in the system logs.
- Immediately terminate the connection.
- Send an e-mail or SNMP alerts about the event.
- Store the event in the connection database of SPS.

SPS currently supports content monitoring in SSH session-shell connections, Telnet connections, RDP and Citrix ICA Drawing channels, and in VNC connections.

**NOTE:** Command, credit card and window detection algorithms use heuristics. In certain (rare) situations, they might not match the configured content. In such cases, contact our Support Team to help analyze the problem.

Real-time content monitoring in graphical protocols is not supported for Arabic and CJK languages.

To list the available Content policies, use the following command.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/content_policies
```

The following sections detail the properties of Content policy objects.

**URL**

```
GET https:<IP-address-of-SPS>/api/configuration/policies/content_policies/<object-id>
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication</td>
</tr>
<tr>
<td><strong>Cookie name</strong></td>
<td><strong>Description</strong></td>
<td><strong>Required</strong></td>
<td><strong>Values</strong></td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
<td>--------------</td>
<td>------------</td>
</tr>
<tr>
<td>token</td>
<td>token of the user</td>
<td>response, for example, a1f71d030e657634730b9e887cf59a5e56162860. For details on authentication, see <em>Authenticate to the SPS REST API</em> on page 19.</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format). |

**Sample request**

The following command lists the properties of a specific Content policy object.

```sh
curl --cookie cookies -https:<IP-address-of-SPS>/api/configuration/policies/content_policies/<object-id>
```

**Response**

The following is a sample response received, showing the properties of Content policy objects.

For details of the *meta* object, see *Message format* on page 10.

```json
{
  "body": {
    "name": "example-content-policy-window-title",
    "rules": [
      {
        "actions": {
          "log": true,
          "notify": true,
          "store_in_connection_database": true,
          "terminate": false
        },
        "event": {
          "ignore": [],
          "match": [
            "mmc.exe"
          ],
          "selection": "window_title"
        },
        "gateway_groups": []
      }
    ]
  }
}
```
### Element Type Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Top level element, the name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>rules</td>
<td>JSON object</td>
<td>Top level element, contains the configuration properties of the object.</td>
</tr>
<tr>
<td>actions</td>
<td>JSON object</td>
<td>The list of actions to perform when the Content policy matches the analyzed traffic. All actions are boolean values (true or false)</td>
</tr>
<tr>
<td>event</td>
<td>JSON object</td>
<td>Specifies the event that triggers an action.</td>
</tr>
<tr>
<td>gateway_groups</td>
<td>list</td>
<td>To apply the Content policy only for users belonging to specific groups, list those groups in the gateway_groups or remote_groups fields. If the gateway_groups or remote_groups field is set, the content policy is applied only to connections of these usergroups. For example:</td>
</tr>
<tr>
<td>remote_groups</td>
<td>list</td>
<td>To apply the Content policy only for users belonging to specific groups, list those groups in the gateway_groups or remote_groups fields. If the gateway_groups or remote_groups field is set, the content policy is applied only to connections of these usergroups. For example:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actions</td>
<td>JSON object</td>
<td>The list of actions to perform when the Content policy matches the analyzed traffic. All actions are boolean values (true or false)</td>
</tr>
<tr>
<td>log</td>
<td>boolean</td>
<td>Log the event in the system logs. Possible values: true or false</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>terminate</td>
<td>boolean</td>
<td>Immediately terminate the connection. Possible values: true or false</td>
</tr>
<tr>
<td>notify</td>
<td>boolean</td>
<td>Send an e-mail or SNMP alerts about the event. Possible values: true or false</td>
</tr>
<tr>
<td>store_in_connection_</td>
<td>boolean</td>
<td>Store the event in the connection database of SPS. Possible values: true or false</td>
</tr>
<tr>
<td>database</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>JSON object</td>
<td>Specifies the event that triggers an action.</td>
</tr>
<tr>
<td>ignore</td>
<td>list</td>
<td>A list of strings or regular expressions. SPS will perform an action if the match expression is found in the connection, unless it is listed in the ignore list. For example:</td>
</tr>
</tbody>
</table>

"ignore": [
"mmc.exe",
"cmd.exe"
]

- Use Perl Compatible Regular Expressions (PCRE).
- The following characters must be escaped using a backslash character: ' (single-quote). For example, instead of .* use .\*
- SPS uses substring search to find the expression in the content. That is, SPS finds the expression even if there is more content before or after the matching part. For example, the conf pattern will match the following texts: conf, configure, reconfigure, arcconf, and so on.
- Using complicated regular expressions or using many regular expressions will affect the performance of SPS.
- If the multiple expressions are set, SPS processes them one after the other, and stops processing the content if the first match is found, even if other expressions would also match the content. Therefore, when using multiple expressions, start with the most specific one, and add general expressions.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>match</td>
<td>list</td>
<td>A list of strings or regular expressions. SPS will perform an action if the match expression is found in the connection, unless it is listed in the ignore list. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;match&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;mmc.exe&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;cmd.exe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Use Perl Compatible Regular Expressions (PCRE).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- The following characters must be escaped using a backslash character: '(single-quote). For example, instead of .<em>' use .</em>''</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SPS uses substring search to find the expression in the content. That is, SPS finds the expression even if there is more content before or after the matching part. For example, the conf pattern will match the following texts: conf, configure, reconfigure, arcconf, and so on.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Using complicated regular expressions or using many regular expressions will affect the performance of SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- If the multiple expressions are set, SPS processes them one after the other, and stops processing the content if the first match is found, even if other expressions would also match the content. Therefore, when using multiple expressions, start with the most specific one, and add general expressions afterward.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>The type of event that you want to monitor.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- command: The commands executed in the session-shell channel of SSH connections, or in Telnet connections.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>

**CAUTION:**
During indexing, if a separate certificate is used to encrypt the upstream traffic, command detection works only if the upstream key is accessible on the machine running the indexer.

- `screen_content`: Every text that appears on the screen. For example, every text that is displayed in the terminal of SSH or Telnet connections. This includes the executed commands as well, unless echoing is turned off for the terminal.

- `creditcard`: Process every text that appears on the screen and attempt to detect credit card numbers in SSH or Telnet connections. SPS performs an action if the number of detected credit card numbers exceeds the value set as **Permitted number of credit card numbers**.

Credit card number detection is based on the Luhn algorithm and lists of known credit card number prefixes.

- `window_title`: Text appearing as window titles in case of RDP, Citrix ICA, and VNC connections. Only Windows Classic Themes are supported. Themes with rounded corners, or Windows Aero themes are not supported.

For example:

```
"selection": "window_title"
```

**Add a content policy**

To add a content policy, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new content policy.**
   
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/content_policies endpoint. You can find a
detailed description of the available parameters listed in Element.

If the POST request is successful, the response includes the key of the new policy. For example:

```json
{
    "key": "1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
    "meta": {
        "href": "/api/configuration/policies/content_policies/1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
        "parent": "/api/configuration/policies/content_policies",
        "transaction": "/api/transaction"
    }
}
```

3. Commit your changes.

For more information, see Commit a transaction on page 31.

**Modify a content policy**

To modify a content policy, you have to:

1. Open a transaction.
   
   For more information, see Open a transaction on page 29.

2. Modify the JSON object of the content policy.
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/content_policies/<policy-key> endpoint. You can find a detailed description of the available parameters listed in Element.

3. Commit your changes.

   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires author-</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**LDAP servers**

SPS can authenticate the users of the controlled SSH or RDP connections to LDAP databases.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/policies/ldap_servers

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d8030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the available LDAP server configurations.
The following command retrieves the properties of a specific LDAP server.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/ldap_servers/<object-id>
```

**Response**

The following is a sample response received when listing LDAP servers. For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "key": "354834825727acc530407",
            "meta": {
                "href": "/api/configuration/policies/ldap_servers/354834825727acc530407"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/policies/audit_policies",
        "href": "/api/configuration/policies/ldap_servers",
        "last": "/api/configuration/policies/usermapping_policies",
        "next": "/api/configuration/policies/signing_cas",
        "parent": "/api/configuration/policies",
        "previous": "/api/configuration/policies/indexing",
        "transaction": "/api/transaction"
    }
}
```

When retrieving the endpoint of a specific LDAP server, the response is the following.

```json
{
    "key": "posix-simple",
    "body": {
        "name": "posix-simple",
        "schema": {
            "selection": "posix",
            "membership_check": {
                "enabled": true,
                "member_uid_attribute": "memberUid"
            },
            "memberof_check": {
                
        }"memberof_check": {
```
```json
{
  "enabled": true,
  "memberof_user_attribute": "memberOf",
  "memberof_group_objectclass": "groupOfNames"
},
  "username_attribute": "uid",
  "user_dn_in_groups": []
},
  "servers": [
    {
      "host": {
        "selection": "ip",
        "value": "10.110.0.1"
      },
      "port": 389
    }
  ],
  "user_base_dn": "ou=People,dc=example,dc=com",
  "group_base_dn": "ou=Groups,dc=example,dc=com",
  "bind_dn": null,
  "bind_password": null,
  "memberof_attribute": null,
  "encryption": {
    "selection": "disabled"
  },
  "publickey_attribute": "sshPublicKey",
  "generated_publickey_attribute": null
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the LDAP server configuration.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the properties of the LDAP server.</td>
</tr>
<tr>
<td>user_base_dn</td>
<td>string</td>
<td>Name of the DN to be used as the base of queries regarding users.</td>
</tr>
</tbody>
</table>

**NOTE:** You must fill in this field. It is OK to use the same value for `user_base_dn` and `group_base_dn`. However, note that specifying a sufficiently narrow base for the LDAP subtrees where users and groups are stored can speed up LDAP operations.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group_base_dn</td>
<td>string</td>
<td>Name of the DN to be used as the base of queries regarding groups.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>NOTE:</strong> You must fill in this field. It is OK to use the same value for user_base_dn and group_base_dn. However, note that specifying a sufficiently narrow base for the LDAP subtrees where users and groups are stored can speed up LDAP operations.</td>
</tr>
<tr>
<td>bind_dn</td>
<td>string</td>
<td>The Distinguished Name that SPS should use to bind to the LDAP directory.</td>
</tr>
<tr>
<td>bind_password</td>
<td>string</td>
<td>References the password SPS uses to authenticate on the server. You can configure passwords at the /api/configuration/passwords/ endpoint. To modify or add a password, use the value of the returned key as the value of the password element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>encryption</td>
<td>Top level item</td>
<td>Configuration settings for encrypting the communication between SPS and the LDAP server.</td>
</tr>
<tr>
<td>generated_publickey_attribute</td>
<td>string</td>
<td>Set this element to null if you use passwords to authenticate. Configure this element if you want SPS to generate server-side encryption keys on-the-fly, and store them in a separate attribute on the LDAP server.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Top level element, the name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>publickey_attribute</td>
<td>string</td>
<td>Set this element to null if you use passwords to authenticate. The name of the LDAP attribute that stores the public keys of the users.</td>
</tr>
<tr>
<td>schema</td>
<td>Top level item</td>
<td>Contains the configuration settings for the AD schema.</td>
</tr>
<tr>
<td>servers</td>
<td>Top level list</td>
<td>Contains the addresses and ports of the LDAP servers.</td>
</tr>
<tr>
<td>Elements of encryption</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the type of encryption SPS uses to communicate with the LDAP server. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- disabled                                                                                     The communication is not encrypted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ssl                                                                                           TLS/SSL encryption. To use a TLS-encrypted with certificate verification to connect to the LDAP server, use the full domain name (for example ldap.example.com) as the server address, otherwise the certificate verification might fail. The name of the LDAP server must appear in the Common Name of the certificate. TLS-encrypted connection to Microsoft Active Directory is supported only on Windows 2003 Server and newer platforms. Windows 2000 Server is not supported.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- starttls                                                                                     Opportunistic TLS.</td>
</tr>
</tbody>
</table>

<p>| client_authentication  | Top level item | Must be used with the selection child element. Configures the X.509 certificate SPS uses to authenticate on the LDAP server.                                                                                  |
| enabled                | boolean        | Must be used with the client-authentication parent element. Set to true if the LDAP server requires mutual authentication.                                                                                     |
| x509_identity          | string         | Must be used if the enabled element is set to true. References the identifier of the X.509 certificate stored on SPS. You can configure X.509 certificates at the /api/configuration/x509/ endpoint. |</p>
<table>
<thead>
<tr>
<th>Elements of encryption</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>To modify or add an X.509 host certificate, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the type of encryption SPS uses to communicate with the LDAP server. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- The communication is not encrypted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ssl</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- TLS/SSL encryption. To use a TLS-encrypted with certificate verification to connect to the LDAP server, use the full domain name (for example ldap.example.com) as the server address, otherwise the certificate verification might fail. The name of the LDAP server must appear in the Common Name of the certificate.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- TLS-encrypted connection to Microsoft Active Directory is supported only on Windows 2003 Server and newer platforms. Windows 2000 Server is not supported.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- starttls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Opportunistic TLS.</td>
</tr>
<tr>
<td>server_certificate_check</td>
<td>Top level item</td>
<td>Must be used with the enabled child element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Configuration settings for verifying the LDAP server's certificate.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Must be used with the server_certificate_check parent element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set to true to verify the LDAP server's certificate using the certificate of a Certificate Authority (CA).</td>
</tr>
</tbody>
</table>
### Elements of encryption

<table>
<thead>
<tr>
<th><strong>server_certificate_ca</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td></td>
<td>Must be used if the enabled element is set to true. The certificate of the CA.</td>
</tr>
</tbody>
</table>

### Elements of servers

<table>
<thead>
<tr>
<th><strong>host</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Top level item</td>
<td></td>
<td>Contains the address of the LDAP server.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>selection</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
</table>
| string        |          | Defines the address type (IP or domain name). Possible values are:  
|               |          |   - fqdn  
|               |          |     The LDAP server address is provided as a fully qualified domain name.  
|               |          |   - ip  
|               |          |     The LDAP server address is provided as an IP address. |

<table>
<thead>
<tr>
<th><strong>value</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td></td>
<td>The address of the LDAP server.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>port</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td></td>
<td>The port of the LDAP server.</td>
</tr>
</tbody>
</table>

### Elements of schema

<table>
<thead>
<tr>
<th><strong>selection</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
</table>
| string        |          | Configures which LDAP schema to use: AD or POSIX. Possible values are:  
|               |          |   - ad: Microsoft Active Directory server. For details and examples, see LDAP servers.  
|               |          |   - posix: The server uses the POSIX LDAP scheme.  
|               |          |     Must be used with the member_uid_attribute and username_attribute elements. For details and examples, see LDAP servers. |

<table>
<thead>
<tr>
<th><strong>membership_check</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>enabled</strong></td>
<td>boolean</td>
<td>POSIX: Enables POSIX primary and supplementary group membership checking. AD: Enables Active Directory specific non-primary group membership checking.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>nested_</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td></td>
<td>Must be used if the selection element is set to</td>
</tr>
<tr>
<td>Elements of schema</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>groups</td>
<td>ad.</td>
<td>Enable nested groups allows AD nested group support.</td>
</tr>
<tr>
<td>member_uid_attribute</td>
<td>string</td>
<td>Must be used if the value of the selection element is set to posix. The POSIX group membership attribute name is the name of the attribute in a posixGroup group object, which lists the plain usernames that are members of the group. These groups are usually referred to as supplementary groups of the referred user. Can be null.</td>
</tr>
<tr>
<td>memberof_check</td>
<td>Top level element</td>
<td>The Enable checking for group DNs in user objects setting allows checking a configurable attribute in the user object. This attribute contains a list of group DNs the user is additionally a member of. This user attribute is usually memberOf.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>To enable memberof_check, set it to true.</td>
</tr>
<tr>
<td>memberof_user_attribute</td>
<td>string</td>
<td>Must be used if the memberof_check is set to true. The name of the user attribute (for example, memberOf) that contains the group DNs.</td>
</tr>
<tr>
<td>username_attribute</td>
<td>string</td>
<td>Must be used if the selection element is set to posix. Attribute name of the username (user ID).</td>
</tr>
<tr>
<td>user_dn_in_groups</td>
<td>Top level list</td>
<td>Add object_class / attribute pairs. SPS will search for the user DN in the group's attribute defined here. If it finds the user DN there, SPS considers the user the member of that group.</td>
</tr>
</tbody>
</table>

For example:

```json
"user_dn_in_groups": [
    {
        "object_class": "groupOfNames",
        "attribute": "member"
    },
    {
        "object_class": "groupOfUniqueNames",
        "attribute": "uniqueMember"
    }
]```
Elements of schema | Type | Description
--- | --- | ---
object_class | string | Consider groups of this objectClass.
attribute | string | Name of the group attribute which contains the user DN.

Example: Configure a POSIX server without communication encryption

```json
{
  "name": "<name-of-ldap-policy>",
  "schema": {
    "selection": "posix",
    "username_attribute": "<uid>",
    "membership_check": {
      "enabled": true,
      "member_uid_attribute": "<memberUid>"
    },
    "memberof_check": {
      "enabled": true,
      "memberof_user_attribute": "<memberOf>",
      "memberof_group_objectclass": "<groupOfNames>"
    },
    "user_dn_in_groups": [
      {
        "object_class": "<groupOfNames>",
        "attribute": "<member>"
      },
      {
        "object_class": "<groupOfUniqueNames>",
        "attribute": "<uniqueMember>"
      }
    ]
  },
  "servers": [
    {
      "host": {
        "selection": "fqdn",
        "value": "<server-name>"
      }
    }
  ]
}
```
Example: Configure a Microsoft Active Directory server with mutual authentication, and the verification of the server's X.509 certificate

```json
{
    "name": "<name-of-ldap-policy>",
    "schema": {
        "selection": "ad",
        "membership_check": {
            "enabled": true,
            "nested_groups": false
        },
        "memberof_check": {
            "enabled": true,
            "memberof_user_attribute": "<memberOf>"
        },
        "user_dn_in_groups": [
            {
                "object_class": "<groupOfNames>",
                "attribute": "<member>"
            }
        ]
    }
}
```
{ "object_class": "<groupOfUniqueNames>",
    "attribute": "<uniqueMember>"
}, "servers": [ {
    "host": {
        "selection": "ip",
        "value": "<server-ip>"
    },
    "port": <server-port>
}, "user_base_dn": "<basedn>",
"group_base_dn": "<basedn>",
"bind_dn": "<binddn>",
"bind_password": "<key-of-password>",
"encryption": {
    "client_authentication": {
        "enabled": true,
        "x509_identity": "<key-of-cert>
    },
    "selection": "starttls",
    "server_certificate_check": {
        "enabled": true,
        "server_certificate_ca": "<ca-cert>
    }
},
"publickey_attribute": "<sshPublicKey>",
"generated_publickey_attribute": null
}

CA certificates

CA certificates must not contain any metadata. SPS uses only the key part of the certificate.

To use a certificate with the SPS API, remove all metadata, and substitute line breaks with \n.

The following is an example certificate, as used on the SPS web interface:
The same certificate, as accepted by the SPS API:

```
"certificate": "-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCCQDc5360b5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC
Q0ExEDAOBgNVBAoTB90ud968a5x8xADAoBGBAEDBT81vcm9udGx8EDAoBGBAED
B03hbGfiaXQfjAUBgNBAsTDURvY3VtZw50YXRpb24x6EDAoBGBAEDBT81vcm9udGx
aXQfjAUBgNBAsTDURvY3VtZw50YXRpb24x6EDAoBGBAEDBT81vcm9udGx

The same certificate, as accepted by the SPS API:

```

Add an LDAP server

To add an LDAP server, you have to:
1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new LDAP server.**
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/ldap_servers endpoint. You can find a detailed description of the available parameters listed in **Element**.
   If the POST request is successful, the response includes the key of the new LDAP server. For example:

   ```
   {
     "key": "f9f9783c-1e28-4ce8-a650-fc4c7311ac52",
     "meta": {
       "href": "/api/configuration/policies/ldap_servers/f9f9783c-1e28-4ce8-a650-fc4c7311ac52",
       "parent": "/api/configuration/policies/ldap_servers",
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

**Modify an LDAP server**
To modify the configuration of an LDAP server, you have to:

1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the LDAP server.**
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/ldap_servers/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in **Element**.

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

**Status and error codes**
The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
### Signing CA policies

SPS can generate the server-side certificates on the fly. This technique is used for example in SSL-encrypted RDP sessions, RDP sessions that use Network Level Authentication (CredSSP), or SSH connections that use X.509-based authentication.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/policies/signing_cas
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists the configured signing Certificate Authorities (CAs).

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/signing_cas
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/signing_cas/<object-id>
```

Response

The following is a sample response received when listing signing CAs.

For details of the meta object, see Message format on page 10.

```
{
   "items": [
      {
         "key": "991699365727ac4eb4606",
         "meta": {
            "href": "/api/configuration/policies/signing_cas/991699365727ac4eb4606"
         }
      }
   ],
   "first": "/api/configuration/policies/audit_policies",
   "href": "/api/configuration/policies/signing_cas",
   "last": "/api/configuration/policies/usermapping_policies",
   "next": "/api/configuration/policies/ticketing_policies",
   "parent": "/api/configuration/policies",
   "previous": "/api/configuration/policies/ldap_servers",
   "transaction": "/api/transaction"
}
```

When retrieving the endpoint of a specific signing CA, the response is the following.

```
{
   "body": {
      "ca": {
         "key": "55b2419c-f94f-4836-9c0b-bc3796b6f556",
         "meta": {
            "href": "/api/configuration/x509/55b2419c-f94f-4836-9c0b-bc3796b6f556"
         }
      }
   }
}
```
"name": "API_CA",
"key": "991699365727ac4eb4606",
"meta": {
  "first": "/api/configuration/policies/signing_cas/991699365727ac4eb4606",
  "href": "/api/configuration/policies/signing_cas/991699365727ac4eb4606",
  "last": "/api/configuration/policies/signing_cas/991699365727ac4eb4606",
  "next": null,
  "parent": "/api/configuration/policies/signing_cas",
  "previous": null,
  "transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the signing CA.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the properties of the signing CA.</td>
</tr>
<tr>
<td>ca</td>
<td>string</td>
<td>References the identifier of the signing CA’s X.509 certificate. You can configure certificates at the /api/configuration/x509/ endpoint. To modify or add an X.509 certificate, use the value of the returned key as the value of the x509_identity element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the signing CA. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
</tbody>
</table>

**Add a signing CA**

To add a signing CA, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create a signing CA**
   
   Have the value of the key element of a valid X.509 CA certificate stored on SPS.

3. **Create the JSON object for the new signing CA.**
   
   Use the X.509 certificate's key as the value of the ca element for the signing CA. You can find a detailed description of the available parameters listed in Element.
POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/signing_cas endpoint. If the POST request is successful, the response includes the key of the new signing CA. For example:

```json
{
    "key": "325768b5-5b85-467d-8e30-e2b57d0869c8",
    "meta": {
        "href": "/api/configuration/policies/signing_cas/325768b5-5b85-467d-8e30-e2b57d0869c8",
        "parent": "/api/configuration/policies/signing_cas",
        "transaction": "/api/transaction"
    }
}
```

4. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

**Modify a signing CA**

To modify a signing CA, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the signing CA.**

   Use the X.509 certificate's key as the value of the `ca` element for the signing CA. You can find a detailed description of the available parameters listed in [Element](#).

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/signing_cas/<key-of-the-object> endpoint.

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes](#) on page 37.
### HTTP response code

<table>
<thead>
<tr>
<th>Status/Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201 Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400 InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>400 Bad Request</td>
<td>The referenced certificate is not a valid CA certificate.</td>
</tr>
<tr>
<td>401 Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403 Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404 NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Time policy

The time policy determines the timeframe when the users are permitted to access a particular channel. To list the available Time policies, use the following command.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/time_policies
```

The following sections detail the properties of Time policy objects.

### URL

**GET** https://<IP-address-of-SPS>/api/configuration/policies/time_policies/<object-id>

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example,</td>
</tr>
</tbody>
</table>
### Sample request

The following request command lists the properties of a specific Time policy object.

```
curl --cookie cookies -https:<IP-address-of-SPS>/api/configuration/policies/time_policies/<object-id>
```

### Response

The following is a sample response received, showing the properties of Content policy objects.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "Fri": [
            ["0:00", "23:59"]
        ],
        "Mon": [
            ["0:00", "23:59"]
        ],
        "Sat": [
            ["0:00", "23:59"]
        ],
        "Sun": [  
```

---

**Cookie**

<table>
<thead>
<tr>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td></td>
<td>a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
```json
{
    "name": "7x24",
    "Fri": [
        ["0:00", "23:59"]
    ],
    "Thu": [
        ["0:00", "23:59"]
    ],
    "Tue": [
        ["0:00", "23:59"]
    ],
    "Wed": [
        ["0:00", "23:59"]
    ]
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Top level element, the name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>Fri</td>
<td>list</td>
<td>A list of intervals for the day when the users are allowed to access the connection. Use the hh:mm format. If the users are not allowed to access the connection for this day, use an empty list. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Sat&quot;: [],</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To allow access for the whole day, use 0:00 for the starting time, and 23:59 for the end. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Wed&quot;: [ [&quot;0:00&quot;, &quot;23:59&quot;] ] You can list multiple intervals for a day, for example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Wed&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[</td>
</tr>
</tbody>
</table>
|         |       |         "8:00",
|         |       |         "18:00" |
```
### Trusted Certificate Authorities

SPS can check the validity of certificates using the certificates and certificate-revocation lists of trusted certificate authorities (CAs) that issued the certificates.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/policies/trusted_ca_lists
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists the trusted CAs.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/trusted_ca_lists
```

The following command retrieves the properties of a specific CA.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/trusted_ca_lists/<policy-id>
```

Response

The following is a sample response received when listing trusted CAs.

For details of the meta object, see Message format on page 10.

```json
{
   "items": [
      {
         "key": "12269547065727ad6e79d9e",
         "meta": {
            "href": "/api/configuration/policies/trusted_ca_lists/12269547065727ad6e79d9e"
         }
      }
   ],
   "meta": {
      "first": "/api/configuration/policies/audit_policies",
      "href": "/api/configuration/policies/trusted_ca_lists",
      "last": "/api/configuration/policies/usermapping_policies",
      "next": "/api/configuration/policies/user_databases",
      "parent": "/api/configuration/policies",
      "previous": "/api/configuration/policies/time_policies",
      "transaction": "/api/transaction"
   }
}
```

When retrieving the endpoint of a specific CA, the response is the following.

```json
{
   "body": {
      "authorities": [
         {
            "certificate": "+cert",
            "crl": "+url-of-revocation-list"
         }
      ]
   }
}
```
```json
"dn_check": {
  "altEmailAddress": "<altEmail>",
  "c": "<country>",
  "cn": "<commonName>",
  "emailAddress": "<email>",
  "l": "<localityName>",
  "o": "<orgName>",
  "ou": "<orgUnitName>",
  "st": "<stateOrProvince>"
},
"dns_lookup": false,
"name": "<ca-name>",
"strict_hostcheck": true
},
"key": "12269547065727ad6e79d9e",
"meta": {
  "first": "/api/configuration/policies/trusted_ca_lists/12269547065727ad6e79d9e",
  "href": "/api/configuration/policies/trusted_ca_lists/12269547065727ad6e79d9e",
  "last": "/api/configuration/policies/trusted_ca_lists/12269547065727ad6e79d9e",
  "next": null,
  "parent": "/api/configuration/policies/trusted_ca_lists",
  "previous": null,
  "transaction": "/api/transaction"
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the CA.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>Contains the properties of the CA.</td>
</tr>
<tr>
<td></td>
<td>list (string)</td>
<td></td>
</tr>
<tr>
<td>authorities</td>
<td>Top level</td>
<td>Contains the certificates and the Certificate Revocation Lists (CLR) of the</td>
</tr>
<tr>
<td></td>
<td>list</td>
<td>trusted CAs. You can add multiple certificate and CRL pairs.</td>
</tr>
<tr>
<td>certificate</td>
<td>string</td>
<td>The certificate of the trusted CA.</td>
</tr>
<tr>
<td>crl</td>
<td>string</td>
<td>The URL of the Certificate Revocation List of the CA.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>dn_check</td>
<td>Top level item</td>
<td>Certificates are only accepted if their content matches the configured values.</td>
</tr>
<tr>
<td>altEmailAddress</td>
<td>string</td>
<td>The certificate is only accepted if its alternative e-mail address matches the value of the altEmailAddress element.</td>
</tr>
<tr>
<td>c</td>
<td>string</td>
<td>The certificate is only accepted if its country matches the value of the c element.</td>
</tr>
<tr>
<td>cn</td>
<td>string</td>
<td>The certificate is only accepted if its common name matches the value of the cn element.</td>
</tr>
<tr>
<td>emailAddress</td>
<td>string</td>
<td>The certificate is only accepted if its e-mail address matches the value of the emailAddress element.</td>
</tr>
<tr>
<td>l</td>
<td>string</td>
<td>The certificate is only accepted if its locality matches the value of the l element.</td>
</tr>
<tr>
<td>o</td>
<td>string</td>
<td>The certificate is only accepted if its organization name matches value of the o element.</td>
</tr>
<tr>
<td>ou</td>
<td>string</td>
<td>The certificate is only accepted if its organization unit name matches value of the ou element.</td>
</tr>
<tr>
<td>st</td>
<td>string</td>
<td>The certificate is only accepted if its state or province matches value of the st element.</td>
</tr>
<tr>
<td>dns_lookup</td>
<td>boolean</td>
<td>Set to true to use the domain name server set on the /api/-configuration/network/naming endpoint to resolve the hostnames and IP addresses for certificate validation. If you have enabled strict_hostcheck, you probably want to enable this option as well.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the trusted CA. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
</tbody>
</table>
**Uploading CA certificates**

SPS uses only the key part of the CA certificate.

To use a certificate with the SPS API, remove all data, and substitute line breaks with \n.

The following is an example certificate, as used on the SPS web interface:

```
-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCCQDc536Ob5tPTQANBgqhkijG9w0BAQUFADCBjzELMAKGA1UEBhMCQUExE
DA08B6NBVAcTB1Rvcm9udG8xEDAO8B8NBVAoT
B03hGfiXqXfIAUBgNVBAsTDURyY3VtZ250YXRpb2c4EDAO8B8NBVAoT
aXiQdAEBgqhkijG9w0BCQEWEWhdGFpEbIxEyhxhym10Lnhm1MB4XDE2MDQyMjE2
MDAynLoXDTExMDQyMjE2MDAynL0wYx8CzAJa8GvBAYTAkBNMRAwDgYDVQQEEx
bnRhcm1vMRAwDgYDVQQHEwdb3JvbnRvMRo8EDAO8B8NBVAoT
vQQLwE1b2N1bWVudGF0aW9ufnRhcm1vMRAwDgYDVQQHEwdb3JvbnRvMRo8EDAO8B8NBVAoT
3mVzZkvtD61CXbezAVArDl6WnsRy0xH8WEmFwzEpwxF9D9Y8N1zDkod7Hnc
X5KlVl0+Z6bBHf7/7whbNjsyDwNuLr0ByTuVPm6wdA83k3Fy2gYNk7RC08bRfBQTX
hJvFujU5u8HvFHqTba4d1k5v0qepfMNCAwEAATANBgkqhkiG9w0BAQUFBAADctsA
R5Di4vOSHkGki13chC2VMnxP2rhTnheh6El0+sfnQpDxrAttnqV4TndNa+FvP
ABzkpmC4hJAsjMLyuq2hJe6+5mzHhRuxMaC6+x+fnYvcaTQ571Rdql2/4oJgEyuy
0jQr7/nnoVMD13YtxDcFXqZhn1ctcBEEh5P5umSmep4mNhY8G9UHJfF3PfovEyG7XDB
uNaUo5wSS5wL5IuF9tq9e86S6DE7c8rBfQYVSoW9jPnUMxzR8xtRSEG70Jc
N5+41mYtFnxNgSk20HypF0ODf2g7FxFxW0/NTxamUF15FwLPzC030v11/Fzo7mD
qYyyD1d898UEYz+a9qD/A==
-----END CERTIFICATE-----
```

The same certificate, as accepted by the SPS API:

```
"certificate": "-----BEGIN CERTIFICATE-----

\nMIIDnDCCAoQCCQDc536Ob5tPTQANBgqhkijG9w0BAQUFADCBjzELMAKGA1UEBhMCQUExE
DA08B6NBVAcTB1Rvcm9udG8xEDAO8B8NBVAoT
B03hGfiXqXfIAUBgNVBAsTDURyY3VtZ250YXRpb2c4EDAO8B8NBVAoT
aXiQdAEBgqhkijG9w0BCQEWEWhdGFpEbIxEyhxhym10Lnhm1MB4XDE2MDQyMjE2
MDAynLoXDTExMDQyMjE2MDAynL0wYx8CzAJa8GvBAYTAkBNMRAwDgYDVQQEEx
bnRhcm1vMRAwDgYDVQQHEwdb3JvbnRvMRo8EDAO8B8NBVAoT
vQQLwE1b2N1bWVudGF0aW9ufnRhcm1vMRAwDgYDVQQHEwdb3JvbnRvMRo8EDAO8B8NBVAoT
3mVzZkvtD61CXbezAVArDl6WnsRy0xH8WEmFwzEpwxF9D9Y8N1zDkod7Hnc
X5KlVl0+Z6bBHf7/7whbNjsyDwNuLr0ByTuVPm6wdA83k3Fy2gYNk7RC08bRfBQTX
hJvFujU5u8HvFHqTba4d1k5v0qepfMNCAwEAATANBgkqhkiG9w0BAQUFBAADctsA
R5Di4vOSHkGki13chC2VMnxP2rhTnheh6El0+sfnQpDxrAttnqV4TndNa+FvP
ABzkpmC4hJAsjMLyuq2hJe6+5mzHhRuxMaC6+x+fnYvcaTQ571Rdql2/4oJgEyuy
0jQr7/nnoVMD13YtxDcFXqZhn1ctcBEEh5P5umSmep4mNhY8G9UHJfF3PfovEyG7XDB
uNaUo5wSS5wL5IuF9tq9e86S6DE7c8rBfQYVSoW9jPnUMxzR8xtRSEG70Jc
N5+41mYtFnxNgSk20HypF0ODf2g7FxFxW0/NTxamUF15FwLPzC030v11/Fzo7mD
qYyyD1d898UEYz+a9qD/A==
-----END CERTIFICATE-----
```

"certificate": "-----BEGIN CERTIFICATE-----

\nMIIDnDCCAoQCCQDc536Ob5tPTQANBgqhkijG9w0BAQUFADCBjzELMAKGA1UEBhMCQUExE
DA08B6NBVAcTB1Rvcm9udG8xEDAO8B8NBVAoT
B03hGfiXqXfIAUBgNVBAsTDURyY3VtZ250YXRpb2c4EDAO8B8NBVAoT
aXiQdAEBgqhkijG9w0BCQEWEWhdGFpEbIxEyhxhym10Lnhm1MB4XDE2MDQyMjE2
MDAynLoXDTExMDQyMjE2MDAynL0wYx8CzAJa8GvBAYTAkBNMRAwDgYDVQQEEx
bnRhcm1vMRAwDgYDVQQHEwdb3JvbnRvMRo8EDAO8B8NBVAoT
vQQLwE1b2N1bWVudGF0aW9ufnRhcm1vMRAwDgYDVQQHEwdb3JvbnRvMRo8EDAO8B8NBVAoT
3mVzZkvtD61CXbezAVArDl6WnsRy0xH8WEmFwzEpwxF9D9Y8N1zDkod7Hnc
X5KlVl0+Z6bBHf7/7whbNjsyDwNuLr0ByTuVPm6wdA83k3Fy2gYNk7RC08bRfBQTX
hJvFujU5u8HvFHqTba4d1k5v0qepfMNCAwEAATANBgkqhkiG9w0BAQUFBAADctsA
R5Di4vOSHkGki13chC2VMnxP2rhTnheh6El0+sfnQpDxrAttnqV4TndNa+FvP
ABzkpmC4hJAsjMLyuq2hJe6+5mzHhRuxMaC6+x+fnYvcaTQ571Rdql2/4oJgEyuy
0jQr7/nnoVMD13YtxDcFXqZhn1ctcBEEh5P5umSmep4mNhY8G9UHJfF3PfovEyG7XDB
uNaUo5wSS5wL5IuF9tq9e86S6DE7c8rBfQYVSoW9jPnUMxzR8xtRSEG70Jc
N5+41mYtFnxNgSk20HypF0ODf2g7FxFxW0/NTxamUF15FwLPzC030v11/Fzo7mD
qYyyD1d898UEYz+a9qD/A==
-----END CERTIFICATE-----
```
Add a trusted CA

To add a trusted CA, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new trusted CA.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/trusted_ca_lists endpoint. You can find a detailed description of the available parameters listed in Element. If the POST request is successful, the response includes the key of the new trusted CA. For example:

   ```
   {
     "key": "becc17b1-e876-4443-b22e-a3baf7825e55",
     "meta": {
       "href": "/api/configuration/policies/trusted_ca_lists/becc17b1-e876-4443-b22e-a3baf7825e55",
       "parent": "/api/configuration/policies/trusted_ca_lists",
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

Modify a trusted CA

To modify a trusted CA, you have to:
1. **Open a transaction.**
   
   For more information, see [Open a transaction on page 29](#).

2. **Modify the JSON object of the trusted CA.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/policies/trusted_ca_lists/<key-of-the-object>` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction on page 31](#).

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes on page 37](#).

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Local user databases

Local User Databases are available for RDP, SSH and Telnet protocols, and can be used to authenticate the clients to credentials that are locally available on SPS. Such credentials include passwords and public keys. Local User Databases are most commonly used in inband gateway authentication scenarios.
URL

GET https://<IP-address-of-SPS>/api/configuration/policies/user_databases

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists local user databases.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/user_databases
```

The following command retrieves the properties of a specific local user database.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/user_databases/<object-id>
```

Response

The following is a sample response received when listing local user databases.

For details of the meta object, see Message format on page 10.

```
{
   "items": [
      {
         "key": "8235074425707e306abf39",
         "meta": {
            "href": "/api/configuration/policies/user_databases/8235074425707e306abf39"
         }
      }
   ]
}```
When retrieving the endpoint of a specific local user database, the response is the following.

```json
{
  "body": {
    "name": "<name-of-the-user-database>",
    "users": [
      {
        "passwords": [
          {
            "key": "ad55822d-fa28-45aa-bca4-220074f770e1",
            "meta": {
              "href": "/api/configuration/passwords/ad55822d-fa28-45aa-bca4-220074f770e1"
            }
          }
        ],
        "public_keys": [
          {
            "selection": "rsa",
            "value": "<public-key>"
          }
        ],
        "username": "<username>"
      }
    ],
    "key": "8235074425707e306abf39",
    "meta": {
      "first": "/api/configuration/policies/user_databases/8235074425707e306abf39",
      "href": "/api/configuration/policies/user_databases/8235074425707e306abf39",
      "last": "/api/configuration/policies/user_databases/8235074425707e306abf39",
      "next": null,
      "previous": null,
      "transaction": "/api/configuration/policies/transaction"
    }
  }
}
```
"parent": "/api/configuration/policies/user_databases",
"previous": null,
"transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the local user database.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>Contains the properties of the local user database.</td>
</tr>
<tr>
<td></td>
<td>element</td>
<td>string</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the local user database. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>users</td>
<td>Top level</td>
<td>Contains the credentials (password, key) of each configured user.</td>
</tr>
<tr>
<td></td>
<td>list</td>
<td>string</td>
</tr>
<tr>
<td>passwords</td>
<td>Top level</td>
<td>References the password of the user. You can configure passwords at the</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td>/api/configuration/passwords/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a password, use the value of the returned key as the value of the password element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>public_keys</td>
<td>Top level</td>
<td>Contains the public keys of the user.</td>
</tr>
<tr>
<td></td>
<td>list</td>
<td>string</td>
</tr>
<tr>
<td>username</td>
<td>Top level</td>
<td>Name of the user.</td>
</tr>
<tr>
<td></td>
<td>list, string</td>
<td>string</td>
</tr>
</tbody>
</table>

**Elements of public_keys**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
<td>Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• rsa</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains an RSA key.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• dss</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains a DSS key.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The public key.</td>
</tr>
</tbody>
</table>
Examples:

Configure password authentication only for test_user. (New passwords can only be provided using the web interface of SPS.)

```json
{
   "name": "<name-of-the-user-database>",
   "users": [
      {
         "certificates": [],
         "passwords": [
            "ad55822d-fa28-45aa-bca4-220074f770e1"
         ],
         "public_keys": [],
         "username": "test_user"
      }
   ]
}
```

Configure two possible X.509 certificates for test_user, and no other authentication options.

```json
{
   "name": "<name-of-the-user-database>",
   "users": [
      {
         "certificates": [
            "<cert1>",
            "<cert2>"
         ],
         "passwords": [],
         "public_keys": [],
         "username": "test_user"
      }
   ]
}
```

Add a local user database

To add a local user database, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.
2. **Create the JSON object for the new local user database.**

   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/policies/user_databases` endpoint. You can find a detailed description of the available parameters listed in *Element*.

   If the POST request is successful, the response includes the key of the new local user database. For example:

   ```json
   {
       "key": "c4e60325-971a-44bc-ac01-e353dc6320d6",
       "meta": {
           "href": "https://<IP-address-of-SPS>/api/configuration/policies/user_databases/c4e60325-971a-44bc-ac01-e353dc6320d6",
           "parent": "https://<IP-address-of-SPS>/api/configuration/policies/user_databases",
           "transaction": "https://<IP-address-of-SPS>/api/transaction"
       }
   }
   ```

3. **Commit your changes.**

   For more information, see *Commit a transaction* on page 31.

**Modify a local user database**

To modify a local user database, you have to:

1. **Open a transaction.**

   For more information, see *Open a transaction* on page 29.

2. **Modify the JSON object of the local user database.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/policies/user_databases/<key-of-the-object>` endpoint. You can find a detailed description of the available parameters listed in *Element*.

3. **Commit your changes.**

   For more information, see *Commit a transaction* on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see *Application level error codes* on page 37.
### Code Description Notes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### User lists

User lists are local white- or blacklists of usernames that allow fine-control over who can access a connection or a channel.

**NOTE:** User lists on SPS cannot prevent a user from accessing the server from a local terminal.

You can use user lists when configuring gateway_groups or remote_groups in the allowed_for element of channel policies. For more information on configuring channel policies, see Channel policy on page 354.

To use this option, you must also configure web gateway authentication in the connection policy, or client-side gateway authentication back-end in the authentication policy.

### URL

```
GET https://<IP-address-of-SPS>/api/configuration/policies/userlists
```

### Cookies

<table>
<thead>
<tr>
<th>Cookie Name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860.</td>
</tr>
<tr>
<td>Cookie name</td>
<td>Description</td>
<td>Required</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
</tr>
</tbody>
</table>

For details on authentication, see [Authenticate to the SPS REST API on page 19](#).

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists the user lists created on SPS.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/userlists
```

The following command retrieves the properties of a specific list.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/userlists/<key-id>
```

### Response

The following is a sample response received when retrieving the user lists.

For details of the meta object, see [Message format on page 10](#).

The keys with negative ID values are the default user lists of SPS.

```json
{
  "meta": {
    "first": "/api/configuration/policies/audit_policies",
    "href": "/api/configuration/policies/userlists",
    "last": "/api/configuration/policies/usermapping_policies",
    "next": "/api/configuration/policies/usermapping_policies",
    "parent": "/api/configuration/policies",
    "previous": "/api/configuration/policies/user_databases",
    "transaction": "/api/transaction"
  },
  "items": [
    {
      "key": "-1",
      "meta": {
        "href": "/api/configuration/policies/userlists/-1"
      }
    }
  ]
}
```
When retrieving the endpoint of a specific user list, the response is the following.

```json
{
  "body": {
    "allow": "no_user",
    "except": [
      "root"
    ],
    "name": "root_only"
  },
  "key": "-4",
  "meta": {
    "href": "/api/configuration/policies/userlists/-4"
  }
},
"key": "-4",
"meta": {
  "first": "/api/configuration/policies/userlists/-1",
  "href": "/api/configuration/policies/userlists/-4",
  "last": "/api/configuration/policies/userlists/20088200245706af301b1ba",
  "next": "/api/configuration/policies/userlists/20088200245706af301b1ba",
  "prev": null
}
}```
"parent": "/api/configuration/policies/userlists",
"previous": "/api/configuration/policies/userlists/-3",
"transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the user list</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the user policy.</td>
</tr>
<tr>
<td>allow</td>
<td>string</td>
<td>The default policy of the user list. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- all_users creates a blacklist, where every user is permitted, except the ones listed in the except field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- no_user creates a whitelist, where only the users listed in the except field are allowed access.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the user list.</td>
</tr>
<tr>
<td>except</td>
<td>list</td>
<td>The usernames added to the list.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes](#) on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Examples

The following defines a blacklist called no_root that permits every username except root.

```json
{
  "allow": "all_users",
  "except": [
    "root"
  ],
  "name": "no_root"
}
```

The following defines a whitelist called my_list that permits only the permitted_user1 and permitted_user2 usernames.

```json
{
  "allow": "no_user",
  "except": [
    "permitted_user1",
    "permitted_user2"
  ],
  "name": "no_root"
}
```

Add a user list

To add a user list, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new user list.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/userlists endpoint. You can find a detailed description of the available parameters listed in Element.

   If the POST request is successful, the response includes the key of the new user list. For example:

   ```json
   {
     "key": "321314dc-eca0-4e97-b445-0612fedc0165",
     "meta": {
       "href": "/api/configuration/policies/userlists/321314dc-eca0-4e97-
   ```
Modify a user list

To modify a user list, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the user list.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/userlists/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.
HTTP connections

List of endpoints for configuring the policies, options and connection rules of HTTP connections.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/http

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the available settings for configuring for HTTP connections.

```shell
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http
```
Response

The following is a sample response received when listing the configuration settings. For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "authentication_policies",
      "meta": {
        "href": "/api/configuration/http/authentication_policies"
      }
    },
    {
      "key": "channel_policies",
      "meta": {
        "href": "/api/configuration/http/channel_policies"
      }
    },
    {
      "key": "connections",
      "meta": {
        "href": "/api/configuration/http/connections"
      }
    },
    {
      "key": "options",
      "meta": {
        "href": "/api/configuration/http/options"
      }
    },
    {
      "key": "settings_policies",
      "meta": {
        "href": "/api/configuration/http/settings_policies"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa",
    "href": "/api/configuration/http",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/ica",
    "parent": "/api/configuration",
    "previous": "/api/configuration/datetime",
    "transaction": "/api/transaction"
  }
}
```
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### HTTP connection policies

Connection policies determine if a server can be accessed from a particular client. Connection policies reference other resources (policies, usergroups, keys) that must be configured and available before creating a connection policy.

### URL

GET https://<IP-address-of-SPS>/api/configuration/http/connections/
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists HTTP connection policies.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/connections/
```

The following command retrieves the properties of a specific policy.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/connections/<connection-key>
```

Response

The following is a sample response received when querying an HTTP connection policy with proxy connection.

For details of the meta object, see Message format on page 10.

```json
{
    "key": "http-connection-simple-proxy",
    "body": {
        "name": "http_proxy",
        "active": true,
        "network": {
            "clients": ["0.0.0.0/0"],
            "targets": ["0.0.0.0/0"],
            "ports": [3128]
        }
    }
}
```
"server_address": {
  "selection": "inband",
  "dns_server": null,
  "dns_suffixes": [],
  "exception_domains": [],
  "domains": [
    {
      "domain": {
        "selection": "domain",
        "value": "**"
      },
      "port": 80
    }
  ]
},
"source_address": {
  "selection": "box_address"
},
"web_proxy": {
  "enabled": true,
  "transport_security": {
    "selection": "disabled"
  }
},
"transport_security": {
  "selection": "disabled"
},
"access_control": [],
"indexing": {
  "enabled": true,
  "policy": {
    "key": "-50000",
    "meta": {
      "href": "/api/configuration/policies/indexing/-50000"
    }
  },
  "priority": 3
},
"rate_limit": {
  "enabled": false
},
"log_audit_trail_downloads": true,
"channel_database_cleanup": {
  "enabled": false
},
"policies": {
  "channel_policy": {
    "key": "-304001002",
    "meta": {
      "href": "/api/configuration/http/channel_policies/-304001002"
    }
  }
}


<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the connection policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the properties of the connection policy.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the connection policy</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Set to false to suspend the connection policy. Connection settings are preserved.</td>
</tr>
<tr>
<td>network</td>
<td>Top level element</td>
<td></td>
</tr>
<tr>
<td>clients</td>
<td>list, string</td>
<td>List of client (&quot;from&quot;) IP addresses.</td>
</tr>
<tr>
<td>ports</td>
<td>list, integers</td>
<td>List of target ports.</td>
</tr>
<tr>
<td>targets</td>
<td>list, integers</td>
<td>List of target IP addresses.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>string</td>
<td>Top level item</td>
<td>Defines the address where the clients connect to.</td>
</tr>
<tr>
<td>server_address</td>
<td>Top level item</td>
<td>Allows you to configure Source Network Address Translation (SNAT) on the server side of SPS. SNAT determines the IP address SPS uses in the server-side connection. The target server will see the connection coming from this address.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures Source Network Address Translation. Possible values are:</td>
</tr>
<tr>
<td>box_address</td>
<td></td>
<td>• Default. Uses the network address of the logical interface of SPS.</td>
</tr>
<tr>
<td>original</td>
<td></td>
<td>• Uses the IP address of the client, as seen by SPS.</td>
</tr>
<tr>
<td>fix</td>
<td></td>
<td>• Uses a fixed address when connecting to the remote server.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if the value of the selection element is set to fix. The IP address to use as the source address in server-side connections.</td>
</tr>
<tr>
<td>web_proxy</td>
<td>Top level item</td>
<td>This will allow the clients to use SPS as an HTTP web proxy.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>When set to true This will allow the clients to use SPS as an HTTP web proxy.</td>
</tr>
<tr>
<td>transport_security</td>
<td>Top level item</td>
<td>Configures the transport security (TLS) of the web proxy connection, between the client and SPS. Note that this setting requires a compatible client application</td>
</tr>
<tr>
<td><strong>Element</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>transport_security</td>
<td>Top level</td>
<td>that is capable of using TLS-secured web proxy connections.</td>
</tr>
<tr>
<td></td>
<td>element</td>
<td>Configures the end-to-end encryption used in the sessions.</td>
</tr>
<tr>
<td>access_control</td>
<td>Top level</td>
<td>Collection of access policies. Access policies define who can authorize and</td>
</tr>
<tr>
<td></td>
<td>list</td>
<td>audit a connection.</td>
</tr>
<tr>
<td>indexing</td>
<td>Top level</td>
<td>Configures indexing for the connection policy.</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable indexing the connections.</td>
</tr>
<tr>
<td>policy</td>
<td>string</td>
<td>References the identifier of the indexing policy. You can configure indexing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>policies at the /api/configuration/policies/indexing/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an indexing policy, use the value of the returned key as the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>value of the policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>priority</td>
<td>int</td>
<td>Specifies the indexing priority for the connection. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Very low priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Low priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Normal (default) priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Very high priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Near real-time priority.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>rate_limit</td>
<td>Top level element</td>
<td>Connection rate limit.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to provide a connection rate limit.</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>The number of connections (per minute) that are allowed in the connection policy.</td>
</tr>
<tr>
<td>log_audit_trail_downloads</td>
<td>boolean</td>
<td>Set to true to log audit trail downloads.</td>
</tr>
<tr>
<td>channel_database_cleanup</td>
<td>Top level item</td>
<td>Configures cleanup of the connection metadata on the connection policy's level.</td>
</tr>
<tr>
<td>days</td>
<td>int</td>
<td>Retention time, in days. Must not exceed the retention time of the archive_cleanup_policy, and the retention time configured in the global settings of the protocol.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The global settings of the HTTP protocol are available at the api/configuration/http/options endpoint.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable periodical cleanup of the connection metadata.</td>
</tr>
<tr>
<td>override_log_level</td>
<td>Top level item</td>
<td>Specifies the verbosity level of sessions handled by this connection policy. The log level of other connection policies is not affected. If disabled, the log level set at the /api/configuration/&lt;protocol&gt;/options endpoint is used.</td>
</tr>
</tbody>
</table>

- To use the default log level, disable this option:

```
"override_log_level": {
    "enabled": false
},
```

- To use a custom log level for the connection policy, enable this
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>option and set the log level to use:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;override_log_level&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;log_level&quot;: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>policies</td>
<td>Top level item</td>
<td>List of policies referenced by the connection policy.</td>
</tr>
<tr>
<td>channel_policy</td>
<td>string</td>
<td>References the identifier of the channel policy. The value of this option cannot be null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a channel policy, use the value of the returned key as the value of the channel_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can configure HTTP channel policies at the /api/configuration/http/channel_policies/ endpoint.</td>
</tr>
<tr>
<td>settings</td>
<td>string</td>
<td>References the identifier of the settings policy. The value of this option cannot be null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a settings policy for this protocol, use the value of the returned key as the value of the settings element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can configure HTTP settings policies at the /api/configuration/http/settings_policies/ endpoint.</td>
</tr>
<tr>
<td>audit_policy</td>
<td>string</td>
<td>Cannot be null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>References the identifier of the audit policy. You can configure audit policies at the /api/configuration/policies/audit_policies/ endpoint.</td>
</tr>
</tbody>
</table>
|              |      | To modify or add an audit policy, use the
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap_server</td>
<td>string</td>
<td>References the identifier of the LDAP server. You can configure LDAP servers at the /api/configuration/policies/ldap_servers/ endpoint. To modify or add an LDAP server, use the value of the returned key as the value of the ldap_server element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>backup_policy</td>
<td>string</td>
<td>References the identifier of the backup policy. You can configure backup policies at the /api/configuration/policies/backup_policies/ endpoint. To modify or add a backup policy, use the value of the returned key as the value of the backup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>authenticatioen_policy</td>
<td>string</td>
<td>Cannot be null. References the identifier of the authentication policy. You can configure authentication policies at the /api/configuration/http/authentication_policies/ endpoint. To modify or add an authentication policy, use the value of the returned key as the value of the authentication_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>usermapping_policy</td>
<td>string</td>
<td>References the identifier of a Usermapping Policy. You can configure Usermapping Policies at the /api/configuration/policies/usermapping_policies/ endpoint. To modify or add a Usermapping Policy, use the value of the returned key as the value of the returned key as the</td>
</tr>
</tbody>
</table>
value of the usermapping_policies element, and remove any child elements (including the key).

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive_cleanup_policy</td>
<td>string</td>
<td>References the identifier of the archive/cleanup policy. You can configure archive and cleanup policies at the /api/configuration/policies/archive_cleanup_policies/ endpoint. To modify or add an archive/cleanup policy, use the value of the returned key as the value of the archive_cleanup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>analytics_policy</td>
<td>string</td>
<td>References the identifier of the analytics policy. You can configure analytics policies at the /api/configuration/analytics/ endpoint. To add or modify an analytics policy, use the value of the returned key as the value of the analytics element, and remove any child elements (including the key).</td>
</tr>
</tbody>
</table>

Elements of server_address

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>custom_dns</td>
<td>string</td>
<td>Configures a DNS server that is used to reverse-resolve the hostname if the Channel Policy contains the address of the target as a hostname instead of an IP address. By default, this is disabled and SPS uses the DNS server set in the /api/configuration/network/dns endpoint.</td>
</tr>
</tbody>
</table>

- To use the default DNS, disable this option:

```json
"server_address": {
"custom_dns": {
  "enabled": false
},
...
```
<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- To use a custom DNS, enable this option and set the IP address of the domain name server to use:

```json
"server_address": {
  "custom_dns": {
    "enabled": true,
    "server": "192.168.1.1"
  },
  ...
},
```

**selection** | string | Configures the address where the clients connect to. Possible values are:

- original
  Connect to the same address specified by the client.

- nat
  Perform a network address translation on the target address.
  Must be used with the network element.

- fix
  Must be used with the address and port elements.

- inband
  Extract the address of the server from the username.
  Must be used with the domains element.

  Optional elements: exception_domains, dns_server, and dns_suffixes.

**dns_server** | string | Can only be used if selection is set to inband.

IP address or the hostname of the
<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dns_suffixes</td>
<td>list, string</td>
<td>Can only be used if selection is set to inband. If the clients do not include the domain name when addressing the server (for example they use username@server instead of <a href="mailto:username@server.example.com">username@server.example.com</a>), SPS can automatically add domain information (for example example.com). You can add multiple domain names. SPS attempts to resolve the target address by appending the domain names in the provided order, and uses the first successfully resolved address to establish the connection.</td>
</tr>
<tr>
<td>domains</td>
<td>Top level list</td>
<td>Must be used if selection is set to inband.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level item</td>
<td>Lists the address ranges that are included in the connection policy.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies if the target address range is provided as a domain or as an IP range. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the target address is an IP range.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- domain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the target address is a domain.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The address range of the target server(s). Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port of the target server(s).</td>
</tr>
<tr>
<td>exception_domains</td>
<td>Top</td>
<td>Can only be used if selection is set to inband.</td>
</tr>
</tbody>
</table>
### Elements of server_address

<table>
<thead>
<tr>
<th><strong>Elements</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>level list</td>
<td></td>
<td>Lists the address ranges that are excluded from the connection policy.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level item</td>
<td>Contains the excluded address range.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies if the excluded address(es) are provided as a domain or as an IP range. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>address</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the excluded address is an IP range.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>domain</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the excluded address is a domain.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The excluded address(es). Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The excluded port.</td>
</tr>
</tbody>
</table>

### Elements of web_proxy.transport_security

<table>
<thead>
<tr>
<th><strong>Elements</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures the encryption used in the sessions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>disabled</strong>: Use unencrypted web proxy connection between the HTTP client and.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;transport_security&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;disabled&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>tls</strong>: Enables TLS-encryption.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;transport_security&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;tls&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Elements of web_proxy.transport_security</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>host_certification_method</td>
<td>JSON object</td>
<td>Selects the certificate to show to the peers. You have the following options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Use the same certificate for each connection:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select this option if you want to use the same certificate for each connection. Note that you must reference a certificate that includes its private key that you have already uploaded to SPS. For details, see Certificates stored on SPS on page 287.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;host_certification_method&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;fix&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;x509_identity&quot;: &quot;893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Generate a certificate for the target requested by the client:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select this option if you want to generate a certificate for the target requested by the client. Note that you must reference a Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 406.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Elements of `web_proxy.transport_security`

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>host_certification_method</code>:</td>
<td>{</td>
</tr>
<tr>
<td>&quot;selection&quot;: &quot;generate&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;signing_ca&quot;: &quot;1904188625a843f11d30a-5&quot; }</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>selection</th>
<th>string</th>
<th>Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>fix</td>
<td>if you want to use the same certificate for every peer.</td>
<td></td>
</tr>
<tr>
<td>generate</td>
<td>if you want to generate a certificate for the target requested by the client.</td>
<td></td>
</tr>
</tbody>
</table>

| x509_identity | string | Reference a certificate that includes its private key that you have already uploaded to SPS. For details, see [Certificates stored on SPS](#) on page 287. |

| signing_ca | string | Reference the Signing CA that you have already configured on SPS. For details, see [Signing CA policies on page 406](#). |

### Elements of `body.transport_security`

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
</tr>
<tr>
<td>- disabled</td>
<td>Use unencrypted connection between the HTTP client and server.</td>
</tr>
</tbody>
</table>

| "transport_security": | {  |
|   "selection": "disabled" |
|   "client-only": Enables half- | |

---

**SPS 6.13.0 REST API Reference Guide**

**HTTP connections**
<table>
<thead>
<tr>
<th>Elements of</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body.transport_security</td>
<td>sidl TLS encryption. Require HTTPS on client side, and HTTP on server side.</td>
</tr>
<tr>
<td>allow_non_encrypted</td>
<td>boolean</td>
</tr>
<tr>
<td>server_certificate_check</td>
<td>Top level item</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
"transport_security": {
  "selection": "client-only"
}

"transport_security": {
  "selection": "client-server",
  "allow_non_encrypted": true
  "server_certificate_check": {}
}
```

```
"server_certificate_check": {
  "enabled": false
},
```

"source": "SPS 6.13.0 REST API Reference Guide"
### Elements of `body.transport_security`

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean, To verify the certificate of the destination server, set to true. In this case, you will also have to reference a trusted_ca.</td>
</tr>
<tr>
<td>trusted_ca</td>
<td>string, Reference a Trusted CA list.</td>
</tr>
<tr>
<td>host_certification_method</td>
<td>JSON object, Selects the certificate to show to the peers. You have the following options:</td>
</tr>
</tbody>
</table>

- **Use the same certificate for each connection:**

Select this option if you want to use the same certificate for each connection. Note that you must reference a certificate that includes its private key that you have already uploaded to SPS. For details, see Certificates stored on SPS on page 287.

```
"host_certification_method": {  
  "selection": "fix",  
  "x509_identity": "893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a"  
}
```
<table>
<thead>
<tr>
<th>Elements of body.transport_security</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

- **Generate a certificate for the target requested by the client:**

  Select this option if you want to generate a certificate for the target requested by the client. Note that you must reference a Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 406.

  ```json
  "host_certification_method": {
    "selection": "generate",
    "signing_ca": "1904188625a843f11d30a-5"
  },
  ```

<table>
<thead>
<tr>
<th>selection</th>
<th>string</th>
<th>Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>fix</td>
<td></td>
<td>if you want to use the same certificate for every peer.</td>
</tr>
<tr>
<td>generate</td>
<td></td>
<td>if you want to generate a certificate for the target requested by the client.</td>
</tr>
</tbody>
</table>

| x509_identity | string | Reference a certificate that includes its private key that you have already uploaded to SPS. For details, see Certificates stored on SPS on page 287. |

<p>| signing_ca | string | Reference the Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 406. |</p>
<table>
<thead>
<tr>
<th>Elements of access control</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizer</td>
<td>string</td>
<td>The usergroup (local or LDAP) who can authorize or audit the connection. Local usergroups can be added or modified at the /api/configuration/aaa/local_database/groups/ endpoint.</td>
</tr>
<tr>
<td>permission</td>
<td>string</td>
<td>Defines the permissions of the authorizer usergroup. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- audit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- authorize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the authorize permission can authorize connection requests.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- audit_and_authorize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the audit_and_authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.</td>
</tr>
<tr>
<td>require_different_ip</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different IP addresses.</td>
</tr>
<tr>
<td>require_different_username</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different usernames.</td>
</tr>
<tr>
<td>subject</td>
<td>Top level item</td>
<td>Defines the subjects of the access control policy.</td>
</tr>
<tr>
<td>group</td>
<td>string</td>
<td>The usergroup (local or LDAP) that is subject to the access control policy. Local usergroups can be added or modified at the /api/configuration/aaa/local_database/groups/ endpoint.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- everybody</td>
</tr>
</tbody>
</table>
### Elements of access control

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Every user is subject to the access control policy.</td>
<td></td>
</tr>
<tr>
<td><strong>only</strong></td>
<td>Requires the group element. Members of the user group specified in the group element are subject to the access control policy.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### HTTP channels

The available HTTP channel types and their functionalities are described below. For details on configuring channel policies, see Channel policy.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>http</td>
<td>No</td>
<td><strong>http</strong>: Enables access to the server. This channel must be enabled for HTTP connections to work.</td>
</tr>
</tbody>
</table>
| websocket | No          | **websocket**: Enables all WebSocket traffic. If the WebSocket channel type is not allowed, HTTP requests trying the WebSocket upgrade are rejected. 

WebSocket/VNC audit trails: You can replay audit trails of a WebSocket connection in your browser or using the Safeguard Desktop Player application only if it contains Virtual Network Computing (VNC) traffic. For all other WebSocket connections, export the audit trail as a PCAP file and replay it using the Safeguard Desktop Player application.
HTTP authentication policies

Lists the configured authentication methods that can be used in a connection. Each connection policy uses an authentication policy to determine how the client can authenticate to SPS.

URL

GET https://<IP-address-of-SPS>/api/configuration/http/authentication_policies

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d03e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists HTTP authentication policies.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/authentication_policies

The following command retrieves the properties of a specific policy.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/authentication_policies<object-id>

Response

The following is a sample response received when listing HTTP authentication policies. For details of the meta object, see Message format on page 10.
When retrieving the endpoint of a specific policy, the response is the following.

```
{
  "key": "http-auth-pol-4",
  "body": {
    "name": "http_radius",
    "gateway_authentication": {
      "selection": "radius",
      "servers": [
        {
          "address": {
            "selection": "ip",
            "value": "1.2.3.4"
          },
          "port": 1812,
          "shared_secret": {
            "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "meta": {
              "href": "/api/configuration/passwords#XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
            }
          }
        }
      ]
    }
  }
}
```
],
"authentication_protocol": "pap",
"timeout": 3600,
"keepalive": true
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element</td>
<td>Contains the elements of the policy.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>gateway_authentication</td>
<td>Top level item</td>
<td>Client-side gateway authentication settings. The value of selection defines which authentication method is used.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the authentication method for client-side gateway authentication. Possible values are:</td>
</tr>
</tbody>
</table>
|                             |        | • none
Disables client-side gateway authentication. |
|                             |        | • ldap
Uses the LDAP server selected for the connection policy. LDAP servers can be configured in the /api/configuration/policies/ldap_servers endpoint). |
|                             |        | • local
Uses the local user database configured in the /api/configuration/policies/user_databases/ endpoint. |
|                             |        | To use this option, you must |

To use this option, you must
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>also configure the user_database element.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>radius</td>
<td></td>
<td>Uses one or more Radius servers for authentication.</td>
</tr>
<tr>
<td>To use this option, you must also configure the authentication_protocol and servers elements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>servers</td>
<td>Top level list</td>
<td>Only if selection is set to radius Defines the properties of the RADIUS servers used for client-side authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A valid list item consists of the address, port and shared_secret elements.</td>
</tr>
<tr>
<td>authentication_protocol</td>
<td>Top level item</td>
<td>Only if selection is set to radius RADIUS setting. Set to pap to use the Password Authentication Protocol. To use the Challenge-Handshake Authentication Protocol, set it to chap.</td>
</tr>
<tr>
<td>user_database</td>
<td>string</td>
<td>Only if selection is set to local References the key of the local user database. You can configure local user databases at the /api/configuration/policies/user_databases/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a local user database, use the value of the returned key as the value of the user_database element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>timeout</td>
<td>integer (second-s)</td>
<td>Specify the time remaining until a successful gateway authentication times out.</td>
</tr>
<tr>
<td>keepalive</td>
<td>boolean</td>
<td>Set to true to avoid interruptions for active HTTP sessions. Active HTTP sessions can extend the gateway authentication beyond the configured timeout.</td>
</tr>
<tr>
<td>Elements of servers</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>address</td>
<td>Top level</td>
<td>Defines the address of a RADIUS server.</td>
</tr>
<tr>
<td></td>
<td>element</td>
<td></td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Required child of the address element. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ip</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains the IP of the RADIUS server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fqdn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains the FQDN of the RADIUS server.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The IP or the FQDN address of the RADIUS server.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number of the RADIUS server.</td>
</tr>
<tr>
<td>shared_secret</td>
<td>string</td>
<td>References the key of the shared secret for the RADIUS server. You can</td>
</tr>
<tr>
<td></td>
<td></td>
<td>configure shared secrets at the /api/configuration/passwords/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a shared secret, use the value of the returned key as</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the value of the shared_secret element, and remove any child elements (</td>
</tr>
<tr>
<td></td>
<td></td>
<td>including the key).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Alternatively, you can include the new password as plain text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;shared_secret&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;plain&quot;: &quot;&lt;new-password&gt;&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

Examples:

Querying base authentication policy without gateway authentication:

```
{
  "key": "-304002001",
  "body": {
    "name": "base",
    "gateway_authentication": {
      "selection": "none"
    }
  }
}
```

Querying authentication policy with LDAP backend:
Querying authentication policy with local backend:

```json
{
    "key": "http-auth-pol-3",
    "body": {
        "name": "http_local",
        "gateway_authentication": {
            "selection": "local",
            "user_database": {
                "key": "local-user-database-1",
                "meta": { "href": "/api/configuration/policies/user_databases/local-user-database-1" }
            },
            "timeout": 3600,
            "keepalive": true
        }
    }
}
```

Querying authentication policy with RADIUS backend:

```json
{
    "key": "http-auth-pol-4",
    "body": {
        "name": "http_radius",
        "gateway_authentication": {
            "selection": "radius",
            "servers": [
                {
                    "address": {
                        "selection": "ip",
                        "value": "1.2.3.4"
                    },
                    "port": 1812,
                    "shared_secret": {
                        "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                        "meta": { "href": "/api/configuration/passwords#XXXXXXXX-XXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" }
                    }
                }
            ],
            "timeout": 3600,
            "keepalive": true
        }
    }
}
```
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Add an HTTP authentication policy

To add an HTTP authentication policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new policy.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/http/authentication_policies/ endpoint. You can find a detailed description of the available parameters listed in Element.
If the POST request is successful, the response includes the key of the new policy. For example:

```json
{
    "key": "6f924f39-e4c9-4b0f-8018-8842e2115ebd",
    "meta": {
        "href": "/api/configuration/http/authentication_policies/6f924f39-e4c9-4b0f-8018-8842e2115ebd",
        "parent": "/api/configuration/http/authentication_policies",
        "transaction": "/api/transaction"
    }
}
```

3. **Commit your changes.**

For more information, see Commit a transaction on page 31.

**Modify an HTTP authentication policy**

To modify an HTTP authentication policy, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the policy.**

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/http/authentication_policies/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

**Global HTTP options**

List of options that affect all HTTP connections.

**URL**

`GET https://<IP-address-of-SPS>/api/configuration/http/options`
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
</table>
| session_id    | Contains the authentication token of the user                | Required | The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists global HTTP options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/options
```

Response

The following is a sample response received when listing global HTTP options.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "audit": {
            "cleanup": {
                "enabled": false
            },
            "timestamping": {
                "selection": "local",
                "signing_interval": 30
            }
        },
        "service": {
            "enabled": true,
            "log_level": 4
        }
    },
    "key": "options",
}
```
"meta": {
    "first": "/api/configuration/http/channel_policies",
    "href": "/api/configuration/http/options",
    "last": "/api/configuration/http/settings_policies",
    "next": "/api/configuration/http/settings_policies",
    "parent": "/api/configuration/http",
    "previous": "/api/configuration/http/channel_policies",
    "transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Top level item</td>
<td>Contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level item</td>
<td>Contains the elements of the global HTTP options.</td>
</tr>
<tr>
<td>audit</td>
<td>Top level item</td>
<td>Contains settings for timestamping and cleanup.</td>
</tr>
<tr>
<td>service</td>
<td>Top level item</td>
<td>Global setting to enable HTTP connections, and specify the logging detail.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable HTTP connections.</td>
</tr>
<tr>
<td>log_level</td>
<td>int</td>
<td>Defines the logging detail of HTTP connections.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of audit</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanup</td>
<td>Top level item</td>
<td>Global retention settings for HTTP connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.</td>
</tr>
</tbody>
</table>

channel_database_cleanup_days int Only if enabled is set to true. Global retention time for the metadata of HTTP connections, in days. Must exceed the retention time of the archiving policy (or policies) used for HTTP connections, and the
<table>
<thead>
<tr>
<th>Elements of audit</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enabled</td>
<td>boolean To enable the global cleanup of HTTP connection metadata, set this element to true.</td>
</tr>
<tr>
<td>timestamping</td>
<td>Top level item</td>
<td>Global timestamping settings for HTTP connections.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures local or remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set local to use SPS for timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set remote to configure a remote timestamping server.</td>
</tr>
<tr>
<td>server_url</td>
<td>string</td>
<td>Required for remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.</td>
</tr>
<tr>
<td>oid</td>
<td>Top level item</td>
<td>The Object Identifier of the policy used for timestamping.</td>
</tr>
<tr>
<td></td>
<td>enabled</td>
<td>boolean Required for remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set to true to configure the Object Identifier of the timestamping policy on the timestamping remote server.</td>
</tr>
<tr>
<td>policy_oid</td>
<td>string</td>
<td>Required if the oid is enabled. The Object Identifier of the timestamping policy on the remote timestamping server.</td>
</tr>
<tr>
<td></td>
<td>signing_ interval</td>
<td>int Time interval for timestamping open connections, in seconds.</td>
</tr>
</tbody>
</table>

**Examples:**
Set SPS as the timestamping server:
Enable cleanup, and set it to occur every 10 days:

```json
{
  "audit": {
    "cleanup": {
      "enabled": false
    },
    "timestamping": {
      "selection": "local",
      "signing_interval": 30
    }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```json
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
      "selection": "local",
      "signing_interval": 30
    }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```
Change timestamping to a remote server, and specify the 1.2.3 timestamping policy:

```
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": true,
                "policy_oid": "1.2.3"
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Modify global HTTP settings

To modify global HTTP settings, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the global HTTP settings endpoint.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/http/options` endpoint. You can find a detailed description
of the available parameters listed in Element. The elements of the audit item are described in Elements of audit.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**HTTP settings policies**

HTTP settings policies define protocol-level settings for idle and session timeout. You can create multiple policies, and choose the appropriate one for each HTTP connection.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/http/settings_policies

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication</td>
</tr>
<tr>
<td>Cookie name</td>
<td>Description</td>
<td>Required</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
</tr>
<tr>
<td>token</td>
<td>token of the user response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists HTTP settings policies.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/settings_policies
```

The following command retrieves the properties of a specific policy.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/settings_policies/<policy-id>
```

### Response

The following is a sample response received when listing HTTP settings policies. For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "key": "-3040010",
            "meta": {
                "href": "/api/configuration/http/settings_policies/-3040010"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/http/channel_policies",
        "href": "/api/configuration/http/settings_policies",
        "last": "/api/configuration/http/settings_policies",
        "next": null,
```
When retrieving the endpoint of a specific policy, the response is the following.

```
{
    "body": {
        "client_tls_security_settings": {
            "cipher_strength": {
                "selection": "recommended"
            },
            "minimum_tls_version": "TLSv1_2"
        },
        "name": "default",
        "server_tls_security_settings": {
            "cipher_strength": {
                "selection": "recommended"
            },
            "minimum_tls_version": "TLSv1_2"
        },
        "session_timeout": 900,
        "timeout": 300
    }
}
```

When retrieving the default settings policy with a built-in HTTP proxy error template, the response is the following.

```
{
    "body": {
        "client_tls_security_settings": {
            "cipher_strength": {
                "selection": "recommended"
            },
            "minimum_tls_version": "TLSv1_2"
        },
        "name": "default",
        "server_tls_security_settings": {
            "cipher_strength": {
                "selection": "recommended"
            },
            "minimum_tls_version": "TLSv1_2"
        },
        "session_timeout": 900,
        "timeout": 300
    }
}
```
When you create a new settings policy with a custom error template, the response is the following.

```json
{
  "key": "-3040010",
  "body": {
    "name": "default",
    "timeout": 300,
    "session_timeout": 900,
    "webapp_session_cookies": [],
    "client_tls_security_settings": {
      "minimum_tls_version": "TLSv1_2",
      "cipher_strength": {
        "selection": "recommended"
      }
    },
    "server_tls_security_settings": {
      "minimum_tls_version": "TLSv1_2",
      "cipher_strength": {
        "selection": "recommended"
      }
    },
    "error_template": {
      "selection": "builtin"
    }
  }
}
```
"error_template": {
    "selection": "custom",
    "reference": "123456789"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>The elements of the HTTP settings policy.</td>
</tr>
<tr>
<td></td>
<td>element (string)</td>
<td></td>
</tr>
<tr>
<td>client_tls_security_settings</td>
<td>JSON object</td>
<td>Configures TLS security settings on the client side.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the HTTP settings policy. Cannot contain whitespace.</td>
</tr>
<tr>
<td>server_tls_security_settings</td>
<td>JSON object</td>
<td>Configures TLS security settings on the server side.</td>
</tr>
<tr>
<td>session_timeout</td>
<td>int</td>
<td>Session timeout, in seconds.</td>
</tr>
<tr>
<td>timeout</td>
<td>int</td>
<td>Idle timeout, in seconds. Note that the SPS web UI displays the same value in seconds.</td>
</tr>
<tr>
<td>webapp_session_cookies</td>
<td>list (string)</td>
<td>To distinguish the audited HTTP requests and responses based on the session cookies of web applications, enter the name of the session cookie, for example, PHPSESSID, JSESSID, or ASP.NET_SessionId. Note that the names of session cookies are case sensitive. Note that this is a priority list. If there are multiple cookie names, SPS will use the first one from this list it finds in the request headers to assign the requests to a session.</td>
</tr>
<tr>
<td>error_template</td>
<td>object</td>
<td></td>
</tr>
<tr>
<td>error_template.selection</td>
<td>enum</td>
<td>The type of the error template. Possible values: builtin, custom</td>
</tr>
<tr>
<td>error_template.reference</td>
<td>number</td>
<td>The identifier of the error template.</td>
</tr>
</tbody>
</table>
### Elements of client_tls_security_settings and server_tls_security_settings

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cipher_strength</td>
<td>JSON object</td>
<td>Specifies the cipher string OpenSSL will use.</td>
</tr>
<tr>
<td>custom_cipher</td>
<td>string</td>
<td>The list of ciphers you want to permit SPS to use in the connection. For more details on customizing this list, check the 'openssl-ciphers' manual page on your SPS appliance.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies the cipher string OpenSSL will use. The following settings options are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>recommended</strong>: this setting only uses ciphers with adequate security level.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>custom</strong>: this setting allows you to specify the list of ciphers you want to permit SPS to use in the connection. This setting is only recommended to ensure compatibility with older systems. For more details on customizing this list, check the 'openssl-ciphers' manual page on your SPS appliance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: <code>ALL:!aNULL:@STRENGTH</code></td>
</tr>
<tr>
<td>minimum_tls_version</td>
<td>string</td>
<td>Specifies the minimal TLS version SPS will offer during negotiation. The following settings options are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>TLSv1_2</strong>: this setting only offers TLS version 1.2 during the negotiation. This is the recommended setting.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>TLSv1_1</strong>: this setting offers TLS version 1.1 and later versions during the negotiation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>TLSv1_0</strong>: this setting offers TLS version 1.0 and later versions during the negotiation.</td>
</tr>
</tbody>
</table>

### Add HTTP settings policies

To add a settings policy, you have to:

1. **Open a transaction.**

   For more information, see [Open a transaction](#) on page 29.
2. **Create the JSON object for the new policy.**

POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/http/settings_policies/ endpoint. You can find a detailed description of the available parameters listed in Element.

If the POST request is successful, the response includes the key of the new policy. For example:

```json
{
  "key": "3848c708-2e1d-4463-b232-0c8c5875ff55",
  "meta": {
    "href": "https://<IP-address-of-SPS>/api/configuration/http/settings_policies/3848c708-2e1d-4463-b232-0c8c5875ff55",
    "parent": "https://<IP-address-of-SPS>/api/configuration/http/settings_policies",
    "transaction": "https://<IP-address-of-SPS>/api/transaction"
  }
}
```

3. **Commit your changes.**

For more information, see Commit a transaction on page 31.

**Modify HTTP settings policies**

To modify a settings policy, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the policy.**

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/http/settings_policies/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Creating custom HTTP error templates**

Configure HTTP error templates to create custom error pages when HTTP proxy errors occur.

**URL**

```plaintext
GET https://<IP-address-of-SPS>/api/configuration/http/error_templates
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
# Operations

Operations with the `/http/error_templates` endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Querying existing error templates</td>
<td>GET</td>
<td>/api/configuration/http/error_templates</td>
<td></td>
</tr>
<tr>
<td>Creating a new error template</td>
<td>POST</td>
<td>/api/configuration/http/error_templates</td>
<td></td>
</tr>
<tr>
<td>Updating an existing error template</td>
<td>PUT</td>
<td>/api/configuration/http/error_templates/&lt;key&gt;</td>
<td></td>
</tr>
<tr>
<td>Querying error templates info</td>
<td>GET</td>
<td>/api/configuration/http/error_templates?info=</td>
<td></td>
</tr>
<tr>
<td>Querying custom error template preview</td>
<td>GET</td>
<td>/api/configuration/http/error_templates?preview=&amp;error_type=&lt;type-of-the-error-template&gt;</td>
<td></td>
</tr>
</tbody>
</table>

## Sample request

The following command lists the available HTTP proxy error templates.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/error_templates
```

## Response

The following is a sample response received when available HTTP proxy error templates are queried.

For details of the `meta` object, see Message format on page 10.

```json
{
    "name": "Template_with_logo",
    "color": "#123456",
    "logo": "<logo_id> from the response of the first creation",
    "brand_name": "Noname brand",
    "md_contents": {
        "auth": "Custom auth failed markdown content.",
        "badcontent": "Custom bad content error markdown content.",
        "clientsyntax": "Custom client syntax error markdown content.",
        "clienttimeout": "Custom client timeout error markdown content.",
        "connecterror": "Custom connection error markdown content.",
        "ftperror": "Custom FTP error markdown content."
    }
}
```
The following is a sample response received when HTTP proxy error template information is queried with /api/configuration/http/error_templates?info=.

```json
{
  "error_type_names": {
    "auth": "Authentication Failed",
    "badcontent": "Bad Content",
    "clientsyntax": "Client Syntax",
    "clienttimeout": "Client Timeout",
    "connecterror": "Connection Error",
    "ftperror": "FTP Error",
    "internal": "Internal Error",
    "invalidurl": "Invalid URL",
    "ioerror": "I/O Error",
    "policysyntax": "Policy Syntax",
    "policyviolation": "Policy Violation",
    "redirect": "Redirect",
    "serversyntax": "Server Syntax",
    "servertimeout": "Server Timeout"
  }
}
```

Elements of the response message include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the template.</td>
<td></td>
</tr>
<tr>
<td>color</td>
<td>number</td>
<td>The color of the brand and links appearing in the error template.</td>
<td>The value is given in hex color code.</td>
</tr>
<tr>
<td>logo</td>
<td>string? union?</td>
<td>The identifier of the logo.</td>
<td>The value of the logo can be 'null'.</td>
</tr>
<tr>
<td>brand_name</td>
<td>string</td>
<td>The name of the brand.</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>md_contents</td>
<td>object</td>
<td>Contains a list of error templates written in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.auth</td>
<td>string</td>
<td>The content of the <strong>Authentication failed</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.badcontent</td>
<td>string</td>
<td>The content of the <strong>Bad content</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.clientsyntax</td>
<td>string</td>
<td>The content of the <strong>Client syntax</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.clienttimeout</td>
<td>string</td>
<td>The content of the <strong>Client timeout</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.connectorror</td>
<td>string</td>
<td>The content of the <strong>Connect error</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.ftperror</td>
<td>string</td>
<td>The content of the <strong>FTP error</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.internal</td>
<td>string</td>
<td>The content of the <strong>Internal error</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.invalidurl</td>
<td>string</td>
<td>The content of the <strong>Invalid URL</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.ioerror</td>
<td>string</td>
<td>The content of the <strong>IO error</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.policysyntax</td>
<td>string</td>
<td>The content of the <strong>Policy syntax</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.policyviolation</td>
<td>string</td>
<td>The content of the <strong>Policy violation</strong> error template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.redirect</td>
<td>string</td>
<td>The content of the <strong>Redirect error</strong> template in Markdown.</td>
<td></td>
</tr>
<tr>
<td>md_contents.serversyntax</td>
<td>string</td>
<td>The content of the <strong>Server syntax</strong> error template in Markdown.</td>
<td></td>
</tr>
</tbody>
</table>
### md_contents.server.timeout

The content of the `Server timeout` error template in Markdown.

For details of the meta object, see [Message format](#).

### HTTP response codes

HTTP response codes comprise of standard or endpoint-specific HTTP status and error codes. The following table lists the endpoint-specific HTTP response codes for this request.

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status/Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>NotSupportedProxyErrorType</td>
<td>Preview could not be created. The used error type is not supported. Use an error type from the following list: Authentication failed, Bad content, Client syntax, Client timeout, Connect error, FTP error, Internal, Invalid URL, IO error, Policy syntax, Policy violation, Redirect, Server syntax, Server timeout.</td>
</tr>
</tbody>
</table>

For more information and a complete list of standard HTTP response codes, see [Application level error codes](#) on page 37.

### Uploading a custom logo to your custom HTTP proxy error pages

Upload a custom logo to your custom HTTP proxy error pages.

Supported formats: PNG, JPEG, GIF.

#### URL

```
GET https://<IP-address-of-SPS>/api/configuration/http/proxy_error_logo
```
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Operations

Operations with the /http/proxy_error_logo endpoint include:

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP method</th>
<th>URL</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uploading a custom logo</td>
<td>POST</td>
<td>/api/upload/http/proxy_error_logo</td>
<td>If you upload an oversized logo, for example 800x800 pixels, your image will be automatically resized to 128x128 pixels, keeping the original aspect ratio.</td>
</tr>
</tbody>
</table>

**NOTE:** GET / PUT / DELETE methods are not allowed on logo upload.

Sample request

The following command uploads a custom logo.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/http/proxy_error_logo
```

Response

The following is a sample response received when the custom logo has been uploaded. The response of the creation without the unique ID is the following.
The response of the query without a unique logo ID is the following.

```json
{
   "changes": [
      {
         "new_value": "rest_XXXXXXXX",
         "path": "/api/upload/http/proxy_error_logo",
         "type": "creation"
      }
   ],
   "meta": {
      "href": "/api/transaction/changes",
      "parent": "/api/transaction",
      "transaction": "/api/transaction"
   }
}
```

For details of the meta object, see Message format on page 10.

Elements of the response message include:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>logo_id</td>
<td>string</td>
<td>The identifier of the custom logo.</td>
<td></td>
</tr>
<tr>
<td>supported_formats</td>
<td>enum</td>
<td>The supported image formats.</td>
<td>Possible values: PNG, JPEG, GIF.</td>
</tr>
<tr>
<td>actual_resolution</td>
<td>number</td>
<td>The actual resolution of the uploaded custom logo.</td>
<td></td>
</tr>
<tr>
<td>max_resolution</td>
<td>number</td>
<td>The maximal resolution of the uploaded custom logo.</td>
<td>The maximum accepted size is 4096x4096 pixels.</td>
</tr>
<tr>
<td>file_size</td>
<td>number</td>
<td>The actual size of the file.</td>
<td></td>
</tr>
<tr>
<td>file_limit</td>
<td>number</td>
<td>The maximum size of the file.</td>
<td>The file size limit is 16 megabyte.</td>
</tr>
</tbody>
</table>
HTTP response codes

HTTP response codes comprise of standard or endpoint-specific HTTP status and error codes. The following table lists the endpoint-specific HTTP response codes for this request.

<table>
<thead>
<tr>
<th>HTTP response code</th>
<th>Status/Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>ResolutionTooLarge</td>
<td>The logo you uploaded (5200x5200 pixels) is larger than the maximum accepted size (4096x4096 pixels). Upload a logo that is not larger than 4096x4096 pixels.</td>
</tr>
<tr>
<td>413</td>
<td>FileTooLarge</td>
<td>The file in the request exceeds the file size limitation.</td>
</tr>
<tr>
<td>415</td>
<td>InvalidImageFormat</td>
<td>The file format of the uploaded logo is not supported. Make sure that you upload a logo in one of our supported file formats (PNG, JPEG, GIF).</td>
</tr>
</tbody>
</table>

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 37.
Citrix ICA connections

ICA connections

List of endpoints for configuring the policies, options and connection rules of ICA connections.

URL

GET https://<IP-address-of-SPS>/api/configuration/ica

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available settings for configuring for ICA connections.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ica
```
Response

The following is a sample response received when listing the configuration settings.

For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "key": "channel_policies",
            "meta": {
                "href": "/api/configuration/ica/channel_policies"
            }
        },
        {
            "key": "connections",
            "meta": {
                "href": "/api/configuration/ica/connections"
            }
        },
        {
            "key": "options",
            "meta": {
                "href": "/api/configuration/ica/options"
            }
        },
        {
            "key": "settings_policies",
            "meta": {
                "href": "/api/configuration/ica/settings_policies"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/aaa",
        "href": "/api/configuration/ica",
        "last": "/api/configuration/x509",
        "next": "/api/configuration/local_services",
        "parent": "/api/configuration",
        "previous": "/api/configuration/http",
        "transaction": "/api/transaction"
    }
}
```

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel_policies</td>
<td>List of the default and custom channel policies.</td>
</tr>
<tr>
<td>Item</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>connections</td>
<td>List of ICA connection policies.</td>
</tr>
<tr>
<td>options</td>
<td>List of global ICA options that affect all connections.</td>
</tr>
<tr>
<td>settings_policies</td>
<td>List of protocol-level settings (timeout, reliability). You can create multiple variations, and choose the appropriate one for each connection policy.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**ICA connection policies**

Connection policies determine if a server can be accessed from a particular client. Connection policies reference other resources (policies, usergroups, keys) that must be configured and available before creating a connection policy.

⚠️ **CAUTION:**

The connection policies of this protocol are available in READ-ONLY mode on the REST API. Also, the returned data is incomplete, it does not include any protocol-specific settings, only the parameters that are common to every supported protocol.

To modify the connection policies of this protocol, you must use the SPS web interface.

Using the REST API, you can modify the connection policies of the RDP and SSH protocols.
URL

GET https://<IP-address-of-SPS>/api/configuration/ica/connections/

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in</td>
</tr>
<tr>
<td></td>
<td>token of the user</td>
<td></td>
<td>the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists ICA connection policies.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ica/connections/

The following command retrieves the properties of a specific policy.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ica/connections/<connection-key>

ICA channels

The available ICA channel types and their functionalities are described below. For details on configuring channel policies, see Channel policy.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTXTW</td>
<td>Yes</td>
<td>Drawing (Thinwire): Enables access to the server’s desktop</td>
</tr>
<tr>
<td>Channel</td>
<td>Special options</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>CTXCAM</td>
<td>None</td>
<td><strong>Audio Mapping</strong>: Enable access to the sound device of the server.</td>
</tr>
<tr>
<td>CTXCDM</td>
<td>None</td>
<td><strong>Drive Mapping</strong>: Enable access to the client's hard drives on the server.</td>
</tr>
<tr>
<td>CTXCLIP</td>
<td>None</td>
<td><strong>Clipboard</strong>: Enable access to the server's clipboard: the clipboard of the remote desktop can be pasted into local applications (and vice-versa). Note that SPS can audit the clipboard channel, but the Safeguard Desktop Player cannot search or display its contents.</td>
</tr>
<tr>
<td>CTXSCRD</td>
<td>None</td>
<td><strong>Smartcard</strong>: Enable using client side installed smartcards in server-side applications.</td>
</tr>
<tr>
<td>CTXCOM1</td>
<td>None</td>
<td><strong>Printer (COM1)</strong>: Enable access to the serial port COM1.</td>
</tr>
<tr>
<td>CTXCOM2</td>
<td>None</td>
<td><strong>Printer (COM2)</strong>: Enable access to the serial port COM2.</td>
</tr>
<tr>
<td>CTXLPT1</td>
<td>None</td>
<td><strong>Printer (LPT1)</strong>: Enable access to the parallel port LPT1.</td>
</tr>
<tr>
<td>CTXLPT2</td>
<td>None</td>
<td><strong>Printer (LPT2)</strong>: Enable access to the parallel port LPT2.</td>
</tr>
<tr>
<td>CTXCPM</td>
<td>None</td>
<td><strong>Printer Spooler</strong>: Enable access to the client's printer from the remote desktops and applications.</td>
</tr>
</tbody>
</table>

This channel is for remoting graphics and user input (keyboard, mouse). This channel must be enabled for ICA to work.

Channel-specific actions:

- `content_policy reference`: The ID of the Content policy to apply to the connection.

For example:

```json
"actions": {
  "audit": true,
  "four_eyes": true,
  "content_policy": {
    "key": "433849548566ab327522e6"
    "meta": {
      "href": "/api/configuration/policies/content_policies/44287216854f482e7f2b24"
    }
  },
}
```
<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTXFLSH</td>
<td>None</td>
<td><strong>HDX Mediastream</strong>: Some user widgets (for example Flash player) will not run on the server but on the client. These widgets are controlled from the server side using this channel. This is not supported by Safeguard Desktop Player and it is disabled by default.</td>
</tr>
<tr>
<td>CTXUSB</td>
<td>None</td>
<td><strong>USB</strong>: Enable using client side installed USB devices in server-side applications.</td>
</tr>
<tr>
<td>CTXTWI</td>
<td>None</td>
<td><strong>Seamless</strong>: Enable seamless channels that run a single application on the ICA server, instead of accessing the entire desktop. When disabled, the application window will be accessed along with an empty desktop.</td>
</tr>
<tr>
<td>SPDBRS</td>
<td>None</td>
<td><strong>Speedbrowse</strong>: Speeds up web browsing. Not currently supported by Safeguard Desktop Player, should be disabled by default.</td>
</tr>
<tr>
<td>custom</td>
<td>Yes</td>
<td><strong>Custom</strong>: Applications can open custom channels to the clients connecting remotely to the server. Enabling the Custom channel allows the clients to access all of these custom channels. To permit only specific channels, configure the channels field. Channel-specific access control rules:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- channels: To permit only specific custom channels, configure this field. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;channels&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;restricted&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;restrictions&quot;: [ &quot;CUSTOM1&quot;, &quot;CUSTOM2&quot; ]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

**Global ICA options**

List of options that affect all ICA connections.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/ica/options
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API on page 19</a>. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Sample request

The following command lists global ICA options.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ica/options
```

Response

The following is a sample response received when listing global ICA options.

For details of the `meta` object, see [Message format on page 10](#).

```json
{
   "body": {
      "audit": {
         "cleanup": {
            "enabled": false
         },
         "timestamping": {
            "selection": "local",
            "signing_interval": 30
         }
      },
      "service": {
         "enabled": true,
         "log_level": 4
      }
   },
   "key": "options",
   "meta": {
```
"first": "/api/configuration/ica/channel_policies",
"href": "/api/configuration/ica/options",
"last": "/api/configuration/ica/settings_policies",
"next": "/api/configuration/ica/settings_policies",
"parent": "/api/configuration/ica",
"previous": "/api/configuration/ica/channel_policies",
"transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Top level item</td>
<td>Contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level item</td>
<td>Contains the elements of the global ICA options.</td>
</tr>
<tr>
<td>audit</td>
<td>Top level item</td>
<td>Contains settings for timestamping and cleanup.</td>
</tr>
<tr>
<td>service</td>
<td>Top level item</td>
<td>Global setting to enable ICA connections, and specify the logging detail.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable ICA connections.</td>
</tr>
<tr>
<td>log_level</td>
<td>int</td>
<td>Defines the logging detail of ICA connections.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of audit</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanup</td>
<td>Top level item</td>
<td>Global retention settings for ICA connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.</td>
</tr>
<tr>
<td>channel_database_cleanup_days</td>
<td>int</td>
<td>Global retention time for the metadata of ICA connections, in days. Must exceed the retention time of the archiving policy (or policies) used for ICA connections, and the connection-specific database cleanup times (if configured).</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>To enable the global cleanup of ICA connection metadata, set this element to true.</td>
</tr>
<tr>
<td><strong>Elements of audit</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>----------------------</td>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>timestamping</td>
<td>Top level item</td>
<td>Global timestamping settings for ICA connections.</td>
</tr>
</tbody>
</table>
| selection            | string    | Configures local or remote timestamping.  
  - Set `local` to use SPS for timestamping.  
  - Set `remote` to configure a remote timestamping server. |
| server_url           | string    | Required for remote timestamping.  
  The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported. |
| oid                  | Top level item | The Object Identifier of the policy used for timestamping. |
| enabled              | boolean   | Required for remote timestamping.  
  Set to `true` to configure the Object Identifier of the timestamping policy on the timestamping remote server. |
| policy_oid           | string    | Required if the `oid` is enabled.  
  The Object Identifier of the timestamping policy on the remote timestamping server. |
| signing_interval     | int       | Time interval for timestamping open connections, in seconds. |

**Examples:**

Set SPS as the timestamping server:

```json
{
    "audit": {
        "cleanup": {
            "enabled": false
        },
        "timestamping": {
            "selection": "local",
            "signing_interval": 30
        }
    }
}
```
Enable cleanup, and set it to occur every 10 days:

```json
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "selection": "local",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```json
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": false
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, and specify the 1.2.3 timestamping policy:
Modify global ICA settings

To modify global ICA settings, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the global ICA settings endpoint.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/ica/options endpoint. You can find a detailed description of the available parameters listed in Element. The elements of the audit item are described in Elements of audit.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**ICA settings policies**

ICA settings policies define protocol-level settings (timeout, reliability). You can create multiple policies, and choose the appropriate one for each ICA connection.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/ica/settings_policies

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format)....
**Sample request**

The following command lists ICA settings policies.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ica/settings_policies
```

The following command retrieves the properties of a specific policy.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ica/settings_policies/<policy-id>
```

**Response**

The following is a sample response received when listing ICA settings policies.

For details of the `meta` object, see [Message format](#) on page 10.

```json
{
  "items": [
    {
      "key": "-301101020",
      "meta": {
        "href": "/api/configuration/ica/settings_policies/-301101020"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/ica/channel_policies",
    "href": "/api/configuration/ica/settings_policies",
    "last": "/api/configuration/ica/settings_policies",
    "next": null,
    "parent": "/api/configuration/ica",
    "previous": "/api/configuration/ica/options",
    "transaction": "/api/transaction"
  }
}
```

When retrieving the endpoint of a specific policy, the response is the following.

```json
{
  "body": {
    "name": "default",
    "timeout": 600,
    "inactivity_timeout": {
      "enabled": true
    },
    "preconnect_channel_check": false,
```

---

**SPS 6.13.0 REST API Reference Guide**

Citrix ICA connections

---

One Identity by Quest
"reliability": {
    "reconnect_attempts": 30,
    "reconnect_sleep": 2,
    "reconnect_timeout": 600
},
"timeout": 600
},
"key": "-301101020",
"meta": {
    "first": "/api/configuration/ica/settings_policies/-301101020",
    "href": "/api/configuration/ica/settings_policies/-301101020",
    "last": "/api/configuration/ica/settings_policies/-301101020",
    "next": null,
    "parent": "/api/configuration/ica/settings_policies",
    "previous": null,
    "transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the ICA settings policy.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the ICA settings policy. Cannot contain whitespace.</td>
</tr>
<tr>
<td>preconnect_channel_check</td>
<td>boolean</td>
<td>Before establishing the server-side connection, SPS can evaluate the connection and channel policies to determine if the connection might be permitted at all. The server-side connection is established only if the evaluated policies permit the client to access the server. To enable this function, set the parameter to true.</td>
</tr>
<tr>
<td>reliability</td>
<td>Top level item</td>
<td>Settings for ICA connection attempts.</td>
</tr>
<tr>
<td>timeout</td>
<td>int</td>
<td>Connection timeout, in seconds.</td>
</tr>
<tr>
<td>inactivity_timeout</td>
<td>Top level element</td>
<td></td>
</tr>
</tbody>
</table>
### Add ICA settings policies

To add a settings policy, you have to:

1. **Open a transaction.**
   
   For more information, see *Open a transaction* on page 29.

2. **Create the JSON object for the new policy.**
   
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/ica/settings_policies/ endpoint. You can find a detailed description of the available parameters listed in the parameter list table.

   If the POST request is successful, the response includes the key of the new policy. For example:

   ```json
   {
     "key": "dcd58077-98b3-4c73-8f0b-b34147863028",
     "meta": {
       "href": "/api/configuration/ica/settings_policies/dcd58077-98b3-4c73-8f0b-b34147863028"
     }
   }
   ```
3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

**Modify ICA settings policies**
To modify a settings policy, you have to:

1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the policy.**
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/ica/settings_policies/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in the parameter list table.

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

**Status and error codes**
The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
MSSQL connections

Limitations in handling MSSQL connections

The current version of One Identity Safeguard for Privileged Sessions (SPS) has the following limitations:

- TDS protocol version 7.3 or later is required.
- Due to the TDS protocol version requirement, Microsoft® SQL Server® 2008, or later, is recommended.
- The Require Gateway Authentication on the SPS Web Interface option in MSSQL Control > Connections does not work in case of MSSQL connections.
- MSSQL server with TCP dynamic port settings is not supported.

You must specify a static TCP port for every instance in the SQL Server Configuration Manager you want to audit. By doing so, you can configure the access to multiple MSSQL instances with multiple connection policies and specify the instances with inband or fixed targets and ports. You can also create and assign different Credential Store policies to check out SQL users’ passwords of the instances.

In the MSSQL client program, always specify the address with the port number of the SPS connection policy you want to connect to.

MSSQL connections

List of endpoints for configuring the policies, options and connection rules of MSSQL connections.

URL

GET https://<IP-address-of-SPS>/api/configuration/mssql
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available settings for configuring for MSSQL connections.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql
```

Response

The following is a sample response received when listing the configuration settings.

For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "authentication_policies",
      "meta": {
        "href": "/api/configuration/mssql/authentication_policies"
      }
    },
    {
      "key": "channel_policies",
      "meta": {
        "href": "/api/configuration/mssql/channel_policies"
      }
    },
    {
      "key": "connections",
      "meta": {
```
Item | Description
--- | ---
connections | List of connection policies.
authentication_policies | List of the default and custom authentication policies.
channel_policies | List of the default and custom channel policies.
options | List of global MSSQL options that affect all connections.
settings_policies | List of protocol-level settings (idle and session timeout). You can create multiple variations, and choose the appropriate one for each connection policy.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**MSSQL connection policies**

Connection policies determine if a server can be accessed from a particular client. Connection policies reference other resources (policies, usergroups, keys) that must be configured and available before creating a connection policy.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/mssql/connections/

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists MSSQL connection policies.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql/connections/

The following command retrieves the properties of a specific policy.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql/connections/<connection-key>

Response

The following is a sample response received when listing MSSQL connection policies.

For details of the meta object, see Message format on page 10.

```
{
   "items": [ 
   {
      "key": "19443158785dee0621437ad",
      "meta": {
         "href": "/api/configuration/mssql/connections/19443158785dee0621437ad"
      }
   },
   "meta": {
      "first": "/api/configuration/mssql/channel_policies",
      "href": "/api/configuration/mssql/connections",
      "last": "/api/configuration/mssql/options",
      "next": "/api/configuration/mssql/options",
      "order": "/api/configuration/mssql/connections/@order",
      "parent": "/api/configuration/mssql",
      "previous": "/api/configuration/mssql/channel_policies",
      "remaining_seconds": 600,
      "transaction": "/api/transaction"
   }
}
```

When retrieving the endpoint of a specific MSSQL Connection Policy, the response is the following.
{
"body": {
"access_control": [],
"active": true,
"channel_database_cleanup": {
"enabled": false
},
"indexing": {
"enabled": true,
"policy": {
"key": "-50000",
"meta": {
"href": "/api/configuration/policies/indexing/-50000"
}
},
"priority": 3
},
"log_audit_trail_downloads": true,
"name": "demo_mssql",
"network": {
"clients": [
"0.0.0.0/0"
],
"ports": [
1433
],
"targets": [
"192.168.1.1/24"
]
},
"override_log_level": {
"enabled": false
},
"policies": {
"aa_plugin": null,
"analytics_policy": {
"key": "20509709385cd578654cdab",
"meta": {
"href":
"/api/configuration/policies/analytics/20509709385cd578654cdab"
}
},
"archive_cleanup_policy": null,
"audit_policy": {
"key": "78101850949e47437dd91d",
"meta": {
"href": "/api/configuration/policies/audit_
policies/78101850949e47437dd91d"

SPS 6.13.0 REST API Reference Guide
MSSQL connections

502


"authentication_policy": {
  "key": "-30700201",
  "meta": {
    "href": "/api/configuration/mssql/authentication_policies/-30700201"
  }
},
"backup_policy": null,
"channel_policy": {
  "key": "-30700102",
  "meta": {
    "href": "/api/configuration/mssql/channel_policies/-30700102"
  }
},
"credential_store": null,
"ldap_server": null,
"settings": {
  "key": "-30700301",
  "meta": {
    "href": "/api/configuration/mssql/settings_policies/-30700301"
  }
},
"usermapping_policy": null,
"rate_limit": {
  "enabled": false
},
"server_address": {
  "custom_dns": {
    "enabled": false
  },
  "selection": "original"
},
"source_address": {
  "selection": "box_address"
},
"transport_security": {
  "selection": "disabled"
},
"web_gateway_authentication": {
  "enabled": false
}
"key": "19443158785dee0621437ad",
"meta": {
  "first": "/api/configuration/mssql/connections/19443158785dee0621437ad",
  "href": "/api/configuration/mssql/connections/19443158785dee0621437ad"
"last": "/api/configuration/mssql/connections/19443158785dee0621437ad",
"next": null,
"parent": "/api/configuration/mssql/connections",
"previous": null,
"remaining_seconds": 600,
"transaction": "/api/transaction"
}
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the connection policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the connection policy.</td>
</tr>
<tr>
<td>access_control</td>
<td>Top level list</td>
<td>Collection of access policies. Access policies define who can authorize and audit a connection.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Set to false to suspend the connection policy. Connection settings are preserved.</td>
</tr>
<tr>
<td>channel_databaseCleanup</td>
<td>Top level item</td>
<td>Configures cleanup of the connection metadata on the connection policy's level.</td>
</tr>
<tr>
<td>days</td>
<td>int</td>
<td>Retention time, in days. Must not exceed the retention time of the archive_cleanup_policy, and the retention time configured in the global settings of the protocol. The global settings of the MSSQL protocol are available at the api/configuration/mssql/options endpoint.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable periodical cleanup of the connection metadata.</td>
</tr>
<tr>
<td>indexing</td>
<td>Top level item</td>
<td>Configures indexing for the connection policy.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable indexing the connections.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>policy</td>
<td>string</td>
<td>References the identifier of the indexing policy. You can configure indexing policies at the /api/configuration/policies/indexing/ endpoint. To modify or add an indexing policy, use the value of the returned key as the value of the policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>priority</td>
<td>int</td>
<td>Specifies the indexing priority for the connection. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Very low priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Low priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Normal (default) priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Very high priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Near real-time priority.</td>
</tr>
<tr>
<td>log_audit_trail_downloads</td>
<td>boolean</td>
<td>Set to true to log audit trail downloads.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the connection policy.</td>
</tr>
<tr>
<td>network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clients</td>
<td>list,</td>
<td>List of client (&quot;from&quot;) IP addresses.</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>ports</td>
<td>list,</td>
<td>List of target ports.</td>
</tr>
<tr>
<td></td>
<td>integer-s</td>
<td></td>
</tr>
<tr>
<td>targets</td>
<td>list,</td>
<td>List of target IP addresses.</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>override_log_level</td>
<td>Top</td>
<td>Specifies the verbosity level of sessions</td>
</tr>
<tr>
<td></td>
<td>level</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>item</td>
<td></td>
<td>handled by this connection policy. The log level of other connection policies is not affected. If disabled, the log level set at the /api/configuration/&lt;protocol&gt;/options endpoint is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- To use the default log level, disable this option:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;override_log_level&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;enabled&quot;: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- To use a custom log level for the connection policy, enable this option and set the log level to use:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;override_log_level&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;log_level&quot;: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>policies</td>
<td></td>
<td>List of policies referenced by the connection policy.</td>
</tr>
<tr>
<td>aa_plugin</td>
<td>string</td>
<td>References the identifier of the AA plug-in. You can configure AA plug-ins at the /api/configuration/plugins/aa/ endpoint. To modify or add an AA plug-in, use the value of the returned key as the value of the aa_plugin element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>analytics_policy</td>
<td>string</td>
<td>References the identifier of the analytics policy. You can configure analytics policies at the /api/configuration/analytics/ endpoint. To add or modify an analytics policy, use the value of the returned key as the value of the analytics element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>archive</td>
<td>string</td>
<td>References the identifier of the</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cleanup_policy</td>
<td></td>
<td>archive/cleanup policy. You can configure archive and cleanup policies at the /api/configuration/policies/archive_cleanup_policies/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an archive/cleanup policy, use the value of the returned key as the value of the archive_cleanup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>audit_policy</td>
<td>string</td>
<td>Cannot be null. References the identifier of the audit policy. You can configure audit policies at the /api/configuration/policies/audit_policies/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an audit policy, use the value of the returned key as the value of the audit_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>authentication_policy</td>
<td>string</td>
<td>Cannot be null. References the identifier of the authentication policy. Note that currently you cannot create or modify MSSQL Authentication Policies using the REST API. Use the web UI instead.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an authentication policy, use the value of the returned key as the value of the authentication_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>backup_policy</td>
<td>string</td>
<td>References the identifier of the backup policy. You can configure backup policies at the /api/configuration/policies/backup_policies/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a backup policy, use the value of the returned key as the value of the backup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>channel_policy</td>
<td>string</td>
<td>References the identifier of the channel policy. The value of this option cannot be null.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a channel policy, use the value of the returned key as the value of the channel_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can configure MSSQL channel policies at the /api/configuration/mssql/channel_policies/ endpoint.</td>
</tr>
<tr>
<td>credential_store</td>
<td>string</td>
<td>References the identifier of the credential store.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can configure credential stores at the /api/configuration/policies/credentialstores/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a credential store, use the value of the returned key as the value of the credential_store element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>ldap_server</td>
<td>string</td>
<td>References the identifier of the LDAP server. You can configure LDAP servers at the /api/configuration/policies/ldap_servers/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an LDAP server, use the value of the returned key as the value of the ldap_server element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>settings</td>
<td>string</td>
<td>References the identifier of the settings policy. The value of this option cannot be null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a settings policy for this protocol, use the value of the returned key as the value of the settings element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>usermapping_policy</td>
<td>string</td>
<td>References the identifier of a Usermapping Policy. You can configure Usermapping Policies at the /api/configuration/policies/usermapping_policies/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a Usermapping Policy,</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>rate_limit</td>
<td>Top level element</td>
<td>Connection rate limit.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to provide a connection rate limit.</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>The number of connections (per minute) that are allowed in the connection policy.</td>
</tr>
<tr>
<td>server_address</td>
<td>Top level item</td>
<td>Defines the address where the clients connect to.</td>
</tr>
<tr>
<td>source_address</td>
<td>Top level element</td>
<td>Allows you to configure Source Network Address Translation (SNAT) on the server side of SPS. SNAT determines the IP address SPS uses in the server-side connection. The target server will see the connection coming from this address.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures Source Network Address Translation. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• box_address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• original</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used if the value of the selection element is set to fix. The IP address to use as the source address in server-side connections.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if the value of the selection element is set to fix. The IP address to use as the source address in server-side connections.</td>
</tr>
<tr>
<td>transport_</td>
<td>Top</td>
<td>Configures the encryption used in the</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>security</td>
<td>level element</td>
<td>sets the security level for element sessions.</td>
</tr>
<tr>
<td>certificate</td>
<td>JSON object</td>
<td>selects the certificate to show to the peers. You have the following options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Use the same certificate for each client:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select this option if you want to use the same certificate for every peer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that you must reference a certificate that includes its private key</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that you have already uploaded to SPS. For details, see</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="#">Certificates stored on SPS</a> on page 287.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;certificate&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;fix&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;x509_identity&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Generate a certificate for each client:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select this option if you want to generate a certificate for each client.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that you must reference a Signing CA that you have already configured</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on SPS. For details, see <a href="#">Signing CA policies</a> on page 406.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;certificate&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;generate&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;signing_ca&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;1904188625a843f11d30a5&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>selection</td>
<td>disabled</td>
<td>configures the encryption used in the sessions.</td>
</tr>
<tr>
<td>selection</td>
<td>enabled</td>
<td>configures the encryption used in the sessions.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>By default, SPS accepts any certificate shown by the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>To verify the certificate of the destination server, configure and reference a Trusted CA list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>When gateway authentication is required.</td>
</tr>
</tbody>
</table>

- **Element**: `transport_security`
- **Type**: `JSON object`
- **Description**:

  ```json
  "transport_security": {
    "selection": "disabled"
  },
  ``

- **Element**: `tls`
- **Type**: `Enabled TLS-encryption. Note that you must also set the certificate and server_certificate_check options.

  ```json
  "transport_security": {
    "certificate": {
      "selection": "generate",
      "signing_ca": "1960594886d07511f09eca"
    },
    "selection": "tls",
    "server_certificate_check": {
      "enabled": true,
      "trusted_ca": "1241814345d074ef0d1ded7"
    }
  },
  ``

- **Element**: `server_certificate_check`
- **Type**: `Enabled`
- **Description**: By default, SPS accepts any certificate shown by the server.

  ```json
  "server_certificate_check": {
    "enabled": false
  },
  ``

- **Element**: `web_gateway_`
- **Type**: `Top`
- **Description**: When gateway authentication is required.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>level item</td>
<td>for a connection, the user must authenticate on SPS as well. This additional authentication can be performed out-of-band on the SPS web interface for every protocol.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable additional gateway authentication on the SPS web interface.</td>
</tr>
<tr>
<td>groups</td>
<td>list, string</td>
<td>By default, any user can perform gateway authentication for the connections. You can restrict authentication to members of specific usergroups. Define the usergroups at the /api/configuration/aaa/local_database/groups/ endpoint, and list the name of each group here.</td>
</tr>
<tr>
<td>require_same_ip</td>
<td>boolean</td>
<td>Set to true to only accept web gateway authentication from the same host that initiated the connection.</td>
</tr>
</tbody>
</table>

### Elements of access control

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizer</td>
<td>string</td>
<td>The usergroup (local or LDAP) who can authorize or audit the connection. Local usergroups can be added or modified at the /api/configuration/aaa/local_database/groups/ endpoint.</td>
</tr>
<tr>
<td>permission</td>
<td>string</td>
<td>Defines the permissions of the authorizer usergroup. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- audit&lt;br&gt;The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- authorize&lt;br&gt;The usergroup with the authorize permission can authorize connection requests.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- audit_and_authorize&lt;br&gt;The usergroup with the audit_and_</td>
</tr>
</tbody>
</table>
### Elements of access_control

<table>
<thead>
<tr>
<th>Elements of access_control</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorize</td>
<td>permission</td>
<td>can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.</td>
</tr>
<tr>
<td>require_different_ip</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different IP addresses.</td>
</tr>
<tr>
<td>require_different_username</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different usernames.</td>
</tr>
<tr>
<td>subject</td>
<td>Top level item</td>
<td>Defines the subjects of the access control policy.</td>
</tr>
</tbody>
</table>

#### group
- String
- The usergroup (local or LDAP) that is subject to the access control policy.
- Local usergroups can be added or modified at the `/api/configuration/aaa/local_database/groups/` endpoint.

#### selection
- String
- Possible values:
  - everybody
    - Every user is subject to the access control policy.
  - only
    - Requires the group element.
    - Members of the usergroup specified in the group element are subject to the access control policy.

### Elements of server_address

<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>custom_dns</td>
<td>string</td>
<td>Configures a DNS server that is used to reverse-resolve the hostname if the Channel Policy contains the address of the target as a hostname instead of an IP address. By default, this is disabled and SPS uses the DNS server set in the...</td>
</tr>
<tr>
<td>Elements of server_address</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>/api/configuration/network/dns endpoint.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• To use the default DNS, disable this option:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;server_address&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;custom_dns&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;enabled&quot;: false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• To use a custom DNS, enable this option and set the IP address of the domain name server to use:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;server_address&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;custom_dns&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;enabled&quot;: true,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;server&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;192.168.1.1&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
<td></td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures the address where the clients connect to. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• original</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Connect to the same address specified by the client.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• nat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Perform a network address translation on the target address.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the network element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the address and port elements.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• inband</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extract the address of the server from the username.</td>
</tr>
</tbody>
</table>
| | | Must be used with the domains
<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Optional elements: exception_domains, dns_server, and dns_suffixes.</td>
</tr>
<tr>
<td>network</td>
<td>string</td>
<td>Must be used if selection is set to nat.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The target address in IP/prefix format. Example: &quot;10.20.30.40/24&quot;.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if selection is set to fix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The IP address of the target server.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>Must be used if selection is set to fix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The port of the target server.</td>
</tr>
<tr>
<td>domains</td>
<td>Top level list</td>
<td>Must be used if selection is set to inband.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level item</td>
<td>Lists the address ranges that are included in the connection policy.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies if the target address range is provided as a domain or as an IP range. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the target address is an IP range.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• domain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the target address is a domain.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The address range of the target server(s).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port of the target server(s).</td>
</tr>
<tr>
<td>exception_domains</td>
<td>Top level list</td>
<td>Can only be used if selection is set to inband.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lists the address ranges that are excluded from the connection policy.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level</td>
<td>Contains the excluded address range.</td>
</tr>
<tr>
<td><strong>Elements of server_address</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td>item</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>selection</strong></td>
<td>string</td>
<td>Specifies if the excluded address(es) are provided as a domain or as an IP range. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the excluded address is an IP range.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- domain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the excluded address is a domain.</td>
</tr>
<tr>
<td><strong>value</strong></td>
<td>string</td>
<td>The excluded address(es). Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
<tr>
<td><strong>port</strong></td>
<td>int</td>
<td>The excluded port.</td>
</tr>
<tr>
<td><strong>dns_server</strong></td>
<td>string</td>
<td>Can only be used if selection is set to inband.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IP address or the hostname of the domain name server used to resolve the address of the target server.</td>
</tr>
<tr>
<td><strong>dns_suffixes</strong></td>
<td>list,</td>
<td>Can only be used if selection is set to inband.</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td>If the clients do not include the domain name when addressing the server (for example they use username@server instead of <a href="mailto:username@server.example.com">username@server.example.com</a>), SPS can automatically add domain information (for example example.com). You can add multiple domain names. SPS attempts to resolve the target address by appending the domain names in the provided order, and uses the first successfully resolved address to establish the connection.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Add a MSSQL connection policy**

To add a MSSQL connection policy, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new MSSQL connection policy.**
   
   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/mssql/connections/` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

   If the POST request is successful, the response includes the key of the new MSSQL connection policy. For example:

   ```json
   {
      "key": "a99be49b-b0a2-4cf9-b70d-f9ea1f9ea188f",
      "meta": {
         "href": "/api/configuration/mssql/connections/a99be49b-b0a2-4cf9-b70d-f9ea1f9ea188f",
         "parent": "/api/configuration/mssql/connections",
         "transaction": "/api/transaction"
      }
   }
   ```

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.
Modify a MSSQL connection policy

To modify a MSSQL connection policy, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the connection policy.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/mssql/connections/<key-of-the-object>` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

MSSQL channels

The available MSSQL channel types and their functionalities are described below. For details on configuring channel policies, see [Channel policy](#).

<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mssql</td>
<td>Yes</td>
<td><strong>mssql</strong>: Enables access to the MSSQL server. This channel must be enabled for MSSQL to work.</td>
</tr>
</tbody>
</table>

Channel-specific actions:

- `content_policy` reference: The ID of the Content policy to apply to the connection.

For example:

```
"actions": {
  "audit": true,
  "four_eyes": true,
  "content_policy": {
    "key": "433849548566ab327522e6",
    "meta": {
      "href": "/api/configuration/policies/content_policies/44287216854f482e7f2b24"
    }
  }
},
```
MSSQL authentication policies

Lists the configured authentication methods that can be used in a connection. Each connection policy uses an authentication policy to determine how the client can authenticate on the SPS gateway.

URL

GET https://<IP-address-of-SPS>/api/configuration/mssql/authentication_policies

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists MSSQL authentication policies.

`curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql/authentication_policies`

The following command retrieves the properties of a specific policy.

`curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql/authentication_policies<object-id>`

Response

The following is a sample response received when listing MSSQL authentication policies. For details of the meta object, see Message format on page 10.
When retrieving the endpoint of a specific policy, the response is the following.

```json
{
    "body": {
        "backend": {
            "selection": "ldap"
        },
        "name": "mssql_auth_policy_with_ldap"
    }
}
```

### Element Type Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element</td>
<td>Contains the elements of the policy.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>backend</td>
<td>Top level item</td>
<td>Client-side gateway authentication settings. The value of selection defines which authentication method is used.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the authentication method for client-side gateway authentication. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• none</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disables client-side gateway authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ldap</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses the LDAP server selected for the connection policy. LDAP servers can be configured in the \api/configuration/policies/ldap_servers endpoint).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To use this option, you must also configure the certificate, password, and public_key elements.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• local</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses the local user database configured in the \api/configuration/policies/user_databases/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To use this option, you must also configure the user_database element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• radius</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses one or more Radius servers for authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To use this option, you must also configure the authentication_protocol and servers elements.</td>
</tr>
<tr>
<td>servers</td>
<td>Top level list</td>
<td>Only if selection is set to radius</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Defines the properties of the RADIUS servers used for client-side authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A valid list item consists of the address, port and shared_secret elements.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>authentication_protocol</td>
<td>Top level item</td>
<td>Only if selection is set to radius RADIUS setting. Set to pap to use the Password Authentication Protocol. To use the Challenge-Handshake Authentication Protocol, set it to chap.</td>
</tr>
<tr>
<td>user_database</td>
<td>string</td>
<td>Only if selection is set to local References the key of the local user database. You can configure local user databases at the /api/configuration/policies/user_databases/ endpoint. To modify or add a local user database, use the value of the returned key as the value of the user_database element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>timeout</td>
<td>integer (seconds)</td>
<td>Specify the time remaining until a successful gateway authentication times out.</td>
</tr>
<tr>
<td>keepalive</td>
<td>boolean</td>
<td>Set to true to avoid interruptions for active HTTP sessions. Active HTTP sessions can extend the gateway authentication beyond the configured timeout.</td>
</tr>
</tbody>
</table>

**Elements of servers**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>Top level element</td>
<td>Defines the address of a RADIUS server.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Required child of the address element. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ip</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains the IP of the RADIUS server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fqdn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains the FQDN of the RADIUS server.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The IP or the FQDN address of the RADIUS server.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number of the RADIUS server.</td>
</tr>
</tbody>
</table>
| shared_ | string | References the key of the shared secret for the
<table>
<thead>
<tr>
<th>Elements of servers</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>secret</td>
<td></td>
<td>RADIUS server. You can configure shared secrets at the <code>/api/configuration/passwords/</code> endpoint. To modify or add a shared secret, use the value of the returned key as the value of the shared_secret element, and remove any child elements (including the key). Alternatively, you can include the new password as plain text.</td>
</tr>
</tbody>
</table>

```json
"shared_secret": {
  "plain": "<new-password>
}
```

### Examples:

**Querying base authentication policy without gateway authentication:**

```json
{
  "key": "-304002001",
  "body": {
    "name": "base",
    "backend": {
      "selection": "none"
    }
  }
}
```

**Querying authentication policy with LDAP backend:**

```json
{
  "key": "mssql-auth-pol-2",
  "body": {
    "name": "mssql_ldap",
    "backend": {
      "selection": "ldap",
      "timeout": 3600,
      "keepalive": true
    }
  }
}
```

**Querying authentication policy with local backend:**

---

**ONE IDENTITY**

© 2023 One Identity

SPS 6.13.0 REST API Reference Guide

MSSQL connections

523
Querying authentication policy with RADIUS backend:

```json
{
  "key": "mssql-auth-pol-4",
  "body": {
    "name": "mssql_radius",
    "backend": {
      "selection": "radius",
      "servers": [ {
        "address": {
          "selection": "ip",
          "value": "1.2.3.4"
        },
        "port": 1812,
        "shared_secret": {
          "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
          "meta": { "href": "/api/configuration/passwords#XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" }
        }
      } ],
      "authentication_protocol": "pap",
      "timeout": 3600,
      "keepalive": true
    }
  }
}
```
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Add a MSSQL authentication policy

To add a MSSQL authentication policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new policy.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/mssql/authentication_policies/ endpoint. You can find a detailed description of the available parameters listed in MSSQL authentication policies.

   If the POST request is successful, the response includes the key of the new policy. For example:

   ```json
   {
     "key": "6f924f39-e4c9-4b0f-8018-8842e2115ebd",
     "meta": {
       "href": "/api/configuration/mssql/authentication_policies/6f924f39-e4c9-4b0f-8018-8842e2115ebd"
     }
   }
   ```
3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

**Modify a MSSQL authentication policy**

To modify a MSSQL authentication policy, you have to:

1. **Open a transaction.**
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the policy.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/mssql/authentication_policies/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in MSSQL authentication policies.

3. **Commit your changes.**
   For more information, see Commit a transaction on page 31.

**Global MSSQL options**

List of options that affect all MSSQL connections.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/mssql/options
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists global MSSQL options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql/options
```

**Response**

The following is a sample response received when listing global MSSQL options.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "audit": {
            "cleanup": {
                "enabled": false
            },
            "timestamping": {
                "selection": "local",
                "signing_interval": 30
            }
        },
        "service": {
            "enabled": true,
            "log_level": 4
        }
    }
}
```


```json
"key": "options",
"meta": {
  "first": "/api/configuration/mssql/channel_policies",
  "href": "/api/configuration/mssql/options",
  "last": "/api/configuration/mssql/options",
  "next": null,
  "parent": "/api/configuration/mssql",
  "previous": "/api/configuration/mssql/channel_policies",
  "transaction": "/api/transaction"
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Top level item</td>
<td>Contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level item</td>
<td>Contains the elements of the global MSSQL options.</td>
</tr>
<tr>
<td>audit</td>
<td>Top level item</td>
<td>Contains settings for timestamping and cleanup.</td>
</tr>
<tr>
<td>service</td>
<td>Top level item</td>
<td>Global setting to enable MSSQL connections, and specify the logging detail.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable MSSQL connections.</td>
</tr>
<tr>
<td>log_level</td>
<td>int</td>
<td>Defines the logging detail of MSSQL connections.</td>
</tr>
</tbody>
</table>

**Elements of audit**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanup</td>
<td>Top level item</td>
<td>Global retention settings for MSSQL connection metadata. To configure</td>
</tr>
<tr>
<td></td>
<td></td>
<td>retention time for a specific connection policy, use the archive_cleanup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>policy element at the endpoint of the policy instead.</td>
</tr>
<tr>
<td>channel_database_cleanup_days</td>
<td>int</td>
<td>Global retention time for the metadata of MSSQL connections, in days. Must exceed the retention time of the archiving policy (or policies) used for MSSQL connections, and the connection-specific database cleanup times (if configured).</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>To enable the global cleanup of MSSQL connections.</td>
</tr>
</tbody>
</table>

---

SPS 6.13.0 REST API Reference Guide

MSSQL connections
<table>
<thead>
<tr>
<th>Elements of audit</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamping</td>
<td>Top level item</td>
<td>Global timestamping settings for MSSQL connections.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures local or remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Set local to use SPS for timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Set remote to configure a remote timestamping server.</td>
</tr>
<tr>
<td>server_url</td>
<td>string</td>
<td>Required for remote timestamping. The URL of the timestamping server.</td>
</tr>
<tr>
<td>oid</td>
<td>Top level item</td>
<td>The Object Identifier of the policy used for timestamping.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required for remote timestamping. Set to true to configure the Object Identifier of the timestamping policy on the timestamping remote server.</td>
</tr>
<tr>
<td>policy_oid</td>
<td>string</td>
<td>Required if the oid is enabled. The Object Identifier of the timestamping policy on the remote timestamping server.</td>
</tr>
<tr>
<td>signing_interval</td>
<td>int</td>
<td>Time interval for timestamping open connections, in seconds.</td>
</tr>
</tbody>
</table>

**Examples:**

Set SPS as the timestamping server:

```json
{
    "audit": {
        "cleanup": {
            "enabled": false
        },
        "timestamping": {
```
Enable cleanup, and set it to occur every 10 days:

```json
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
      "selection": "local",
      "signing_interval": 30
    }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```json
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
      "oid": {
        "enabled": false
      },
      "selection": "remote",
      "server_url": "<url-of-timestamping-server>",
      "signing_interval": 30
    }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```
Modify global MSSQL settings

To modify global MSSQL settings, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the global MSSQL settings endpoint.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/mssql/options endpoint. You can find a detailed description of the available parameters listed in [Element](#). The elements of the audit item are described in [Elements of audit](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### MSSQL settings policies

MSSQL settings policies define protocol-level settings for idle and session timeout. You can create multiple policies, and choose the appropriate one for each MSSQL connection.

#### URL

```
GET https://<IP-address-of-SPS>/api/configuration/mssql/settings_policies
```

#### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS.
Sample request

The following command lists MSSQL settings policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql/settings_policies
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/mssql/settings_policies/<policy-id>
```

Response

The following is a sample response received when listing MSSQL settings policies.

For details of the meta object, see Message format on page 10.

```json
{
   "items": [
      {
         "key": "-3040010",
         "meta": {
            "href": "/api/configuration/mssql/settings_policies/-3040010"
         }
      }
   ],
   "meta": {
      "first": "/api/configuration/mssql/channel_policies",
      "href": "/api/configuration/mssql/settings_policies",
      "last": "/api/configuration/mssql/settings_policies",
      "next": null,
      "parent": "/api/configuration/mssql",
      "previous": "/api/configuration/mssql/options",
      "transaction": "/api/transaction"
   }
}
```

When retrieving the endpoint of a specific policy, the response is the following.
```json
{
  "body": {
    "client_tls_security_settings": {
      "cipher_strength": {
        "selection": "recommended"
      },
      "minimum_tls_version": "TLSv1_2"
    },
    "name": "default",
    "server_tls_security_settings": {
      "cipher_strength": {
        "selection": "recommended"
      },
      "minimum_tls_version": "TLSv1_2"
    },
    "session_timeout": 900,
    "timeout": 300
  },
  "key": "-3040010",
  "meta": {
    "first": "/api/configuration/mssql/settings_policies/-3040010",
    "href": "/api/configuration/mssql/settings_policies/-3040010",
    "last": "/api/configuration/mssql/settings_policies/-3040010",
    "next": null,
    "parent": "/api/configuration/mssql/settings_policies",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>The elements of the MSSQL settings policy.</td>
</tr>
<tr>
<td>body (string)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>client_tls_security_settings</td>
<td>JSON object</td>
<td>Configures TLS security settings on the client side.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the MSSQL settings policy. Cannot contain whitespace.</td>
</tr>
<tr>
<td>server_tls_security_settings</td>
<td>JSON object</td>
<td>Configures TLS security settings on the server side.</td>
</tr>
<tr>
<td>timeout</td>
<td>int</td>
<td>Idle timeout, in seconds. Note that the SPS web UI displays the same value in seconds.</td>
</tr>
</tbody>
</table>
### Elements of client_tls_security_settings and server_tls_security_settings

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cipher_strength</td>
<td>JSON object</td>
<td>Specifies the cipher string OpenSSL will use.</td>
</tr>
<tr>
<td>custom_cipher</td>
<td>string</td>
<td>The list of ciphers you want to permit SPS to use in the connection. For more details on customizing this list, check the 'openssl-ciphers' manual page on your SPS appliance.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies the cipher string OpenSSL will use. The following settings options are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- recommended: this setting only uses ciphers with adequate security level.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- custom: this setting allows you to specify the list of ciphers you want to permit SPS to use in the connection. This setting is only recommended to ensure compatibility with older systems. For more details on customizing this list, check the 'openssl-ciphers' manual page on your SPS appliance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: ALL:!aNULL:@STRENGTH</td>
</tr>
<tr>
<td>minimum_tls_version</td>
<td>string</td>
<td>Specifies the minimal TLS version SPS will offer during negotiation. The following settings options are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- TLSv1_2: this setting only offers TLS version 1.2 during the negotiation. This is the recommended setting.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- TLSv1_1: this setting offers TLS version 1.1 and later versions during the negotiation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- TLSv1_0: this setting offers TLS version 1.0 and later versions during the negotiation.</td>
</tr>
</tbody>
</table>

### Add MSSQL settings policies

To add a settings policy, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.
2. **Create the JSON object for the new policy.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/mssql/settings_policies/ endpoint. You can find a detailed description of the available parameters listed in Element.

   If the POST request is successful, the response includes the key of the new policy. For example:

   ```json
   {
     "key": "3848c708-2e1d-4463-b232-0c8c5875ff55",
     "meta": {
       "href": "/api/configuration/mssql/settings_policies/3848c708-2e1d-4463-b232-0c8c5875ff55",
       "parent": "/api/configuration/mssql/settings_policies",
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

**Modify MSSQL settings policies**

To modify a settings policy, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the policy.**

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/mssql/settings_policies/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
RDP connections

List of endpoints for configuring the policies, options and connection rules of RDP connections.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/rdp

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the available settings for configuring for RDP connections.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/rdp
```
Response

The following is a sample response received when listing the configuration settings.
For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "key": "channel_policies",
            "meta": {
                "href": "/api/configuration/rdp/channel_policies"
            }
        },
        {
            "key": "connections",
            "meta": {
                "href": "/api/configuration/rdp/connections"
            }
        },
        {
            "key": "domain_membership",
            "meta": {
                "href": "/api/configuration/rdp/domain_membership"
            }
        },
        {
            "key": "options",
            "meta": {
                "href": "/api/configuration/rdp/options"
            }
        },
        {
            "key": "settings_policies",
            "meta": {
                "href": "/api/configuration/rdp/settings_policies"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/aaa",
        "href": "/api/configuration/rdp",
        "last": "/api/configuration/x509",
        "next": "/api/configuration/reporting",
        "parent": "/api/configuration",
        "previous": "/api/configuration/private_keys",
        "transaction": "/api/transaction"
    }
}
```
## Item Description

- **channel_policies**
  List of the default and custom channel policies.

- **connections**
  List of connection policies.

- **domain_membership**
  Domain membership configuration. Prerequisite for configuring Credential Security Service Provider / Network Layer Authentication.

- **options**
  List of global RDP options that affect all connections.

- **settings_policies**
  List of protocol-level settings (timeout, display, protocol version, and authentication). You can create multiple variations, and choose the appropriate one for each connection policy.

## Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

## RDP connection policies

Connection policies determine if a server can be accessed from a particular client. Connection policies reference other resources (policies, usergroups, keys) that must be configured and available before creating a connection policy.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/rdp/connections/
```
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format)..

Sample request

The following command lists RDP connection policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/rdp/connections/
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/rdp/connections/<connection-key>
```

Response

The following is a sample response received when listing RDP connection policies. For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "12932832285a830b4d2f5d7",
            "meta": {
                "href": "/api/configuration/rdp/connections/12932832285a830b4d2f5d7"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/rdp/channel_policies",
```
When retrieving the endpoint of a specific RDP connection policy, the response is the following.

```
{
  "body": {
    "access_control": [
      {
        "authorizer": "reporting",
        "permission": "audit_and_authorize",
        "require_different_ip": true,
        "require_different_username": true,
        "subject": {
          "selection": "everybody"
        }
      }
    ],
    "active": true,
    "channel_database_cleanup": {
      "days": 550,
      "enabled": true
    },
    "indexing": {
      "enabled": true,
      "policy": {
        "key": "-50000",
        "meta": {
          "href": "/api/configuration/policies/indexing/-50000"
        }
      },
      "priority": 3
    },
    "log_audit_trail_downloads": true,
    "name": "rdp_demo",
    "network": {
      "clients": [
        "0.0.0.0/0"
      ],
      "ports": [
        3389
      ]
    }
  }
}
```
{
  "targets": [
    "10.30.255.28/24"
  ]
},
"policies": {
  "aa_plugin": null,
  "analytics_policy": null,
  "archive_cleanup_policy": {
    "key": "1854671967571b9063c4c82",
    "meta": {
      "href": "/api/configuration/policies/archive_cleanup_policies/1854671967571b9063c4c82"
    }
  },
  "audit_policy": {
    "key": "78101850949e47437dd91d",
    "meta": {
      "href": "/api/configuration/policies/audit_policies/78101850949e47437dd91d"
    }
  },
  "backup_policy": {
    "key": "512524636571b903540804",
    "meta": {
      "href": "/api/configuration/policies/backup_policies/512524636571b903540804"
    }
  },
  "channel_policy": {
    "key": "-20200",
    "meta": {
      "href": "/api/configuration/rdp/channel_policies/-20200"
    }
  },
  "credential_store": {
    "key": "505008562571b936560254",
    "meta": {
      "href": "/api/configuration/policies/credentialstores/505008562571b936560254"
    }
  },
  "ldap_server": {
    "key": "250588254571b931066482",
    "meta": {
      "href": "/api/configuration/policies/ldap_servers/250588254571b931066482"
    }
  }
}
"settings": {
  "key": "-301",
  "meta": {
    "href": "/api/configuration/rdp/settings_policies/-301"
  }
},
  "usermapping_policy": null
},
  "rate_limit": {
    "enabled": false
},
  "remote_desktop_gateway": {
    "enabled": false
},
  "server_address": {
    "address": "10.30.255.70",
    "port": 3389,
    "selection": "fix"
},
  "server_certificate_check": {
    "enabled": false
},
  "source_address": {
    "selection": "box_address"
},
  "transport_security": {
    "certificate": {
      "selection": "self_signed"
    },
    "legacy_fallback": false,
    "selection": "tls"
},
  "web_gateway_authentication": {
    "enabled": false
  }
},
  "key": "12932832285a830b4d2f5d7",
  "meta": {
    "first": "/api/configuration/rdp/connections/12932832285a830b4d2f5d7",
    "href": "/api/configuration/rdp/connections/12932832285a830b4d2f5d7",
    "last": "/api/configuration/rdp/connections/12932832285a830b4d2f5d7",
    "next": null,
    "parent": "/api/configuration/rdp/connections",
    "previous": null,
    "remaining_seconds": 600,
    "transaction": "/api/transaction"
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the connection policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the connection policy.</td>
</tr>
<tr>
<td>access_control</td>
<td>Top level list</td>
<td>Collection of access policies. Access policies define who can authorize and audit a connection.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Set to false to suspend the connection policy. Connection settings are preserved.</td>
</tr>
<tr>
<td>channel_database_cleanup</td>
<td>Top level item</td>
<td>Configures cleanup of the connection metadata on the connection policy's level.</td>
</tr>
<tr>
<td>days</td>
<td>int</td>
<td>Retention time, in days. Must not exceed the retention time of the archive_cleanup_policy, and the retention time configured in the global settings of the protocol. The global settings of the SSH protocol are available at the api/configuration/ssh/options endpoint.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable periodical cleanup of the connection metadata.</td>
</tr>
<tr>
<td>indexing</td>
<td>Top level item</td>
<td>Configures indexing for the connection policy.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable indexing the connections.</td>
</tr>
<tr>
<td>policy</td>
<td>string</td>
<td>References the identifier of the indexing policy. You can configure indexing policies at the /api/configuration/policies/indexing/endpoint. To modify or add an indexing policy, use the value of the returned key as the value of the policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>priority</td>
<td>int</td>
<td>Specifies the indexing priority for the connection. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Very low priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Low priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Normal (default) priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Very high priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Near real-time priority.</td>
</tr>
<tr>
<td>log_audit_trail_downloads</td>
<td>boolean</td>
<td>Set to true to log audit trail downloads.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the connection policy.</td>
</tr>
<tr>
<td>network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clients</td>
<td>list,</td>
<td>List of client (&quot;from&quot;) IP addresses.</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>ports</td>
<td>list,</td>
<td>List of target ports.</td>
</tr>
<tr>
<td></td>
<td>integers</td>
<td></td>
</tr>
<tr>
<td>targets</td>
<td>list,</td>
<td>List of target IP addresses.</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>override_log_level</td>
<td>Top</td>
<td>Specifies the verbosity level of sessions handled by this connection policy.</td>
</tr>
<tr>
<td></td>
<td>level</td>
<td>The log level of other connection policies is not affected. If disabled, the</td>
</tr>
<tr>
<td></td>
<td>item</td>
<td>log level set at the /api/configuration/&lt;protocol&gt;/options endpoint is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• To use the default log level, disable this option:</td>
</tr>
</tbody>
</table>

SPS 6.13.0 REST API Reference Guide
RDP connections
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policies</td>
<td>Top level item</td>
<td>List of policies referenced by the connection policy.</td>
</tr>
<tr>
<td>aa_plugin</td>
<td>string</td>
<td>References the identifier of the AA plug-in. You can configure AA plug-ins at the /api/configuration/plugins/aa/ endpoint. To modify or add an AA plug-in, use the value of the returned key as the value of the aa_plugin element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>analytics_policy</td>
<td>string</td>
<td>References the identifier of the analytics policy. You can configure analytics policies at the /api/configuration/analytics/ endpoint. To add or modify an analytics policy, use the value of the returned key as the value of the analytics element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>archive_cleanup_policy</td>
<td>string</td>
<td>References the identifier of the archive/cleanup policy. You can configure archive and cleanup policies at the /api/configuration/policies/archive_cleanup_policies/ endpoint. To modify or add an archive/cleanup policy, use the value of the returned key as the value of the archive_cleanup_policy element, and remove any child elements (including the key).</td>
</tr>
</tbody>
</table>

- To use a custom log level for the connection policy, enable this option and set the log level to use:

```json
"override_log_level": {
  "enabled": true,
  "log_level": 5
},
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>audit_policy</td>
<td>string</td>
<td>Cannot be null. References the identifier of the audit policy. You can configure audit policies at the <code>/api/configuration/policies/audit_policies/</code> endpoint. To modify or add an audit policy, use the value of the returned key as the value of the audit_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>backup_policy</td>
<td>string</td>
<td>References the identifier of the backup policy. You can configure backup policies at the <code>/api/configuration/policies/backup_policies/</code> endpoint. To modify or add a backup policy, use the value of the returned key as the value of the backup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>channel_policy</td>
<td>string</td>
<td>References the identifier of the channel policy. The value of this option cannot be null. To modify or add a channel policy, use the value of the returned key as the value of the channel_policy element, and remove any child elements (including the key). You can configure RDP channel policies at the <code>/api/configuration/rdp/channel_policies/</code> endpoint.</td>
</tr>
<tr>
<td>credential_store</td>
<td>string</td>
<td>References the identifier of the credential store. You can configure credential stores at the <code>/api/configuration/policies/credential_stores/</code> endpoint. To modify or add a credential store, use the value of the returned key as the value of the credential_store element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>ldap_server</td>
<td>string</td>
<td>References the identifier of the LDAP server. You can configure LDAP servers at the SPS 6.13.0 REST API Reference Guide RDP connections</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>settings</td>
<td>string</td>
<td>References the identifier of the settings policy. The value of this option cannot be null. To modify or add a settings policy for this protocol, use the value of the returned key as the value of the settings element, and remove any child elements (including the key). You can configure RDP settings policies at the /api/configuration/ssh/settings_policies/ endpoint.</td>
</tr>
<tr>
<td>usermappin g_policy</td>
<td>string</td>
<td>References the identifier of a Usermapping Policy. You can configure Usermapping Policies at the /api/configuration/policies/usermappin g_policies/ endpoint. To modify or add a Usermapping Policy, use the value of the returned key as the value of the usermapping_policies element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>rate_limit</td>
<td>Top level element</td>
<td>Connection rate limit.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to provide a connection rate limit.</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>The number of connections (per minute) that are allowed in the connection policy.</td>
</tr>
<tr>
<td>remote_desktop_gateway</td>
<td>Top level element</td>
<td>Configure SPS to act as a Remote Desktop Gateway. Otherwise, simply disable this option:</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>server_address</td>
<td>Top level item</td>
<td>Defines the address where the clients connect to.</td>
</tr>
<tr>
<td>server_certificate_check</td>
<td>Top level item</td>
<td>By default, SPS accepts any certificate shown by the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;server_certificate_check&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;enabled&quot;: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To verify the certificate of the destination server, configure and reference a Trusted CA list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;server_certificate_check&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;trusted_ca&quot;: &quot;9106862955a844051d7bf6&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>source_address</td>
<td>Top level element</td>
<td>Allows you to configure Source Network Address Translation (SNAT) on the server side of SPS. SNAT determines the IP address SPS uses in the server-side connection. The target server will see the connection coming from this address.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures Source Network Address Translation. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- box_address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default. Uses the network address of the logical interface of SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- original</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses the IP address of the client, as seen by SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- fix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses a fixed address when connecting to the remote server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the address</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if the value of the selection element is set to fix. The IP address to use as the source address in server-side connections.</td>
</tr>
<tr>
<td>transport_security</td>
<td>Top level element</td>
<td>Configures the encryption used in the sessions.</td>
</tr>
<tr>
<td>certificate</td>
<td>JSON object</td>
<td>Selects the certificate to show to the peers. You have the following options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Use a self-signed certificate:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select this option if you want to enable TLS-encryption, but you do not have a certificate that is generated by an external CA, or a signing CA.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;certificate&quot;: { &quot;selection&quot;: &quot;self_signed&quot; }</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Use the same certificate for each client:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select this option if you want to use the same certificate for every peer. Note that you must reference a certificate that includes its private key that you have already uploaded to SPS. For details, see Certificates stored on SPS on page 287.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;certificate&quot;: { &quot;selection&quot;: &quot;fix&quot;, &quot;x509_identity&quot;: &quot;893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a&quot; }</td>
</tr>
</tbody>
</table>
## Generate a certificate for each client:
Select this option if you want to generate a certificate for each client. Note that you must reference a Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 406.

```
"certificate": {
    "selection": "generate",
    "signing_ca": "1904188625a843f11d30a5"
}
```

### legacy_fallback
- **Type:** boolean
- **Description:** Set to true to permit the clients to disable TLS encryption and use only the Legacy RDP Security Layer (also known as: Standard RDP Security). You might want to do this if you are experiencing compatibility issues. For example, when you attempt to connect to a very old Windows machine (for example, Windows Server 2003 or older).

### CAUTION:
**Security Hazard!**
Selecting this option can significantly reduce the strength of the encryption used!

### selection
- **Type:** legacy | tls
- **Description:** Configures the encryption used in the sessions.
  - **legacy:** Disables TLS encryption for RDP connections completely, and uses only the Legacy RDP Security Layer (also known as: Standard RDP Security). You might want to do this if you are experiencing compatibility issues. For example, when you attempt to connect to a very old Windows machine (for example, Windows Server 2003 or older).
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUTION:</strong></td>
<td></td>
<td>Security Hazard! Selecting this option can significantly reduce the strength of the encryption used!</td>
</tr>
<tr>
<td>&quot;transport_security&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;selection&quot;: &quot;legacy&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• tls: Enables TLS-encryption.</td>
<td></td>
<td>Note that you must also set the certificate and legacy_fallback options.</td>
</tr>
<tr>
<td>&quot;transport_security&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;certificate&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;selection&quot;: &quot;self_signed&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;legacy_fallback&quot;: false,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;selection&quot;: &quot;tls&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

web_gateway_authentication

Top level item

When gateway authentication is required for a connection, the user must authenticate on SPS as well. This additional authentication can be performed out-of-band on the SPS web interface for every protocol.

enabled

boolean

Set to true to enable additional gateway authentication on the SPS web interface.

groups

list, string

By default, any user can perform gateway authentication for the connections. You can restrict authentication to members of specific usergroups. Define the usergroups at the /api/configuration/aaa/local_database/groups/ endpoint, and list the name of each group here.

require_same_ip

boolean

Set to true to only accept web gateway authentication from the same host that initiated the connection.
## Elements of access_control

<table>
<thead>
<tr>
<th>Elements of access_control</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizer</td>
<td>string</td>
<td>The usergroup (local or LDAP) who can authorize or audit the connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Local usergroups can be added or modified at the /api/configuration/aaa/local_database/groups/ endpoint.</td>
</tr>
<tr>
<td>permission</td>
<td>string</td>
<td>Defines the permissions of the authorizer usergroup. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• audit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• authorize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the authorize permission can authorize connection requests.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• audit_and_authorize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the audit_and_authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.</td>
</tr>
<tr>
<td>require_different_ip</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different IP addresses.</td>
</tr>
<tr>
<td>require_different_username</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different usernames.</td>
</tr>
<tr>
<td>subject</td>
<td>Top level</td>
<td>Defines the subjects of the access control policy.</td>
</tr>
<tr>
<td>group</td>
<td>string</td>
<td>The usergroup (local or LDAP) that is subject to the access control policy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Local usergroups can be added or modified at the /api/configuration/aaa/local_database/groups/ endpoint.</td>
</tr>
</tbody>
</table>

SPS 6.13.0 REST API Reference Guide
### Elements of access_control

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• everybody</td>
</tr>
<tr>
<td></td>
<td>Every user is subject to the access control policy.</td>
</tr>
<tr>
<td></td>
<td>• only</td>
</tr>
<tr>
<td></td>
<td>Requires the group element.</td>
</tr>
<tr>
<td></td>
<td>Members of the usergroup specified in the group element are subject to the access control policy.</td>
</tr>
</tbody>
</table>

### Elements of remote_desktop_gateway

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>Set to true and configure the other options as needed for your environment to use SPS as a Remote Desktop Gateway. For details and prerequisites, see &quot;Using One Identity Safeguard for Privileged Sessions (SPS) as a Remote Desktop Gateway&quot; in the Administration Guide.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>host_certification_method</th>
<th>JSON object</th>
<th>To act as a Remote Desktop Gateway, SPS needs to display a certificate to the clients.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• To display always the same certificate, set &quot;selection&quot;: &quot;single&quot;, and reference a X.509 certificate and the matching private key. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;host_certification_method&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;signing&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;value&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;signing_ca&quot;: &quot;53449998258a4ceba80fde&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;common_name&quot;: &quot;examplecn&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Elements of remote_desktop_gateway</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For details on uploading certificates, see Certificates stored on SPS on page 287.</td>
</tr>
<tr>
<td>To automatically create new certificates on SPS for every client, set &quot;selection&quot;: &quot;signing&quot;, and reference the Certificate Authority (CA) to sign the generated certificates. For example:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;host_certification_method&quot;: {</td>
<td></td>
<td>For details on creating a signing CA, see Signing CA policies on page 406.</td>
</tr>
<tr>
<td>&quot;selection&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;single&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;value&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;1904188625a843f11d30a5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
<td></td>
</tr>
<tr>
<td>selection</td>
<td>single</td>
<td>Determines if SPS displays the same certificate to every client (single), or generates a separate certificate (signing) for every client.</td>
</tr>
<tr>
<td>value</td>
<td>JSON object or string</td>
<td>Contains the options and parameters related to the option set in selection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If selection is set to signing, this is a JSON object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If selection is set to single, this is a string containing the reference ID of the certificate that SPS displays to the clients.</td>
</tr>
<tr>
<td>common_name</td>
<td>string</td>
<td>Available only if selection is set to signing. You can specify the Common Name of the generated certificates in this parameter. For</td>
</tr>
<tr>
<td>Elements of remote_desktop_gateway</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>example</td>
<td>example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;common_name&quot;: &quot;examplecn&quot;</td>
</tr>
<tr>
<td></td>
<td>signing_ca</td>
<td>Available only if selection is set to signing. Contains the reference key of the signing CA used to sign the certificates that SPS shows to the clients. For example:</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td>&quot;signing_ca&quot;: &quot;1904188625a843f11d30a5&quot;</td>
</tr>
<tr>
<td></td>
<td>local_</td>
<td>Determines how SPS authenticates the clients: either using Active Directory (SPS must be member of a domain), or using a Local User Database.</td>
</tr>
<tr>
<td>authentication</td>
<td>JSON object</td>
<td>selection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Determines how SPS authenticates the clients:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- using Active Directory (SPS must be member of a domain)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;local_authentication&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;local_user_database&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;value&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;domain&quot;: &quot;example&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;local_user_database&quot;: &quot;15646962145a843f758501-d&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
Elements of remote_desktop_gateway

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>using a Local User Database</td>
<td>Set to null if selection is set to active_directory.</td>
</tr>
<tr>
<td></td>
<td>If selection is set to local_user_database, value contains a JSON object</td>
</tr>
<tr>
<td></td>
<td>with the domain and local_user_database keys.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>value</th>
<th>JSON object</th>
<th>Set to null if selection is set to active_directory.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>If selection is set to local_user_database, value contains a JSON object with the domain and local_user_database keys.</td>
</tr>
<tr>
<td>domain</td>
<td>string</td>
<td>Available only if selection is set to local_user_database.</td>
</tr>
<tr>
<td>local_user_database</td>
<td>string</td>
<td>Available only if selection is set to local_user_database. Contains the reference ID of a Local User Database that SPS will use to authenticate the clients.</td>
</tr>
</tbody>
</table>

Elements of server_address

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>custom_dns</td>
<td>Configures a DNS server that is used to reverse-resolve the hostname if the Channel Policy contains the address of the target as a hostname instead of an IP address. By default, this is disabled and SPS uses the DNS server set in the /api/configuration/network/dns endpoint.</td>
</tr>
</tbody>
</table>

- To use the default DNS, disable this option:

  ```json
  "server_address": { 
    "custom_dns": { 
      "enabled": false 
    },
    ...
  },
  ...
  
- To use a custom DNS, enable this option:
<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>option and set the IP address of the domain name server to use:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;server_address&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;custom_dns&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;enabled&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;server&quot;: &quot;192.168.1.1&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
</tbody>
</table>

**selection**  
string  
Configures the address where the clients connect to. Possible values are:

- original  
  Connect to the same address specified by the client.

- nat  
  Perform a network address translation on the target address.  
  Must be used with the network element.

- fix  
  Must be used with the address and port elements.

- inband  
  Extract the address of the server from the username.  
  Must be used with the domains element.

Optional elements: exception_domains, dns_server, and dns_suffixes.

| network | string | Must be used if selection is set to nat.  
The target address in IP/prefix format.  
Example: "10.20.30.40/24". |
|---------|--------|-------------|

| address | string | Must be used if selection is set to fix.  
The IP address of the target server. |
<table>
<thead>
<tr>
<th><strong>Elements of</strong> server_address</th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>int</td>
<td>Must be used if selection is set to fix. The port of the target server.</td>
</tr>
<tr>
<td>domains</td>
<td>Top level list</td>
<td>Must be used if selection is set to inband.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level item</td>
<td>Lists the address ranges that are included in the connection policy.</td>
</tr>
</tbody>
</table>
| selection                       | string   | Specifies if the target address range is provided as a domain or as an IP range. Possible values are:  
  * address  
    The value of the target address is an IP range.  
  * domain  
    The value of the target address is a domain. |
| value                           | string   | The address range of the target server(s). Use the selection element to specify if the address is an IP range, or a domain. |
| port                            | int      | The port of the target server(s). |
| exception_domains               | Top level list | Can only be used if selection is set to inband. Lists the address ranges that are excluded from the connection policy. |
| domain                          | Top level item | Contains the excluded address range. |
| selection                       | string   | Specifies if the excluded address(es) are provided as a domain or as an IP range. Possible values are:  
  * address  
    The value of the excluded address is an IP range.  
  * domain |
### Elements of server_address

<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>string</td>
<td>The excluded address(es). Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The excluded port.</td>
</tr>
<tr>
<td>dns_server</td>
<td>string</td>
<td>Can only be used if selection is set to inband. IP address or the hostname of the domain name server used to resolve the address of the target server.</td>
</tr>
<tr>
<td>dns_suffixes</td>
<td>list, string</td>
<td>Can only be used if selection is set to inband. If the clients do not include the domain name when addressing the server (for example they use username@server instead of <a href="mailto:username@server.example.com">username@server.example.com</a>), SPS can automatically add domain information (for example example.com). You can add multiple domain names. SPS attempts to resolve the target address by appending the domain names in the provided order, and uses the first successfully resolved address to establish the connection.</td>
</tr>
</tbody>
</table>

### Examples

For practical purposes, the following examples show only the relevant parts of a connection policy JSON object. To modify or add a connection policy, always submit the full JSON object.

**Access control list:** configuring the "security" usergroup to only audit connections made by the "root_only" usergroup.

```json
"access_control": [
  {
    "authorizer": "security",
    "permission": "audit",
    "require_different_ip": true,
    "require_different_username": true,
  }
]
```
Access control list: configuring the "security" usergroup to only audit connections made by the "root_only" usergroup.

```
"access_control": [
  {
    "authorizer": "security",
    "permission": "audit",
    "require_different_ip": true,
    "require_different_username": true,
    "subject": {
      "group": "root_only",
      "selection": "only"
    }
  }
]
```

Target server: use the address specified by the client.

```
"server_address": {
  "selection": "original"
}
```

Target server: use a fix address.

```
"server_address": {
  "address": "<fix-IP>",
  "port": 22,
  "selection": "fix"
}
```

Target server: configure inband destination selection, where the client can specify the target address in the username. The target can be either an IP range, or a domain.

```
"server_address": {
  "dns_server": "<ip-of-dns-server>",
  "dns_suffixes": null,
  "domains": [
    {
      "domain": {
        "selection": "address",
        "value": "<IP-range>"
      },
      "port": 22
    }
  ]
}
```
Source address: use the same fix IP when connecting to the remote server.

```
"source_address": {  
  "address": "<ip-address>",  
  "selection": "fix"  
}
```

Web gateway authentication: require the admin usergroup to perform an additional gateway authentication on the SPS web interface. They must authenticate from the same host which initiated the connection.

```
"web_gateway_authentication": {  
  "enabled": true,  
  "groups": [  
    "admin"  
  ],  
  "require_same_ip": true  
}
```

Policies: configure only the required policies.

```
"policies": {  
  "aa_plugin": null,  
  "analytics_policy": null,  
  "archive_cleanup_policy": null,  
  "audit_policy": "<key-of-audit-policy>",  
  "backup_policy": null,  
  "channel_policy": "<key-of-channel-policy>",  
  "credential_store": null,  
  "ldap_server": null,  
  "settings": "<key-of-settings-policy>",  
  "usermapping_policy": null  
}
```
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Add a connection policy

To add an RDP connection policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new RDP connection policy.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/rdp/connections/ endpoint. You can find a detailed description of the available parameters listed in Element.

   If the POST request is successful, the response includes the key of the new connection policy. For example:

   ```
   {
     "key": "a99be49b-b0a2-4cf9-b70d-fea1f9ea188f",
     "meta": {
       "href": "/api/configuration/rdp/connections/a99be49b-b0a2-4cf9-b70d-fea1f9ea188f",
       "parent": "/api/configuration/rdp/connections",
       "transaction": "/api/transaction"
     }
   }
   ```
3. **Commit your changes.**
   For more information, see *Commit a transaction* on page 31.

**Modify an RDP connection policy**

To modify an RDP connection policy, you have to:

1. **Open a transaction.**
   For more information, see *Open a transaction* on page 29.

2. **Modify the JSON object of the connection policy.**
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/rdp/connections/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**
   For more information, see *Commit a transaction* on page 31.

---

**RDP channels**

The available RDP channel types and their functionalities are described below. For details on configuring channel policies, see *Channel policy*.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
</table>
| #drawing | Yes             | **Drawing**: Enables access to the server's graphical desktop (screen). This channel must be enabled for RDP to work. Channel-specific actions:
   - content_policy reference: The ID of the Content policy to apply to the connection. For example:

```
"actions": {
  "audit": true,
  "content_policy": {
    "key": "433849548566ab327522e6"
  },
  "four_eyes": false,
```

SPS 6.13.0 REST API Reference Guide

565

RDP connections
<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cliprdr</td>
<td>None</td>
<td><strong>Clipboard</strong>: Enable access to the server's clipboard: the clipboard of the remote desktop can be pasted into local applications (and vice-versa). Note that SPS can audit the clipboard channel, but cannot search or display its contents.</td>
</tr>
</tbody>
</table>
| rdpdr   | Yes             | **Redirects**: Enables access to every device redirections available in RDP, like file-sharing, printer sharing, device (for example CD-ROM) sharing, and so on. To enable only a specific type of redirection, use the specific channels instead (for example, rdpdr-serial for serial device redirection). Channel-specific actions:  
  - `log_transfer_to_db (true|false)`: Make the list of file operations available in the Search > File operations column of the SPS web interface  
  - `log_transfer_to_syslog (true|false)`: Send the file operations into the system log  
  Channel-specific access control rules:  
  - `devices (list)`: To permit only specific redirections, list the unique name of the redirection in this field. Leave it empty to permit access to every redirection available. |
| rdpsnd  | None            | **Sound**: Enable access to the sound device of the server. |
| customs | Yes             | **Custom**: Applications can open custom channels to the clients connecting remotely to the server. Enabling the Custom channel allows the clients to access all of these custom channels. To permit only specific channels, list the unique names of the channels into the customs field.  
  For example, to monitor RemoteApp connections, you need to configure custom channels. For more information, see "Configuring RemoteApps" in the Administration Guide.  
  Channel-specific access control rules:  
  - `customs (list)`: To permit only specific custom channels, list the unique name of the channels in this field. Leave it empty to permit access to every custom channel available. |
<p>| seamrdp | None            | <strong>Seamless</strong>: Enable seamless channels that run a single application on the RDP server, instead of accessing the entire desktop. |</p>
<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>drdynvc</td>
<td>Yes</td>
<td><strong>Dynamic virtual channel</strong>: Enable the server to open channels back to the client dynamically. Enabling this channel allows access to all of such dynamic channels. To restrict which dynamic channels are permitted, list the unique names of the channels into the <code>drdynvcs</code> field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Channel-specific access control rules:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>drdynvcs</code> (list): To restrict which dynamic channels are permitted, list the unique names of the channels in this field. Leave it empty to permit access to every dynamic channel available.</td>
</tr>
<tr>
<td>rdpdr-serial</td>
<td>Yes</td>
<td><strong>Serial redirect</strong>: Enables access to serial-port redirections. To restrict access to specific redirections, list the unique names of the channels in the <code>devices</code> field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Channel-specific access control rules:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>devices</code> (list): To permit only specific redirections, list the unique name of the redirection in this field. Leave it empty to permit access to every redirection available.</td>
</tr>
<tr>
<td>rdpdr-parallel</td>
<td>Yes</td>
<td><strong>Parallel redirect</strong>: Enables access to parallel-port redirections. To restrict access to specific redirections, list the unique names of the channels in the <code>devices</code> field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Channel-specific access control rules:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>devices</code> (list): To permit only specific redirections, list the unique name of the redirection in this field. Leave it empty to permit access to every redirection available.</td>
</tr>
<tr>
<td>rdpdr-printer</td>
<td>Yes</td>
<td><strong>Printer redirect</strong>: Enables access to printer-port redirections. To restrict access to specific redirections, list the unique names of the channels in the <code>devices</code> field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Channel-specific access control rules:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>devices</code> (list): To permit only specific redirections, list the unique name of the redirection in this field. Leave it empty to permit access to every redirection available.</td>
</tr>
<tr>
<td>rdpdr-disk</td>
<td>Yes</td>
<td><strong>Disk redirect</strong>: Enables access to shared disk drives. To restrict access to specific redirections, list the unique names of the channels in the <code>devices</code> field, for example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;devices&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;C:&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>]</td>
</tr>
<tr>
<td>Channel</td>
<td>Special options</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Channel-specific actions:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>log_transfer_to_db</code> (true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>log_transfer_to_syslog</code> (true</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Channel-specific access control rules:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>devices</code> (list): To permit only specific redirections, list the unique name of the redirection in this field. Leave it empty to permit access to every redirection available.</td>
</tr>
</tbody>
</table>

**SCard redirect:** Enables access to shared SCard devices. To restrict access to specific redirections, list the unique names of the channels in the `devices` field, for example:

**Channel-specific access control rules:**

- `devices` (list): To permit only specific redirections, list the unique name of the redirection in this field. Leave it empty to permit access to every redirection available.

## Configuring domain membership

You can use Credential Security Service Provider (CredSSP, also called Network Level Authentication or NLA) when One Identity Safeguard for Privileged Sessions (SPS) is member of the domain.

### Prerequisites

- The target servers and SPS must be in the same domain, or you must establish trust between the domains that contain the target servers and SPS. For details on the type of trust required, see "Using One Identity Safeguard for Privileged Sessions (SPS) across multiple domains" in the Administration Guide.

The SPS configuration API allows you to view, disable, or modify the domain membership configuration. To join the configured domain, you have to use the web interface of SPS.

### URL

GET https://<IP-address-of-SPS>/api/rdp/domain_membership
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Sample request

The following command lists the configuration options for domain membership.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/rdp/domain_membership
```

Response

The following is a sample response received when querying the domain membership configuration.

For details of the meta object, see Message format on page 10.

```json
{
   "body": {
      "domain": "testdomain",
      "enabled": true,
      "realm": "testdomain.api.test"
   },
   "key": "domain_membership",
   "meta": {
      "first": "/api/configuration/rdp/channel_policies",
      "href": "/api/configuration/rdp/domain_membership",
      "last": "/api/configuration/rdp/settings_policies",
      "next": "/api/configuration/rdp/options",
      "parent": "/api/configuration/rdp",
      "previous": "/api/configuration/rdp/channel_policies",
      "transaction": "/api/transaction"
   }
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the domain membership configuration.</td>
</tr>
<tr>
<td>domain</td>
<td>string</td>
<td>The name of the domain. Must be used if enabled is set to true.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to configure domain membership.</td>
</tr>
<tr>
<td>realm</td>
<td>string</td>
<td>Name of the realm. Must be used if enabled is set to true.</td>
</tr>
</tbody>
</table>

**Examples:**

Configure domain membership for the "test" domain on the "config.api" realm:

```json
{
    "domain": "test",
    "enabled": true,
    "realm": "test.config.api"
}
```

Disable domain membership.

```json
{
    "enabled": false
}
```

**Modify domain membership settings**

To modify domain membership settings, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the domain membership configuration.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/rdp/domain_membership/` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Global RDP options

List of options that affect all RDP connections.

#### URL

```
GET https://<IP-address-of-SPS>/api/configuration/rdp/options
```

#### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions.
<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
</table>

that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists global RDP options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/rdp/options
```

Response

The following is a sample response received when listing global RDP options.

For details of the meta object, see Message format on page 10.

```json
{
  "body": {
    "audit": {
      "cleanup": {
        "enabled": false
      },
      "timestamping": {
        "selection": "local",
        "signing_interval": 30
      }
    },
    "service": {
      "enabled": true,
      "log_level": 4
    }
  },
  "key": "options",
  "meta": {
    "first": "/api/configuration/rdp/channel_policies",
    "href": "/api/configuration/rdp/options",
    "last": "/api/configuration/rdp/settings_policies",
    "next": "/api/configuration/rdp/settings_policies",
    "parent": "/api/configuration/rdp",
    "previous": "/api/configuration/rdp/domain_membership",
    "transaction": "/api/transaction"
  }
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Top level</td>
<td>Contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>Contains the elements of the global RDP options.</td>
</tr>
<tr>
<td>audit</td>
<td>Top level</td>
<td>Contains settings for timestamping and cleanup.</td>
</tr>
<tr>
<td>service</td>
<td>Top level</td>
<td>Global setting to enable RDP connections, and specify the logging detail.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable RDP connections.</td>
</tr>
<tr>
<td>log_level</td>
<td>int</td>
<td>Defines the logging detail of RDP connections.</td>
</tr>
</tbody>
</table>

### Elements of audit

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanup</td>
<td>Top level</td>
<td>Global retention settings for RDP connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.</td>
</tr>
<tr>
<td>channel_database_cleanup_days</td>
<td>int</td>
<td>Global retention time for the metadata of RDP connections, in days. Must exceed the retention time of the archiving policy (or policies) used for RDP connections, and the connection-specific database cleanup times (if configured).</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>To enable the global cleanup of RDP connection metadata, set this element to true.</td>
</tr>
<tr>
<td>timestamping</td>
<td>Top level</td>
<td>Global timestamping settings for RDP connections.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures local or remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Set local to use SPS for timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Set remote to configure a remote timestamping server.</td>
</tr>
<tr>
<td>server_url</td>
<td>string</td>
<td>Required for remote timestamping.</td>
</tr>
</tbody>
</table>
### Elements of audit

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.</td>
</tr>
<tr>
<td>oid</td>
<td>Top level</td>
<td>The Object Identifier of the policy used for timestamping.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required for remote timestamping. Set to true to configure the Object Identifier of the timestamping policy on the timestamping remote server.</td>
</tr>
<tr>
<td>policy_oid</td>
<td>string</td>
<td>Required if the oid is enabled. The Object Identifier of the timestamping policy on the remote timestamping server.</td>
</tr>
<tr>
<td>signing_interval</td>
<td>int</td>
<td>Time interval for timestamping open connections, in seconds.</td>
</tr>
</tbody>
</table>

**Examples:**

Set SPS as the timestamping server:

```json
{
    "audit": {
        "cleanup": {
            "enabled": false
        },
        "timestamping": {
            "selection": "local",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Enable cleanup, and set it to occur every 10 days:

```json
{
    "audit": {
        "cleanup": {
            "enabled": false
        },
        "timestamping": {
            "signing_interval": 10
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```
Change timestamping to a remote server, without specifying a timestamping policy:

```json
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": false
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, and specify the 1.2.3 timestamping policy:

```json
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": true
            },
            "selection": "remote",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```
Modify global RDP settings

To modify global RDP settings, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the global RDP settings endpoint.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/rdp/options endpoint. You can find a detailed description of the available parameters listed in Element. The elements of the audit item are described in Elements of audit.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

## RDP settings policies

RDP settings policies define protocol-level settings (timeout, display, protocol version, and authentication). You can create multiple policies, and choose the appropriate one for each RDP connection.

### URL

GET https://<IP-address-of-SPS>/api/configuration/rdp/settings_policies

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

### Sample request

The following command lists RDP settings policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/rdp/settings_policies
```

The following command retrieves the properties of a specific policy.
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/rdp/settings_policies/<policy-id>

Response

The following is a sample response received when listing RDP settings policies. For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "-301",
      "meta": {
        "href": "/api/configuration/rdp/settings_policies/-301"
      }
    },
    {
      "key": "-303",
      "meta": {
        "href": "/api/configuration/rdp/settings_policies/-303"
      }
    },
    {
      "key": "13298899495727c51f725cf",
      "meta": {
        "href": "/api/configuration/rdp/settings_policies/13298899495727c51f725cf"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/rdp/channel_policies",
    "href": "/api/configuration/rdp/settings_policies",
    "last": "/api/configuration/rdp/settings_policies",
    "next": null,
    "parent": "/api/configuration/rdp",
    "previous": "/api/configuration/rdp/options",
    "transaction": "/api/transaction"
  }
}
```

When retrieving the endpoint of a specific policy, the response is the following.

```
{
  "body": {
    "autologon_domain_suffix": "-AUTO",
    "name": "API_test",
    "timeout": 600,
  }
}
```
"inactivity_timeout": {
  "enabled": true,
  "value": 13000
},
"permit_unreliable_usernames": true,
"preconnect_channel_check": true,
"protocol_features": {
  "nla": {
    "enabled": true,
    "require_domain_membership": true
  },
  "rdp4_auth_enabled": true,
  "rdp4_enabled": true,
  "rdp5_enabled": true
},
"screen": {
  "maximum bpp": 32,
  "maximum_height": 2000,
  "maximum_width": 2000
},
"timeout": 600,
"userauth_banner": "Click 'OK' to log in."
},
"key": "13298899495727c51f725cf",
"meta": {
  "first": "/api/configuration/rdp/settings_policies/-301",
  "href": "/api/configuration/rdp/settings_policies/13298899495727c51f725cf",
  "last": "/api/configuration/rdp/settings_policies/13298899495727c51f725cf",
  "next": null,
  "parent": "/api/configuration/rdp/settings_policies",
  "previous": "/api/configuration/rdp/settings_policies/-303",
  "transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the RDP settings policy.</td>
</tr>
<tr>
<td>autologon_domain_suffix</td>
<td>string</td>
<td>Enter the suffix that the client will append to the domain when using autologon in conjunction with Network Level Authentication (CredSSP).</td>
</tr>
</tbody>
</table>
### Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the RDP settings policy. Cannot contain whitespace.</td>
</tr>
<tr>
<td>permit_unreliable_usernames</td>
<td>boolean</td>
<td>Set to true to automatically terminate RDP connections if SPS cannot reliably extract the username.</td>
</tr>
<tr>
<td>preconnect_channel_check</td>
<td>boolean</td>
<td>Before establishing the server-side connection, SPS can evaluate the connection and channel policies to determine if the connection might be permitted at all. The server-side connection is established only if the evaluated policies permit the client to access the server. To enable this function, set the parameter to true.</td>
</tr>
<tr>
<td>protocol_features</td>
<td>Top level item</td>
<td>Settings for RDP protocol versions, and Network Layer Authentication.</td>
</tr>
<tr>
<td>screen</td>
<td>Top level item</td>
<td>Display size and depth settings.</td>
</tr>
<tr>
<td>timeout</td>
<td>int</td>
<td>Connection timeout, in seconds.</td>
</tr>
</tbody>
</table>
| inactivity_timeout                          | Top level element | • true: If no user activity is detected, it terminates the session after the configured time has passed since the last user activity.  
• false: No user inactivity timeout. |
| value                                       | int             | Only if enabled is true. The value of user activity timeout. Must be greater than or equal to the value of timeout. |
| userauth_banner                             | string          | You can display a banner message to the clients before authentication.      |

### Elements of protocol

<table>
<thead>
<tr>
<th>Component</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nla</td>
<td>Top level item</td>
<td>Settings for Network Level Authentication.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable Network Level Authentication.</td>
</tr>
</tbody>
</table>
## Elements of protocol

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>require_domain_membership boolean</td>
<td>If set to true, the require_domain_membership element is required in the JSON. Must be in the JSON if NLA is enabled.</td>
</tr>
<tr>
<td>rdp4_auth_enabled boolean</td>
<td>Set to true to enable RDP4 authentication within the RDP5 protocol. This might be needed for compatibility reasons with certain client applications.</td>
</tr>
<tr>
<td>rdp4_enabled boolean</td>
<td>Set to true to enable the version 4 of the Remote Desktop Protocol.</td>
</tr>
<tr>
<td>rdp5_enabled boolean</td>
<td>Set to true to enable the version 5 of the Remote Desktop Protocol. To also configure SSL-encryption for RDP5, enable the nla element, or configure a Signing CA in your connection policies.</td>
</tr>
</tbody>
</table>

## Elements of screen

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximum_bpp int</td>
<td>The maximum allowed color depth of the remote desktop, in bits. The following values are valid: 8, 15, 16, 24.</td>
</tr>
<tr>
<td>maximum_height int</td>
<td>The maximum allowed height of the remote desktop, in pixels.</td>
</tr>
<tr>
<td>maximum_width int</td>
<td>The maximum allowed width of the remote desktop, in pixels.</td>
</tr>
</tbody>
</table>

## Examples:

Turn off NLA.

```json
{
    "autologon_domain_suffix": "-AUTO",
    "name": "API_test",
    "permit_unreliable_usernames": true,
    "preconnect_channel_check": true,
    "protocol_features": {
        "nla": {
            "enabled": false
        },
        "rdp4_auth_enabled": true,
        "rdp4_enabled": true,
        "rdp5_enabled": true
    }
}
```
Configure NLA.

```json
{
  "autologon_domain_suffix": "-AUTO",
  "name": "API_test",
  "permit_unreliable_usernames": true,
  "preconnect_channel_check": true,
  "protocol_features": {
    "nla": {
      "enabled": true,
      "require_domain_membership": false
    },
    "rdp4_auth_enabled": true,
    "rdp4_enabled": true,
    "rdp5_enabled": true
  },
  "screen": {
    "maximum_bpp": 24,
    "maximum_height": 2000,
    "maximum_width": 2000
  },
  "timeout": 600
}
```

Add RDP settings policies

To add a settings policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new policy.**
   
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/rdp/settings_policies/ endpoint. You can find a detailed description of the available parameters listed in the table of RDP settings policy parameters.
If the POST request is successful, the response includes the key of the new policy. For example:

```
{
    "key": "9c3a0419-53e6-43a4-902c-2b3b0ce7a7a7",
    "meta": {
        "href": "/api/configuration/rdp/settings_policies/9c3a0419-53e6-43a4-902c-2b3b0ce7a7a7",
        "parent": "/api/configuration/rdp/settings_policies",
        "transaction": "/api/transaction"
    }
}
```

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Modify RDP settings policies**

To modify a settings policy, you have to:

1. **Open a transaction.**

   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the policy.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/rdp/settings_policies/<key-of-the-object>` endpoint. You can find a detailed description of the available parameters listed in the table of RDP settings policy parameters.

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes](#) on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>You have set require_domain_membership to true, but SPS is not the member of a domain.</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;RDP Settings&quot;</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
SSH connections

List of endpoints for configuring the policies, options and connection rules of SSH connections.

URL

GET https://<IP-address-of-SPS>/api/configuration/ssh

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available settings for configuring for SSH connections.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ssh

SPS 6.13.0 REST API Reference Guide
SSH connections

ONE IDENTITY by Quest
Response

The following is a sample response received when listing the configuration settings.
For details of the meta object, see Message format on page 10.

```json
{
    "items": [ 
        {
            "key": "authentication_policies",
            "meta": {
                "href": "/api/configuration/ssh/authentication_policies"
            }
        },
        {
            "key": "channel_policies",
            "meta": {
                "href": "/api/configuration/ssh/channel_policies"
            }
        },
        {
            "key": "connections",
            "meta": {
                "href": "/api/configuration/ssh/connections"
            }
        },
        {
            "key": "options",
            "meta": {
                "href": "/api/configuration/ssh/options"
            }
        },
        {
            "key": "settings_policies",
            "meta": {
                "href": "/api/configuration/ssh/settings_policies"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/aaa",
        "href": "/api/configuration/ssh",
        "last": "/api/configuration/x509",
        "next": "/api/configuration/telnet",
        "parent": "/api/configuration",
        "previous": "/api/configuration/reporting",
        "transaction": "/api/transaction"
    }
}
```
### Item Description

**authentication_policies**  List of the default and custom authentication policies.

**channel_policies**  List of the default and custom channel policies.

**connections**  List of connection policies.

**options**  List of global SSH options that affect all connections.

**settings_policies**  List of protocol-level settings (algorithms, greetings and banners, timeout). You can create multiple variations, and choose the appropriate one for each connection policy.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### SSH connection policies

Connection policies determine if a server can be accessed from a particular client. Connection policies reference other resources (policies, usergroups, keys) that must be configured and available before creating a connection policy.

### URL

GET https://<IP-address-of-SPS>/api/configuration/ssh/connections/
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e6576347309e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists SSH connection policies.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ssh/connections/
```

The following command retrieves the properties of a specific policy.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ssh/connections/<connection-key>
```

Response

The following is a sample response received when listing SSH connection policies.

For details of the meta object, see Message format on page 10.

```json
{
  "items": [
    {
      "key": "8348340645707e2575e3c6",
      "meta": {
        "href": "/api/configuration/ssh/connections/8348340645707e2575e3c6"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/ssh/authentication_policies",
    "href": "/api/configuration/ssh/connections",
```
When retrieving the endpoint of a specific SSH connection policy, the response is the following.

```json
{
    "body": {
        "access_control": [
            {
                "authorizer": "reporting",
                "permission": "audit_and_authorize",
                "require_different_ip": true,
                "require_different_username": true,
                "subject": {
                    "selection": "everybody"
                }
            }
        ],
        "active": true,
        "channel_database_cleanup": {
            "days": 550,
            "enabled": true
        },
        "client_side_hostkey": {
            "plain_hostkey": {
                "dsa_key": null,
                "enabled": true,
                "rsa_key": {
                    "key": "e5a58682-6189-4477-9415-67c1c9b20b0d",
                    "meta": {
                        "href": "/api/configuration/private_keys/e5a58682-6189-4477-9415-67c1c9b20b0d"
                    }
                }
            },
            "x509_hostkey": {
                "enabled": false
            }
        },
        "indexing": {
            "enabled": true,
            "policy": {
```
"key": "-50000",
"meta": {
   "href": "/api/configuration/policies/indexing/-50000"
},
"priority": 2,
"log_audit_trail_downloads": true,
"name": "API_test_SSH",
"network": {
   "clients": ["0.0.0.0/24"],
   "ports": [22],
   "targets": ["192.168.56.102/24"]
},
"policies": {
   "aa_plugin": null,
   "analytics_policy": null,
   "archive_cleanup_policy": {
      "key": "1854671967571b9063c4c82",
      "meta": {
         "href": "/api/configuration/policies/archive_cleanup_policies/1854671967571b9063c4c82"
      }
   },
   "audit_policy": {
      "key": "78101850949e47437dd91d",
      "meta": {
         "href": "/api/configuration/policies/audit_policies/78101850949e47437dd91d"
      }
   },
   "authentication_policy": {
      "key": "1895203635707e3340262f",
      "meta": {
         "href": "/api/configuration/ssh/authentication_policies/1895203635707e3340262f"
      }
   },
   "backup_policy": {
      "key": "512524636571b903540804",
      "meta": {
         "href": "/api/configuration/policies/backup_
"policies": { "key": "512524636571b903540804", "meta": { }, "channel_policy": { "key": "-10000", "meta": { "href": "/api/configuration/ssh/channel_policies/-10000" } }, "credential_store": { "key": "505008562571b936560254", "meta": { "href": "/api/configuration/policies/credentialstores/505008562571b936560254" } }, "ldap_server": { "key": "250588254571b931066482", "meta": { "href": "/api/configuration/policies/ldap_servers/250588254571b931066482" } }, "settings": { "key": "-300", "meta": { "href": "/api/configuration/ssh/settings_policies/-300" } }, "usermapping_policy": { "key": "9328731525704545f5e3de", "meta": { "href": "/api/configuration/policies/usermapping_policies/9328731525704545f5e3de" } }, "rate_limit": { "enabled": true, "value": 200 }, "server_address": { "selection": "original" }, "server_side_hostkey": { "plain_hostkey": { "enabled": true, "hostkey_check": "accept-first-time" } } }
```
{
   "x509_hostkey": {
      "enabled": false
   },
   "source_address": {
      "custom_dns": {
         "enabled": false
      },
      "selection": "box_address"
   },
   "web_gateway_authentication": {
      "enabled": true,
      "groups": [
         "reporting"
      ],
      "require_same_ip": true
   },
   "key": "8348340645707e2575e3c6",
   "meta": {
      "first": "/api/configuration/ssh/connections/8348340645707e2575e3c6",
      "href": "/api/configuration/ssh/connections/8348340645707e2575e3c6",
      "last": "/api/configuration/ssh/connections/8348340645707e2575e3c6",
      "next": null,
      "parent": "/api/configuration/ssh/connections",
      "previous": null,
      "transaction": "/api/transaction"
   }
}
```

<table>
<thead>
<tr>
<th><strong>Element</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the connection policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the connection policy.</td>
</tr>
<tr>
<td>access_control</td>
<td>Top level list</td>
<td>Collection of access policies. Access policies define who can authorize and audit a connection.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Set to false to suspend the connection policy. Connection settings are preserved.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>channel_database_cleanup</td>
<td>Top level item</td>
<td>Configures cleanup of the connection metadata on the connection policy's level.</td>
</tr>
<tr>
<td>days</td>
<td>int</td>
<td>Retention time, in days. Must not exceed the retention time of the archive_cleanup_policy, and the retention time configured in the global settings of the protocol. The global settings of the SSH protocol are available at the api/configuration/ssh/options endpoint.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable periodical cleanup of the connection metadata.</td>
</tr>
<tr>
<td>indexing</td>
<td>Top level item</td>
<td>Configures indexing for the connection policy.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable indexing the connections.</td>
</tr>
<tr>
<td>policy</td>
<td>string</td>
<td>References the identifier of the indexing policy. You can configure indexing policies at the /api/configuration/policies/indexing/endpoint. To modify or add an indexing policy, use the value of the returned key as the value of the policy element, and remove any child elements (including the key).</td>
</tr>
</tbody>
</table>
| priority                | int       | Specifies the indexing priority for the connection. Possible values are: 5  
  Very low priority. 4  
  Low priority. 3  
  Normal (default) priority. 2  
  High priority. |
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Very high priority.</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>Near real-time priority.</td>
</tr>
<tr>
<td>log_audit_trail_downloads</td>
<td>boolean</td>
<td>Set to true to log audit trail downloads.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the connection policy.</td>
</tr>
<tr>
<td>network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clients</td>
<td>list, string</td>
<td>List of client (&quot;from&quot;) IP addresses.</td>
</tr>
<tr>
<td>ports</td>
<td>list, integers</td>
<td>List of target ports.</td>
</tr>
<tr>
<td>targets</td>
<td>list, string</td>
<td>List of target IP addresses.</td>
</tr>
<tr>
<td>override_log_level</td>
<td>Top level item</td>
<td>Specifies the verbosity level of sessions handled by this connection policy. The log level of other connection policies is not affected. If disabled, the log level set at the /api/configuration/&lt;protocol&gt;/options endpoint is used.</td>
</tr>
</tbody>
</table>

- To use the default log level, disable this option:

```json
"override_log_level": {
    "enabled": false
},
```

- To use a custom log level for the connection policy, enable this option and set the log level to use:

```json
"override_log_level": {
    "enabled": true,
    "log_level": 5
},
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policies</td>
<td>Top level item</td>
<td>List of policies referenced by the connection policy.</td>
</tr>
<tr>
<td>aa_plugin</td>
<td>string</td>
<td>References the identifier of the AA plug-in. You can configure AA plug-ins at the /api/configuration/plugins/aa/ endpoint. To modify or add an AA plug-in, use the value of the returned key as the value of the aa_plugin element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>analytics_policy</td>
<td>string</td>
<td>References the identifier of the analytics policy. You can configure analytics policies at the /api/configuration/analytics/ endpoint. To add or modify an analytics policy, use the value of the returned key as the value of the analytics element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>archive_cleanup_policy</td>
<td>string</td>
<td>References the identifier of the archive/cleanup policy. You can configure archive and cleanup policies at the /api/configuration/policies/archive_cleanup_policies/ endpoint. To modify or add an archive/cleanup policy, use the value of the returned key as the value of the archive_cleanup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>audit_policy</td>
<td>string</td>
<td>Cannot be null. References the identifier of the audit policy. You can configure audit policies at the /api/configuration/policies/audit_policies/ endpoint. To modify or add an audit policy, use the value of the returned key as the value of the audit_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>authentication_policy</td>
<td>string</td>
<td>Cannot be null.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>References the identifier of the authentication policy. You can configure authentication policies at the /api/configuration/ssh/authentication_policies/ endpoint. To modify or add an authentication policy, use the value of the returned key as the value of the authentication_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>backup_policy</td>
<td>string</td>
<td>References the identifier of the backup policy. You can configure backup policies at the /api/configuration/policies/backup_policies/ endpoint. To modify or add a backup policy, use the value of the returned key as the value of the backup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>channel_policy</td>
<td>string</td>
<td>References the identifier of the channel policy. The value of this option cannot be null. To modify or add a channel policy, use the value of the returned key as the value of the channel_policy element, and remove any child elements (including the key). You can configure SSH channel policies at the /api/configuration/ssh/channel_policies/ endpoint.</td>
</tr>
<tr>
<td>credential_store</td>
<td>string</td>
<td>References the identifier of the credential store. You can configure credential stores at the /api/configuration/policies/credentialstores/ endpoint. To modify or add a credential store, use the value of the returned key as the value of the credential_store element, and remove any child elements (including the key).</td>
</tr>
</tbody>
</table>
| ldap_server            | string    | References the identifier of the LDAP

SPS 6.13.0 REST API Reference Guide
SSH connections
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server.</td>
<td></td>
<td>You can configure LDAP servers at the /api/configuration/policies/ldap_servers/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an LDAP server, use the value of the returned key as the value of the ldap_server element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>settings</td>
<td>string</td>
<td>References the identifier of the settings policy. The value of this option cannot be null.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a settings policy for this protocol, use the value of the returned key as the value of the settings element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can configure SSH settings policies at the /api/configuration/ssh/settings_policies/ endpoint.</td>
</tr>
<tr>
<td>usermapping_policy</td>
<td>string</td>
<td>References the identifier of a Usermapping Policy. You can configure Usermapping Policies at the /api/configuration/policies/usermapping_policies/ endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a Usermapping Policy, use the value of the returned key as the value of the usermapping_policies element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>rate_limit</td>
<td>Top level element</td>
<td>Connection rate limit.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to provide a connection rate limit.</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>The number of connections (per minute) that are allowed in the connection policy.</td>
</tr>
<tr>
<td>server_address</td>
<td>Top level item</td>
<td>Defines the address where the clients connect to.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>server_side_hostkey</td>
<td>Top level</td>
<td>Settings for verifying the server’s identity using plain host keys and X.509 host certificates. At least one of the options (plain_hostkey or X509_hostkey) must be enabled.</td>
</tr>
<tr>
<td>source_address</td>
<td>Top level</td>
<td>Allows you to configure Source Network Address Translation (SNAT) on the server side of SPS. SNAT determines the IP address SPS uses in the server-side connection. The target server will see the connection coming from this address.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures Source Network Address Translation. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- box_address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default. Uses the network address of the logical interface of SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- original</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses the IP address of the client, as seen by SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- fix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses a fixed address when connecting to the remote server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the address element.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if the value of the selection element is set to fix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The IP address to use as the source address in server-side connections.</td>
</tr>
<tr>
<td>web_gateway_authentication</td>
<td>Top level</td>
<td>When gateway authentication is required for a connection, the user must authenticate on SPS as well. This additional authentication can be performed out-of-band on the SPS web interface for every protocol.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable additional gateway authentication on the SPS web interface.</td>
</tr>
<tr>
<td>groups</td>
<td>list,</td>
<td>By default, any user can perform</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>require_same_ip</td>
<td>boolean</td>
<td>Set to true to only accept web gateway authentication from the same host that initiated the connection.</td>
</tr>
<tr>
<td>require_different_ip</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different IP addresses.</td>
</tr>
<tr>
<td>require_different_username</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different usernames.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of access_control</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizer</td>
<td>string</td>
<td>The usergroup (local or LDAP) who can authorize or audit the connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Local usergroups can be added or modified at the /api/configuration/aaa/local_database/groups/ endpoint.</td>
</tr>
<tr>
<td>permission</td>
<td>string</td>
<td>Defines the permissions of the authorizer usergroup. Possible values are:</td>
</tr>
</tbody>
</table>
|                               |         |   - audit  
|                               |         |     The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection.                                                                |
|                               |         |     The usergroup with the authorize permission can authorize connection requests.                                                                                                                           |
|                               |         |     The usergroup with the audit_and_authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.                                      |
### Elements of access_control

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td></td>
<td>Defines the subjects of the access control policy.</td>
</tr>
</tbody>
</table>
| subject     | Top level item | The usergroup (local or LDAP) that is subject to the access control policy.  
Local usergroups can be added or modified at the /api/configuration/aaa/local_database/groups/ endpoint. |
| group       | string     | Possible values:  
everybody  
Every user is subject to the access control policy.  
only  
Requires the group element.  
Members of the usergroup specified in the group element are subject to the access control policy. |
| selection   | string     | Possible values:  
everybody  
Every user is subject to the access control policy.  
only  
Requires the group element.  
Members of the usergroup specified in the group element are subject to the access control policy. |

### Elements of client_side_hostkey

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>plain_hostkey</td>
<td>Top level item</td>
<td>Configures the RSA key SPS shows to the clients.</td>
</tr>
</tbody>
</table>
| rsa_key       | string     | References the identifier of the RSA key. You can add RSA keys at the /api/configuration/private_keys/ endpoint.  
To modify or add an RSA key, use the value of the returned key as the value of the rsa_key element, and remove any child elements (including the key). |
| x509_hostkey  | Top level item | Configures the X.509 keys SPS shows to the clients.                                                                                     |
| enabled       | boolean    | Set to true to allow presenting X.509 host keys to clients.  
You must enable either plain_hostkey or x509_hostkey (or both).                                                                            |
| x509          | Top level  | Parameters for X.509 hostkeys.                                                                                                             |
### Elements of client_side_hostkey

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td><strong>selection</strong> string Possible values:</td>
</tr>
<tr>
<td></td>
<td>- <strong>fix</strong></td>
</tr>
<tr>
<td></td>
<td>Presents the same certificate for every connection.</td>
</tr>
<tr>
<td></td>
<td>Must be used with the <strong>x509_identity</strong> element.</td>
</tr>
<tr>
<td></td>
<td>- <strong>generate</strong></td>
</tr>
<tr>
<td></td>
<td>Generates a X.509 certificate for the connection policy.</td>
</tr>
<tr>
<td></td>
<td>Must be used with the <strong>signing_CA</strong> element.</td>
</tr>
<tr>
<td></td>
<td><strong>signing_ca</strong> string Must be used when generating the X.509 certificate.</td>
</tr>
<tr>
<td></td>
<td>References the signing Certificate Authority (CA). You can configure signing CAs at the <strong>/api/configuration/policies/signing_cas/</strong> endpoint.</td>
</tr>
<tr>
<td></td>
<td>To modify or add a signing CA, use the value of the returned key as the value of the <strong>rsa_key</strong> element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td></td>
<td><strong>x509_identity</strong> string Must be used when using the same X.509 host certificate across connection policies.</td>
</tr>
<tr>
<td></td>
<td>References the identifier of the X.509 certificate stored on SPS. You can configure certificates at the <strong>/api/configuration/x509/</strong> endpoint.</td>
</tr>
<tr>
<td></td>
<td>To modify or add an X.509 host certificate, use the value of the returned key as the value of the <strong>x509_identity</strong> element, and remove any child elements (including the key).</td>
</tr>
</tbody>
</table>

### Elements of server_address

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td><strong>custom_dns</strong> Configures a DNS server that is used to</td>
</tr>
<tr>
<td>Elements of server_address</td>
<td>Type</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>reverse-resolve the hostname if the Channel Policy contains the address of the target as a hostname instead of an IP address. By default, this is disabled and SPS uses the DNS server set in the /api/configuration/network/dns endpoint.</td>
<td></td>
</tr>
</tbody>
</table>

- To use the default DNS, disable this option:

```json
"server_address": {
  "custom_dns": {
    "enabled": false
  },
  ...
},
```

- To use a custom DNS, enable this option and set the IP address of the domain name server to use:

```json
"server_address": {
  "custom_dns": {
    "enabled": true,
    "server": "192.168.1.1"
  },
  ...
},
```

<table>
<thead>
<tr>
<th>selection</th>
<th>string</th>
<th>Configures the address where the clients connect to. Possible values are:</th>
</tr>
</thead>
<tbody>
<tr>
<td>original</td>
<td></td>
<td>Connect to the same address specified by the client.</td>
</tr>
<tr>
<td>nat</td>
<td></td>
<td>Perform a network address translation on the target address. Must be used with the network element.</td>
</tr>
<tr>
<td>fix</td>
<td></td>
<td>Must be used with the address and</td>
</tr>
</tbody>
</table>
### Elements of server_address

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port elements.</td>
<td></td>
</tr>
<tr>
<td>inband</td>
<td>Extract the address of the server from the username. Must be used with the domains element. Optional elements: exception_domains, dns_server, and dns_suffixes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>network</th>
<th>string</th>
<th>Must be used if selection is set to nat. The target address in IP/prefix format. Example: &quot;10.20.30.40/24&quot;.</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if selection is set to fix. The IP address of the target server.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>Must be used if selection is set to fix. The port of the target server.</td>
</tr>
<tr>
<td>domains</td>
<td>Top level list</td>
<td>Must be used if selection is set to inband.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level item</td>
<td>Lists the address ranges that are included in the connection policy.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies if the target address range is provided as a domain or as an IP range. Possible values are:</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The address range of the target server(s). Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
</tbody>
</table>

*Selection* Possible values are:
- address: The value of the target address is an IP range.
- domain: The value of the target address is a domain.
<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>int</td>
<td>The port of the target server(s).</td>
</tr>
<tr>
<td>exception_domains</td>
<td>Top level list</td>
<td>Can only be used if selection is set to inband. Lists the address ranges that are excluded from the connection policy.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level item</td>
<td>Contains the excluded address range.</td>
</tr>
</tbody>
</table>
| selection                  | string | Specifies if the excluded address(es) are provided as a domain or as an IP range. Possible values are:  
  • address  
    The value of the excluded address is an IP range.  
  • domain  
    The value of the excluded address is a domain. |
| value                      | string | The excluded address(es). Use the selection element to specify if the address is an IP range, or a domain. |
| port                       | int  | The excluded port. |
| dns_server                 | string | Can only be used if selection is set to inband. IP address or the hostname of the domain name server used to resolve the address of the target server. |
| dns_suffixes               | list, string | Can only be used if selection is set to inband.  
If the clients do not include the domain name when addressing the server (for example they use username@server instead of username@server.example.com), SPS can automatically add domain information (for example example.com).  
You can add multiple domain names. SPS attempts to resolve the target address by appending the domain names in the
<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>provided order, and uses the first successfully resolved address to establish the connection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of server_side_hostkey</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>plain_hostkey</td>
<td>Top level element</td>
<td>Verifies the identity of the servers based on their hostkeys.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable plain host key checking. If enabled, the hostkey_check element is mandatory.</td>
</tr>
<tr>
<td>hostkey_check</td>
<td>string</td>
<td>Defines the method for checking the host keys of the target server. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disables host key verification.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• accept-first-time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Records the key shown for the first connection, and accepts only the same key for any subsequent connections.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• accept-known-keys</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Only accepts host keys that are already stored on SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can manage host keys at the /api/ssh-host-keys endpoint.</td>
</tr>
</tbody>
</table>

<p>| x509_hostkey                   | Top level element | Verifies the identity of the servers based on their X.509 certificates. |
| enabled                        | string  | Set to true to enable X.509 host key verification. If enabled, the x509_check element is mandatory. |
| x509_check                     | Top level item | Contains the configuration settings for verifying X.509 certificates. |
| selection                      | string  | Configures the validation of X.509 certificates. Possible values are: |
|                                  |        | • disabled |</p>
<table>
<thead>
<tr>
<th>Elements of server_side_hostkey</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Disables X.509 certificate verification.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• accept-first-time Records the X.509 certificate shown for the first connection, and accepts only the same certificate for any subsequent connections.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• accept-known-certificates Only accepts X.509 certificates that are already stored on SPS. You can add X.509 certificates at the /api/ssh-host-keys endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• accept-signed-by Accepts all X.509 certificates that are signed by a trusted Certificate Authority. Must be used with the trusted_ca element.</td>
</tr>
</tbody>
</table>

| trusted_ca | string | Must be used if the selection element is set to accept-signed-by. References the identifier of the trusted CA. You can add or modify the list of trusted CAs at the /api/configuration/policies/trusted_ca_lists/ endpoint. To modify or add a trusted CA, use the value of the returned key as the value of the trusted_ca element, and remove any child elements (including the key). |

**Examples**

For practical purposes, the following examples show only the relevant parts of a connection policy JSON object. To modify or add a connection policy, always submit the full JSON object.

Access control list: configuring the "security" usergroup to only audit connections made by the "root_only" usergroup.
"access_control": [
  {
    "authorizer": "security",
    "permission": "audit",
    "require_different_ip": true,
    "require_different_username": true,
    "subject": {
      "group": "root_only",
      "selection": "only"
    }
  }
]

Target server: use the address specified by the client.

"server_address": {
  "selection": "original"
}

Target server: use a fix address.

"server_address": {
  "address": "<fix-IP>",
  "port": 22,
  "selection": "fix"
}

Target server: configure inband destination selection, where the client can specify the target address in the username. The target can be either an IP range, or a domain.

"server_address": {
  "dns_server": "<ip-of-dns-server>",
  "dns_suffixes": null,
  "domains": [
  {
    "domain": {
      "selection": "address",
      "value": "<IP-range>"
    },
    "port": 22
  },
  {
    "domain": {
      "selection": "domain",
      "value": "*.example"
    }
  }
}
"port": 22
}
,"selection": "inband"
}

Source address: use the same fix IP when connecting to the remote server.

"source_address": {
  "address": "<ip-address>",
  "selection": "fix"
}

Web gateway authentication: require the admin usergroup to perform an additional gateway authentication on the SPS web interface. They must authenticate from the same host which initiated the connection.

"web_gateway_authentication": {
  "enabled": true,
  "groups": [
    "admin"
  ],
  "require_same_ip": true
}

Client-side hostkey: use plain host keys uploaded to SPS, and generate X.509 certificates for the connection.

"client_side_hostkey": {
  "plain_hostkey": {
    "dsa_key": "<id-of-dsa-key>",
    "enabled": true,
    "rsa_key": "<id-of-rsa-key>"
  },
  "x509_hostkey": {
    "enabled": true,
    "x509": {
      "selection": "generate",
      "signing_ca": "<key-of-signing-ca>"
    }
  }
}

Policies: configure only the required policies.
"policies": {
  "aa_plugin": null,
  "analytics_policy": null,
  "archive_cleanup_policy": null,
  "audit_policy": "<key-of-audit-policy>",
  "authentication_policy": "<key-of-auth-policy>",
  "backup_policy": null,
  "channel_policy": "<key-of-channel-policy>",
  "credential_store": null,
  "ldap_server": null,
  "settings": "<key-of-settings-policy>",
  "usermapping_policy": null
}

Server-side hostkey: accept the host key or X.509 certificate presented at the first connection, and require the same host key or certificate for any subsequent connections.

"server_side_hostkey": {
  "plain_hostkey": {
    "enabled": true,
    "hostkey_check": "accept-first-time"
  },
  "x509_hostkey": {
    "enabled": true,
    "x509_check": {
      "selection": "accept-first-time"
    }
  }
}

Server-side hostkey: only accept X.509 certificates that are verified by a trusted CA.

"server_side_hostkey": {
  "plain_hostkey": {
    "enabled": false
  },
  "x509_hostkey": {
    "enabled": true,
    "x509_check": {
      "selection": "accept-signed-by",
      "trusted_ca": "<id-of-trusted-ca>"
    }
  }
}

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Add an SSH connection policy**

To add an SSH connection policy, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new SSH connection policy.**
   
   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/ssh/connections/` endpoint. You can find a detailed description of the available parameters listed in [Element](#).
   
   If the POST request is successful, the response includes the key of the new SSH connection policy. For example:

   ```
   {  
     "key": "a99be49b-b0a2-4cf9-b70d-fea1f9ea188f",  
     "meta": {  
       "href": "/api/configuration/ssh/connections/a99be49b-b0a2-4cf9-b70d-fea1f9ea188f",  
       "parent": "/api/configuration/ssh/connections",  
       "transaction": "/api/transaction"  
     }  
   }
   ```

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.
Modify an SSH connection policy

To modify an SSH connection policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the SSH connection policy.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/ssh/connections/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**SSH channels**

The available SSH channel types and their functionalities are described below. For details on configuring channel policies, see Channel policy.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>auth-agent</td>
<td>None</td>
<td><strong>Agent</strong>: Forwards the SSH authentication agent from the client to the server.</td>
</tr>
<tr>
<td>x11</td>
<td>Yes</td>
<td><strong>X11 Forward</strong>: Forwards the graphical X-server session from the server to the client. List the address of the client in the networks field to permit X11-forwarding only to the specified clients. Specify IP addresses or networks (in IP address/Prefix format). For example:</td>
</tr>
</tbody>
</table>

```
"networks": [ 
    { 
        "selection": "address",
        "value": "192.168.1.1"
    },
    { 
        "selection": "address",
        "value": "192.168.1.2"
    }
]
```
### Channel-specific access control rules:

- **local-forwards** (list): To X11-forwarding only to specific clients, list the IP addresses or networks of the clients in this field. Leave it empty to permit access to every client. For details, see Limiting addresses in port forwarding.

**Local Forward:** Forwards traffic arriving to a local port of the client to a remote host. To enable forwarding only between selected hosts, use the `local_forwards` field. If the `local_forwards` field is empty, local forwarding is enabled without restriction, the client may forward any traffic to the remote host.

For example:

```json
"local_forwards": [
  {
    "host_address": {
      "selection": "address",
      "value": "192.168.100.1"
    },
    "host_port": 5555,
    "originator_address": {
      "selection": "address",
      "value": "192.168.1.1"
    }
  }
]
```

### Channel-specific access control rules:

- **local_forwards** (list): To permit local forwarding only to specific addresses, list the addresses in this field. Leave it empty to enable without restriction. In this case the client may forward any traffic to the remote host.

Enter the source of the forwarded traffic into the `originator_address` field, the target of the traffic into the `host_address` field. Specify IP addresses or networks (in IP address/Prefix format). These parameters are the end-
<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
</table>
| remote-forwards | Yes | **Remote Forward**: Forwards traffic arriving a remote port of the server to the client. To enable forwarding only between selected hosts, enter their IP addresses into the `remote_forwards` field. If the `remote_forwards` field is empty, remote forwarding is enabled without restriction, the SSH server may forward any traffic to the client. For example:

```
"remote_forwards": [
    {
        "connected_address": {
            "selection": "address",
            "value": "192.168.100.1"
        },
        "connected_port": 5555,
        "originator_address": {
            "selection": "address",
            "value": "192.168.1.1"
        }
    }
]
```

Channel-specific access control rules:

- `remote_forwards` (list): To permit only specific forwardings, list the permitted addresses in this field. Leave it empty to permit forwarding without restrictions.

Enter the source of the forwarded traffic into the `originator_address`, the target of the traffic into the `connected_address` field. Specify IP addresses or networks (in IP address/Prefix format). These parameters are the end-points of the forwarded traffic (that is, the remote host that sends data to the client), and not the SSH server. For example, to enable forwarding from the 192.168.20.20 remote host to the client 192.168.50.50, enter 192.168.20.20 into the `originator_address`, and 192.168.50.50 into the `connected_address` field. For details, see Limiting addresses in port forwarding.
<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>details, see Limiting addresses in port forwarding.</td>
</tr>
<tr>
<td>session-exec</td>
<td>Yes</td>
<td><strong>Session Exec</strong>: Execute a remote command (for example <code>rsync</code>) without opening a session shell. List the permitted command in the <code>execs</code> field. You can use regular expressions to specify the commands. This field can contain only letters (a-z, A-Z), numbers (0-9), and the following special characters ({}()*?|[]).</td>
</tr>
</tbody>
</table>

⚠️ **CAUTION:**

Restricting the commands available in Session Exec channels does not guarantee that no other commands can be executed. Commands can be renamed, or executed from shell scripts to circumvent such restrictions.

Channel-specific access control rules:

- `execs (list)`: List the permitted command in the `execs` field. Regular expressions may be used to specify the commands.

For example:

```json
"execs": [
  "top",
  "ls"
]
```

<table>
<thead>
<tr>
<th>session-exec-scp</th>
<th>Yes</th>
<th><strong>Session Exec SCP</strong>: Transfers files using the Secure Copy (SCP) protocol.</th>
</tr>
</thead>
</table>

Channel-specific actions:

- `log_transfer_to_db (list)`: (true|false): Make the list of file operations available in the Search > File operations column of the SPS web interface

- `log_transfer_to_syslog (list)`: (true|false): Send the file operations into the system log

For example:

```json
"actions": {
  "audit": false,
  "four_eyes": false,
  "ids": false,
  "scp": false
}
```
<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>&quot;log_transfer_to_db&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;log_transfer_to_syslog&quot;: true</td>
</tr>
<tr>
<td>session-subsystem</td>
<td>Yes</td>
<td><strong>Session Subsystem</strong>: Use a subsystem. Enter the name of the permitted subsystem into the subsystems field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Channel-specific access control rules:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- subsystems (list): List the permitted subsystems in this field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;execs&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;top&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ls&quot;</td>
</tr>
<tr>
<td>session-exec-sftp</td>
<td>Yes</td>
<td><strong>Session SFTP</strong>: Transfers files using the Secure File Transfer Protocol (SFTP).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Channel-specific actions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- log_transfer_to_db (list): (true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- log_transfer_to_syslog (list): (true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;actions&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;audit&quot;: false,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;four_eyes&quot;: false,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ids&quot;: false,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;log_transfer_to_db&quot;: true,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;log_transfer_to_syslog&quot;: true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>session-shell</td>
<td>Yes</td>
<td><strong>Session Shell</strong>: The traditional remote terminal session.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Channel-specific actions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- content_policy reference: The ID of the Content policy to apply to the connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example:</td>
</tr>
</tbody>
</table>
```
"actions": {
    "audit": true,
    "content_policy": {
        "key": "433849548566ab327522e6"
    },
    "four_eyes": false,
    "ids": false
}
```

**Limiting addresses in port forwarding**

The connected_address, host_address, network, and originator_address options that you can use in SSH channel policies that allow port-forwarding and X11 forwarding have the following parameters.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connected_address, host_address, network, or originator_address</td>
<td>list of JSON objects</td>
<td>Container objects for limiting access to port-forwarding in SSH channel policies. For details, see SSH channels on page 611.</td>
</tr>
<tr>
<td>selection</td>
<td>address or network</td>
<td>Specifies the type of the address. Possible values: address or network</td>
</tr>
<tr>
<td>value</td>
<td>IPv4 address or network</td>
<td>The IP address, or the network in IP-address:prefix format. For example, 192.168.1.1 or 192.168.0.0/16</td>
</tr>
</tbody>
</table>

**SSH authentication policies**

Lists the configured authentication methods that can be used in a connection. Each connection policy uses an authentication policy to determine how the client can authenticate to the target server. Separate authentication methods can be used on the client and the server-side of the connection.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/ssh/authentication_policies
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists SSH authentication policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ssh/authentication_policies
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ssh/authentication_policies<object-id>
```

**Response**

The following is a sample response received when listing SSH authentication policies.

For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "-200",
      "meta": {
        "href": "/api/configuration/ssh/authentication_policies/-200"
    },
    {
      "key": "1895203635707e3340262f",
      "meta": {
```
When retrieving the endpoint of a specific policy, the response is the following.

```json
{
    "body": {
        "backend": {
            "selection": "none"
        },
        "gateway_methods": {
            "kerberos": false,
            "password": false,
            "public_key": false
        },
        "relayed_methods": {
            "kerberos": false,
            "keyboard_interactive": true,
            "password": true,
            "public_key": {
                "selection": "agent"
            }
        }
    },
    "name": "base",
    "key": "-200",
    "meta": {
        "first": "(/api/configuration/ssh/authentication_policies/-200",
        "href": "(/api/configuration/ssh/authentication_policies/-200",
        "last": "(/api/configuration/ssh/authentication_policies/1895203635707e3340262f",
        "next": "(/api/configuration/ssh/authentication_policies/1895203635707e3340262f",
    }
}
```
Elements of authentication policies | Type | Description |
--- | --- | --- |
key | string | Top level element, contains the ID of the policy. |
body | Top level element | Contains the elements of the policy. |
backend | Top level item | The authentication database used on the client-side. |
gateway_methods | Top level item | Client-side gateway authentication settings. The value of selection defines which authentication method is used. |
mode | Top level element | Obsolete node. Any settings submitted into this node is ignored. In a response, this node may contain inaccurate data. |\nname | string | The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace. |
relayed_methods | Top level element | Server-side authentication settings. |

Elements of backend | Type | Description |
--- | --- | --- |
selection | string | Defines the authentication method for client-side gateway authentication. Possible values are: |
| | | - none |
| | | Disables client-side gateway authentication. |
| | | - 1dap |
| | | Uses the LDAP server selected for the connection policy. LDAP servers can be configured in the /api/configuration/policies/ldap_servers endpoint). |
| | | To use this option, you must also configure the password and
<table>
<thead>
<tr>
<th>Elements of backend</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set it to true to enable public key-based client-side authentication.</td>
</tr>
<tr>
<td>user_database</td>
<td>string</td>
<td>References the key of the local user database. You can configure local user databases at the /api/configuration/policies/user_databases/ endpoint. To modify or add a local user database, use the value of the returned key as the value of the user_database element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>servers</td>
<td>Top level list</td>
<td>Defines the properties of the RADIUS servers used for client-side authentication. A valid list item consists of the address, port and shared_secret elements.</td>
</tr>
<tr>
<td>address</td>
<td>Top level element</td>
<td>Defines the address of a RADIUS server.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Required child of the address element. Possible values are:</td>
</tr>
</tbody>
</table>

- **local**
  - Uses the local user database configured in the `/api/configuration/policies/user_databases/` endpoint.
  - To use this option, you must also configure the password, public_key, and user_database elements.

- **radius**
  - Uses one or more Radius servers for authentication.
  - To use this option, you must also configure the authentication_protocol and servers elements.
### Elements of backend

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td>string</td>
<td>The value element contains the IP of the RADIUS server.</td>
</tr>
<tr>
<td>fqdn</td>
<td>string</td>
<td>The value element contains the FQDN of the RADIUS server.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The IP or the FQDN address of the RADIUS server.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number of the RADIUS server.</td>
</tr>
<tr>
<td>shared_secret</td>
<td>string</td>
<td>References the key of the shared secret for the RADIUS server. You can configure shared secrets at the <code>/api/configuration/passwords/</code> endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a shared secret, use the value of the returned key as the value of the shared_secret element, and remove any child elements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(including the key).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Alternatively, you can include the new password as plain text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;shared_secret&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;plain&quot;: &quot;:&lt;new-password&gt;&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

### Elements of gateway_methods

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kerberos</td>
<td>boolean</td>
<td>Authentication based using Kerberos. Set it to true to enable Kerberos-based client-side authentication. If required, you can select other</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gateway authentication methods in addition to Kerberos, and also authentication backends and related to the selected gateway authentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>methods.</td>
</tr>
<tr>
<td>Elements of gateway_methods</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To use Kerberos authentication on the target server, you must use Kerberos authentication both on the SPS gateway and on the target server (in relayed_methods).</td>
</tr>
<tr>
<td>password</td>
<td>boolean</td>
<td>Authentication based on username and password. Set it to true to enable password-based client-side authentication.</td>
</tr>
<tr>
<td>public_key</td>
<td>Top level item</td>
<td>Authentication based on public-private encryption keypairs.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of relayed_methods</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kerberos</td>
<td>boolean</td>
<td>Authentication based using Kerberos. Set it to true to enable Kerberos-based client-side authentication. If required, you can select other gateway authentication methods in addition to Kerberos, and also authentication backends and related to the selected gateway authentication methods. To use Kerberos authentication on the target server, you must use Kerberos authentication both on the SPS gateway and on the target server (in relayed_methods).</td>
</tr>
<tr>
<td>keyboard_interactive</td>
<td>boolean</td>
<td>Authentication based on exchanging messages between the user and the server. This method includes authentication schemes like S/Key or TIS authentication. Depending on the configuration of the SSH server, might have to be used together with password-based authentication. Set to true to enable interactive authentication on the remote server.</td>
</tr>
<tr>
<td>password</td>
<td>boolean</td>
<td>Authentication based on username and password. Set to true to enable password-based authentication on the remote server.</td>
</tr>
<tr>
<td>public_key</td>
<td>Top level item</td>
<td>Authentication based on public-private encryption keypairs. Use the selection child element to disable or</td>
</tr>
<tr>
<td>Elements of relayed_ methods</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures authentication on the remote server using public-private keypairs. The following values are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disables the authentication method.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• publish_to_ldap</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPS generates a keypair, and uses this keypair in the server-side connection. The public key of this keypair is also uploaded to the LDAP database set in the LDAP Server of the connection policy. That way the server can authenticate the client to the generated public key stored under the user's username in the LDAP database. You can configure LDAP servers using the /api/configuration/policies/ldap_servers endpoint, and connection policies using the /api/configuration/ssh/connections endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses a private key in the server-side connection. You have to use the private_key element to reference the private key.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• agent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Allow the client to use agent-forwarding, and use its own keypair on the server-side. If this option is used, SPS requests the client to use its SSH agent to authenticate on the target server. Therefore, you must configure your clients to enable agent forwarding, otherwise authentication will fail. For details on enabling agent forwarding in your SSH application, see the documentation of the application.</td>
</tr>
</tbody>
</table>
Elements of *relayed_* methods

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private_key</td>
<td>References the key of the private key used for authenticating with a public-private keypair on the remote server. You can configure private keys at the <code>/api/configuration/private_keys/</code> endpoint. To modify or add a private key, use the value of the returned key as the value of the <code>private_key</code> element, and remove any child elements (including the key).</td>
</tr>
</tbody>
</table>

**Examples:**

Password authentication against LDAP on the client side, and using a username and password on the remote server:

```
{
    "backend": {
        "selection": "ldap"
    },
    "gateway_methods": {
        "kerberos": false,
        "password": true,
        "public_key": false
    },
    "name": "password_ldap",
    "relayed_methods": {
        "kerberos": false,
        "keyboard_interactive": false,
        "password": true,
        "public_key": {
            "selection": "disabled"
        }
    }
}
```

Password authentication against a local user database on SPS, and using a username and password on the remote server. You can find the key of the local user database is available at the `/api/configuration/policies/user_databases/` endpoint.

```
{
    "backend": {
        "selection": "local",
        "user_database": "<key-of-the-local-user-database>"
    },
```
Authenticating against an RADIUS server on the client side, and using a username and password on the remote server. You can configure the key of the shared secret at the `/api/configuration/passwords/` endpoint. The IP of the RADIUS server is used.
Using Kerberos authentication both on the client side and on the remote server.

```json
{
  "backend": {
    "selection": "none"
  },
  "gateway_methods": {
    "kerberos": true,
    "password": false,
    "public_key": false
  },
  "name": "kerberos_only",
  "relayed_methods": {
    "kerberos": true,
    "keyboard_interactive": false,
    "password": true,
    "public_key": {
      "selection": "disabled"
    }
  }
}
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>--------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Add an SSH authentication policy**

To add an SSH authentication policy, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new policy.**

   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/ssh/authentication_policies/` endpoint. You can find a detailed description of the available parameters listed in [Elements of authentication policies](#). The elements of `gateway_methods` are listed in [Elements of gateway methods](#). The elements of `relayed_methods` are listed in [Elements of relayed methods](#).

   If the POST request is successful, the response includes the key of the new policy. For example:

   ```json
   {
     "key": "6f924f39-e4c9-4b0f-8018-8842e2115ebd",
     "meta": {
       "href": "/api/configuration/ssh/authentication_policies/6f924f39-e4c9-4b0f-8018-8842e2115ebd",
       "parent": "/api/configuration/ssh/authentication_policies",
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Modify an SSH authentication policy**

To modify an SSH authentication policy, you have to:

1. **Open a transaction.**

   For more information, see [Open a transaction](#) on page 29.
2. **Modify the JSON object of the policy.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/ssh/authentication_policies/<key-of-the-object>` endpoint. You can find a detailed description of the available parameters listed in [Elements of authentication policies](#). The elements of gateway_methods are listed in [Elements of gatewayMethods](#). The elements of relayed_methods are listed in [Elements of relayedMethods](#).

3. **Commit your changes.**

   For more information, see *Commit a transaction* on page 31.

### Global SSH options

List of options that affect all SSH connections.

**URL**

GET `https://<IP-address-of-SPS>/api/configuration/ssh/options`

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists global SSH options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ssh/options
```
Response

The following is a sample response received when listing global SSH options.

For details of the meta object, see Message format on page 10.

```json
{
  "body": {
    "audit": {
      "cleanup": {
        "channel_database_cleanup_days": 600,
        "enabled": true
      },
      "timestamping": {
        "selection": "local",
        "signing_interval": 30
      }
    },
    "gssapi": {
      "enabled": false
    },
    "service": {
      "enabled": true,
      "log_level": 4
    }
  },
  "key": "options",
  "meta": {
    "first": "/api/configuration/ssh/authentication_policies",
    "href": "/api/configuration/ssh/options",
    "last": "/api/configuration/ssh/settings_policies",
    "next": "/api/configuration/ssh/settings_policies",
    "parent": "/api/configuration/ssh",
    "previous": "/api/configuration/ssh/connections",
    "transaction": "/api/transaction"
  }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Top level item</td>
<td>Contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level item</td>
<td>Contains the elements of the global SSH options.</td>
</tr>
<tr>
<td>audit</td>
<td>Top level item</td>
<td>Contains settings for timestamping and cleanup.</td>
</tr>
<tr>
<td>service</td>
<td>Top level item</td>
<td>Global setting to enable SSH connections, and</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>item</td>
<td>item</td>
<td>specify the logging detail.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable SSH connections.</td>
</tr>
<tr>
<td>log_level</td>
<td>int</td>
<td>Defines the logging detail of SSH connections.</td>
</tr>
<tr>
<td>gssapi</td>
<td>Top level item</td>
<td>Deprecated setting.</td>
</tr>
</tbody>
</table>

**Elements of audit**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanup</td>
<td>Top level item</td>
<td>Global retention settings for SSH connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.</td>
</tr>
<tr>
<td>channel_database_cleanup_days</td>
<td>int</td>
<td>Global retention time for the metadata of SSH connections, in days. Must exceed the retention time of the archiving policy (or policies) used for SSH connections, and the connection-specific database cleanup times (if configured).</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>To enable the global cleanup of SSH connection metadata, set this element to true.</td>
</tr>
<tr>
<td>timestamping</td>
<td>Top level item</td>
<td>Global timestamping settings for SSH connections.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures local or remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set local to use SPS for timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set remote to configure a remote timestamping server.</td>
</tr>
<tr>
<td>server_url</td>
<td>string</td>
<td>Required for remote timestamping. The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.</td>
</tr>
<tr>
<td>oid</td>
<td>Top level</td>
<td>The Object Identifier of the policy</td>
</tr>
<tr>
<td>Elements of audit</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required for remote timestamping. Set to true to configure the Object Identifier of the timestamping policy on the timestamping remote server.</td>
</tr>
<tr>
<td>policy_oid</td>
<td>string</td>
<td>Required if the oid is enabled. The Object Identifier of the timestamping policy on the remote timestamping server.</td>
</tr>
<tr>
<td>signing_interval</td>
<td>int</td>
<td>Time interval for timestamping open connections, in seconds.</td>
</tr>
</tbody>
</table>

**Examples:**

Set SPS as the timestamping server:

```json
{
  "audit": {
    "cleanup": {
      "enabled": false
    },
    "timestamping": {
      "selection": "local",
      "signing_interval": 30
    }
  },
  "gssapi": {
    "enabled": false
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Enable cleanup, and set it to occur every 10 days:

```json
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    }
  }
}
```
Change timestamping to a remote server, without specifying a timestamping policy:

```
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": false
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "gssapi": {
        "enabled": false
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, and specify the 1.2.3 timestamping policy:

```
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": false
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "gssapi": {
        "enabled": false
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```
"oid": {
   "enabled": true,
   "policy_oid": "1.2.3"
},
   "selection": "remote",
   "server_url": "<url-of-timestamping-server>",
   "signing_interval": 30
},
"gssapi": {
   "enabled": false
},
"service": {
   "enabled": true,
   "log_level": 4
}

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Modify global SSH settings

To modify global SSH settings, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.
2. **Modify the JSON object of the global SSH settings endpoint.**

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/ssh/options endpoint. You can find a detailed description of the available parameters listed in Element. The elements of the audit item are described in Elements of audit.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

## SSH settings policies

SSH settings policies define protocol-level settings (algorithms, greetings and banners, timeout). You can create multiple policies, and choose the appropriate one for each SSH connection.

### URL

GET https://<IP-address-of-SPS>/api/configuration/ssh/settings_policies

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists SSH settings policies.
The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/ssh/settings_policies/<policy-id>
```

**Response**

The following is a sample response received when listing SSH settings policies. For details of the meta object, see Message format on page 10.

```
{
   "items": [
      {
         "key": "-300",
         "meta": {
            "href": "/api/configuration/ssh/settings_policies/-300"
         }
      },
      {
         "key": "236283841571912b948b88",
         "meta": {
            "href": "/api/configuration/ssh/settings_policies/236283841571912b948b88"
         }
      }
   ],
   "meta": {
      "first": "/api/configuration/ssh/authentication_policies",
      "href": "/api/configuration/ssh/settings_policies",
      "last": "/api/configuration/ssh/settings_policies",
      "next": null,
      "parent": "/api/configuration/ssh",
      "previous": "/api/configuration/ssh/options",
      "transaction": "/api/transaction"
   }
}
```

When retrieving the endpoint of a specific policy, the response is the following.

```
{
   "body": {
      "name": "default",
      "timeout": 600,
      "inactivity_timeout": {
         "enabled": true
      }
   }
}
```
"value": 13000
},
"preconnect_channel_check": false,
"greeting": "",
"userauth_banner": "",
"software_version": "SSH",
"strict_mode": true,
"client_side_algorithms": {
    "kex": ["diffie-hellman-group14-sha1", "diffie-hellman-group1-sha1"],
    "cipher": ["aes128-ctr", "aes192-ctr", "aes256-ctr", "aes128-cbc",
        "arcfour"],
    "mac": ["hmac-sha1", "hmac-md5"],
    "compression": ["none"]
},
"server_side_algorithms": {
    "kex": ["diffie-hellman-group14-sha1", "diffie-hellman-group1-sha1"],
    "cipher": ["aes128-ctr", "aes192-ctr", "aes256-ctr", "aes128-cbc",
        "arcfour"],
    "mac": ["hmac-sha1", "hmac-md5"],
    "compression": ["none"]
},
"key": "236283841571912b948b88",
"meta": {
    "first": "/api/configuration/ssh/settings_policies/-300",
    "href": "/api/configuration/ssh/settings_policies/236283841571912b948b88",
    "last": "/api/configuration/ssh/settings_policies/236283841571912b948b88",
    "next": null,
    "parent": "/api/configuration/ssh/settings_policies",
    "previous": "/api/configuration/ssh/settings_policies/-300",
    "transaction": "/api/transaction"
}
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cipher</td>
<td>list</td>
<td>Lists the permitted client-side cipher algorithms.</td>
</tr>
<tr>
<td>compression</td>
<td>list</td>
<td>Lists the permitted client-side compression algorithms.</td>
</tr>
<tr>
<td>kex</td>
<td>list</td>
<td>Lists the permitted client-side KEX algorithms.</td>
</tr>
<tr>
<td>mac</td>
<td>list</td>
<td>Lists the permitted client-side MAC algorithms.</td>
</tr>
<tr>
<td>greeting</td>
<td>string</td>
<td>Greeting message for the connection.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the SSH settings policy.</td>
</tr>
<tr>
<td>preconnect_channel_check</td>
<td>boolean</td>
<td>Before establishing the server-side connection, SPS can evaluate the connection and channel policies to determine if the connection might be permitted at all. The server-side connection is established only if the evaluated policies permit the client to access the server. To enable this function, set the parameter to true.</td>
</tr>
<tr>
<td>server_side_algorithms</td>
<td>Top level element (list)</td>
<td>Lists the permitted server-side encryption parameters.</td>
</tr>
<tr>
<td>cipher</td>
<td>list</td>
<td>Lists the permitted server-side cipher algorithms.</td>
</tr>
<tr>
<td>compression</td>
<td>list</td>
<td>Lists the permitted server-side compression algorithms.</td>
</tr>
<tr>
<td>kex</td>
<td>list</td>
<td>Lists the permitted server-side KEX algorithms.</td>
</tr>
<tr>
<td>mac</td>
<td>list</td>
<td>Lists the permitted server-side MAC algorithms.</td>
</tr>
<tr>
<td>software_version</td>
<td>string</td>
<td>Specifies additional text to append to the SSH protocol banner sent by the server upon connection.</td>
</tr>
<tr>
<td>strict_mode</td>
<td>boolean</td>
<td>When this option is enabled, SPS rejects connections that use unrealistic parameters (for example,</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>terminals</td>
<td></td>
<td>terminals of thousand by thousand characters) and port-forwarding connections where the address in the port-forwarding request and the channel-opening request does not match. Note that this can interfere with certain client or server applications. Strict mode is allowed by default. To turn it off, set the parameter to false.</td>
</tr>
<tr>
<td>timeout</td>
<td>int</td>
<td>Connection timeout, in seconds.</td>
</tr>
<tr>
<td>inactivity_timeout</td>
<td>Top level element</td>
<td></td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>• true: If no user activity is detected, it terminates the session after the configured time has passed since the last user activity.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: No user inactivity timeout.</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>Only if enabled is true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of user activity timeout. Must be greater than or equal to the value of timeout.</td>
</tr>
<tr>
<td>userauth_banner</td>
<td>string</td>
<td>You can display a banner message to the clients before authentication (as specified in RFC 4252 â The Secure Shell (SSH) Authentication Protocol). You can use this banner to inform the users that the connection is audited.</td>
</tr>
</tbody>
</table>

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>----------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Add SSH settings policies

To add a settings policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new policy.**
   
   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/ssh/settings_policies/` endpoint. You can find a detailed description of the available parameters listed in Element.
   
   If the POST request is successful, the response includes the key of the new policy. For example:

   ```json
   { 
     "key": "59790911-415c-4ed3-a0d2-1164637472ca",
     "meta": { 
       "href": "/api/configuration/ssh/settings_policies/59790911-415c-4ed3-a0d2-1164637472ca",
       "parent": "/api/configuration/ssh/settings_policies",
       "transaction": "/api/transaction"
     } 
   }
   ```

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

Modify SSH settings policies

To modify a settings policy, you have to:
1. **Open a transaction.**
   
   For more information, see **Open a transaction on page 29.**

2. **Modify the JSON object of the policy.**

   PUT the modified JSON object to the **https://<IP-address-of-SPS>/api/configuration/ssh/settings_policies/<key-of-the-object>** endpoint. You can find a detailed description of the available parameters listed in **Element**.

3. **Commit your changes.**

   For more information, see **Commit a transaction on page 31.**

### SSH host keys and certificates

SPS stores the host keys and X.509 certificates of the trusted servers. When a client tries to connect to a server, SPS verifies the host key or the certificate of the server, and allows connections only to the servers that have their keys available on SPS (unless the SSH Connection Policy is configured differently).

#### URL

GET **https://<IP-address-of-SPS>/api/ssh-host-keys**

#### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d03e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API on page 19.</strong></td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists the SSH host keys and certificates of the servers that the users can connect to using SSH.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/ssh-host-keys/
```

The following command retrieves the properties of a specific key.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/ssh-host-keys/<object-id>
```

Response

The following is a sample response received when listing SSH host keys and certificates from the https:<IP-address-of-SPS>/api/ssh-host-keys/ endpoint.

For details of the meta object, see Message format on page 10.

The key of these objects is in the following format: <type-of-the-key>-<host-address>::<host-port>.

```
{
    "meta": {
        "href": "/api/ssh-host-keys",
        "parent": "/api"
    },
    "items": [
        {
            "key": "ssh-dss-10.110.0.1:22",
            "meta": {
                "href": "/api/ssh-host-keys/ssh-dss-10.110.0.1:22"
            }
        },
        {
            "key": "ssh-dss-10.110.0.2:2222",
            "meta": {
                "href": "/api/ssh-host-keys/ssh-dss-10.110.0.2:2222"
            }
        },
        {
            "key": "ssh-rsa-10.110.0.1:22",
            "meta": {
                "href": "/api/ssh-host-keys/ssh-rsa-10.110.0.1:22"
            }
        },
        {
            "key": "x509v3-sign-rsa-d00::2222:dead:2222",
            "meta": {
                "href": "/api/ssh-host-keys/x509v3-sign-rsa-d00::2222:dead:2222"
            }
        }
    ]
}
```

When retrieving the endpoint of a specific host key, the response is the following.
Element                                      Type      Description                                                  
---                                           ---        ---                                                          
key                                          string    Top level element, contains the ID of the host key or certificate in the following format: <type-of-the-key>-<host-address>:<host-port>.  
<id-of-the-host-key>                        Top level element (string) The ID of the host key or certificate in the following format: <type-of-the-key>-<host-address>:<host-port>.  
address                                      string    The IPv4 or IPv6 address of the host that the key belongs to. Note that for IPv6 addresses, this is always the canonical format of the address.  
port                                         number    The port number where the host uses the key or certificate.  
type                                         JSON object The ID of the host key or certificate in the following format: <type-of-the-key>-<host-address>:<host-port>.  
selection                                    string    Specifies the type of the host key. Possible values: ssh-rsa, ssh-dss, x509v3-sign-rsa, x509v3-sign-dss  
value                                        string    The host key or certificate as a string in PEM format.  

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
### Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Search and filter host keys

To list only specific host keys, you can use the following filters.

- List every host key and certificate:
  
  ```
  GET https://<IP-address-of-SPS>/api/ssh-host-keys
  ```

- List host keys of a specific type:
  
  ```
  GET https://<IP-address-of-SPS>/api/ssh-host-keys?type=<type-to-list>
  ```

  Possible values: ssh-rsa, ssh-dss, x509v3-sign-rsa, x509v3-sign-dss. For example:

  ```
  GET https://<IP-address-of-SPS>/api/ssh-host-keys?type=ssh-rsa
  ```

- List host keys for a specific port number:
  
  ```
  GET https://<IP-address-of-SPS>/api/ssh-host-keys?port=<port-number-to-list>
  ```

- List host keys for a specific host address (IPv4 or IPv6):
  
  ```
  GET https://<IP-address-of-SPS>/api/ssh-host-keys?address=<host-address>
  ```

- For a complex filter, separate the parameters with an ampersand (&) character, for example:
  
  ```
  GET https://<IP-address-of-SPS>/api/ssh-host-keys?port=<port-number-to-list>&type=<type-to-list>
  ```
The response to such requests is a JSON object, where the `items` list includes the IDs of the selected host keys (or an empty list). For example, filtering for ssh-dss keys could return a similar list:

```json
{
   "meta": {
      "href": "/api/ssh-host-keys",
      "parent": "/api"
   },
   "items": [
      {
         "key": "ssh-dss-10.110.0.1:22",
         "meta": {
            "href": "/api/ssh-host-keys/ssh-dss-10.110.0.1:22"
         }
      },
      {
         "key": "ssh-dss-10.110.0.2:2222",
         "meta": {
            "href": "/api/ssh-host-keys/ssh-dss-10.110.0.2:2222"
         }
      }
   ]
}
```

### Add new host key

To upload a new host key or certificate, you have to POST the host key and other data as a JSON object to the `https://<IP-address-of-SPS>/api/ssh-host-keys` endpoint. For details, see [Create a new object](on page 45). The body of the POST request must contain a JSON object with the parameters listed in [Element](#). If the POST request is successful, the response includes an ID for the host key in the following format: `<type-of-the-key>-<host-address>:<host-port>`. For example:

```json
{
   "address": "10.110.0.1",
   "port": 22,
   "type": {
      "selection": "ssh-rsa",
      "value": "AAAAB3NzaC1yc2EAAAAD...zvMwg=="
   }
}
```

Note that for IPv6 addresses, SPS will automatically convert the address to its canonical format.

### Delete host key

To delete a host key or certificate, you have to DELETE `https://<IP-address-of-SPS>/api/ssh-host-keys/ID-of-the-host-key` endpoint. For details, see [Delete an object](#) on page 43. If the DELETE request is successful, the response includes only the meta object, for example:

```json
{}
```
You must commit your changes to actually delete the object from SPS.
Telnet connections

List of endpoints for configuring the policies, options and connection rules of Telnet connections.

URL

GET https://<IP-address-of-SPS>/api/configuration/telnet

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available settings for configuring for Telnet connections.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/telnet
```
Response

The following is a sample response received when listing the configuration settings. For details of the meta object, see Message format on page 10.

```
{
   "items": [
   {
   "key": "authentication_policies",
   "meta": { "href": "/api/configuration/telnet/authentication_policies" }
   },
   {
   "key": "channel_policies",
   "meta": { "href": "/api/configuration/telnet/channel_policies" }
   },
   {
   "key": "connections",
   "meta": { "href": "/api/configuration/telnet/connections" }
   },
   {
   "key": "options",
   "meta": { "href": "/api/configuration/telnet/options" }
   },
   {
   "key": "pattern_sets",
   "meta": { "href": "/api/configuration/telnet/pattern_sets" }
   }
   ],
   "meta": {
   "first": "/api/configuration/aaa",
   "href": "/api/configuration/telnet",
   "last": "/api/configuration/x509",
   "next": "/api/configuration/troubleshooting",
   "parent": "/api/configuration",
   "previous": "/api/configuration/ssh",
   "remaining_seconds": 600,
   "transaction": "/api/transaction"
   }
}
```

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connections</td>
<td>List of Telnet connection policies.</td>
</tr>
<tr>
<td>channel_policies</td>
<td>List of available Telnet channel types.</td>
</tr>
<tr>
<td>Item</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>authentication policies</td>
<td>List of the configured authentication methods that can be used in a connection.</td>
</tr>
<tr>
<td>pattern_sets</td>
<td>List of the default and custom channel policies.</td>
</tr>
<tr>
<td>options</td>
<td>List of global Telnet options that affect all connections.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Telnet connection policies

Connection policies determine if a server can be accessed from a particular client. Connection policies reference other resources (policies, usergroups, keys) that must be configured and available before creating a connection policy.

### URL

GET https://<IP-address-of-SPS>/api/configuration/telnet/connections/
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists Telnet connection policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/telnet/connections/
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/telnet/connections/<connection-key>
```

Response

The following is a sample response received when listing Telnet connection policies. For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "8348340645707e2575e3c6",
            "meta": {
                "href": "/api/configuration/telnet/connections/8348340645707e2575e3c6"
            }
        },
        
        "meta": {
            "first": "/api/configuration/telnet/channel_policies",
```
When retrieving the endpoint of a specific Telnet Connection Policy, the response is the following.

```json
{
  "body": {
    "access_control": [],
    "active": true,
    "channel_database_cleanup": {
      "enabled": false
    },
    "client_side_transport_security": {
      "selection": "disabled"
    },
    "indexing": {
      "enabled": true,
      "policy": {
        "key": "-50000",
        "meta": {
          "href": "/api/configuration/policies/indexing/-50000"
        }
      }
    },
    "priority": 3
  },
  "log_audit_trail_downloads": true,
  "name": "demo_telnet",
  "network": {
    "clients": [
      "0.0.0.0/0"
    ],
    "ports": [23],
    "targets": [
      "10.30.255.0/24"
    ]
  },
  "override_log_level": {
```
"enabled": true,
"log_level": 3
},
"policies": {
"aa_plugin": null,
"analytics_policy": {
"key": "20509709385cd578654cdab",
"meta": {
  "href": "/api/configuration/policies/analytics/20509709385cd578654cdab"
}
},
"archive_cleanup_policy": null,
"audit_policy": {
  "key": "78101850949e47437dd91d",
  "meta": {
    "href": "/api/configuration/policies/audit_policies/78101850949e47437dd91d"
  }
},
"authentication_policy": {
  "key": "-400",
  "meta": {
    "href": "/api/configuration/telnet#authentication_policies/-400"
  }
},
"backup_policy": null,
"channel_policy": {
  "key": "-30200",
  "meta": {
    "href": "/api/configuration/telnet/channel_policies/-30200"
  }
},
"credential_store": null,
"ldap_server": null,
"settings": {
  "key": "-302",
  "meta": {
    "href": "/api/configuration/telnet#settings_policies/-302"
  }
},
"usermapping_policy": null
},
"rate_limit": {
  "enabled": false
},
"server_address": {

}
```json
"custom_dns": {
   "enabled": false
},
"selection": "original"
},
"server_side_transport_security": {
   "selection": "disabled"
},
"source_address": {
   "selection": "box_address"
},
"web_gateway_authentication": {
   "enabled": false
}
},
"key": "18762920615d68fa3d858d0",
"meta": {
   "first": "/api/configuration/telnet/connections/18762920615d68fa3d858d0",
   "href": "/api/configuration/telnet/connections/18762920615d68fa3d858d0",
   "last": "/api/configuration/telnet/connections/18762920615d68fa3d858d0",
   "next": null,
   "parent": "/api/configuration/telnet/connections",
   "previous": null,
   "remaining_seconds": 600,
   "transaction": "/api/transaction"
}
```

### Element Type Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the connection policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the connection policy.</td>
</tr>
<tr>
<td>access_control</td>
<td>Top level list</td>
<td>Collection of access policies. Access policies define who can authorize and audit a connection.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>Set to false to suspend the connection policy. Connection settings are preserved.</td>
</tr>
<tr>
<td>channel_</td>
<td>Top</td>
<td>Configures cleanup of the connection</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>database_cleanup</td>
<td>level item</td>
<td>metadata on the connection policy's level.</td>
</tr>
<tr>
<td>client_side_transport_security</td>
<td>Top level item</td>
<td>Defines the Transport Layer Security (TLS) settings for the connection between SPS and the client. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;client_side_transport_security&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;disabled&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>days</td>
<td>int</td>
<td>Retention time, in days. Must not exceed the retention time of the archive_cleanup_policy, and the retention time configured in the global settings of the protocol.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The global settings of the Telnet protocol are available at the api/configuration/telnet/options endpoint.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable periodical cleanup of the connection metadata.</td>
</tr>
<tr>
<td>indexing</td>
<td>Top level item</td>
<td>Configures indexing for the connection policy.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable indexing the connections.</td>
</tr>
<tr>
<td>policy</td>
<td>string</td>
<td>References the identifier of the indexing policy. You can configure indexing policies at the /api/configuration/policies/indexing/endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add an indexing policy, use the value of the returned key as the value of the policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>priority</td>
<td>int</td>
<td>Specifies the indexing priority for the connection. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Very low priority.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 4</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>log_audit_trail_downloads</td>
<td>boolean</td>
<td>Set to true to log audit trail downloads.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the connection policy.</td>
</tr>
<tr>
<td>network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clients</td>
<td>list, string</td>
<td>List of client (&quot;from&quot;) IP addresses.</td>
</tr>
<tr>
<td>ports</td>
<td>list, integer-s</td>
<td>List of target ports.</td>
</tr>
<tr>
<td>targets</td>
<td>list, string</td>
<td>List of target IP addresses.</td>
</tr>
</tbody>
</table>
| override_log_level     | Top level item | Specifies the verbosity level of sessions handled by this connection policy. If disa
<p>|                        |          | The log level of other connection policies is not affected. If disabled, the log level set at the /api/configuration/&lt;protocol&gt;/options endpoint is used. |
|                        |          |                                                                             |
|                        |          | To use the default log level, disable this option:                         |
|                        |          | &quot;override_log_level&quot;: {                                                    |
|                        |          |   &quot;enabled&quot;: false                                                        |
|                        |          | },                                                                          |
|                        |          |                                                                             |
|                        |          | To use a custom log level for the connection policy, enable this option and set the log level to use: |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policies</td>
<td>Top level item</td>
<td>List of policies referenced by the connection policy.</td>
</tr>
<tr>
<td>aa_plugin</td>
<td>string</td>
<td>References the identifier of the AA plug-in. You can configure AA plug-ins at the /api/configuration/plugins/aa/ endpoint. To modify or add an AA plug-in, use the value of the returned key as the value of the aa_plugin element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>analytics_policy</td>
<td>string</td>
<td>References the identifier of the analytics policy. You can configure analytics policies at the /api/configuration/analytics/ endpoint. To add or modify an analytics policy, use the value of the returned key as the value of the analytics element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>archive_cleanup_policy</td>
<td>string</td>
<td>References the identifier of the archive/cleanup policy. You can configure archive and cleanup policies at the /api/configuration/policies/archive_cleanup_policies/ endpoint. To modify or add an archive/cleanup policy, use the value of the returned key as the value of the archive_cleanup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>audit_policy</td>
<td>string</td>
<td>Cannot be null. References the identifier of the audit policy. You can configure audit policies at the /api/configuration/policies/audit_policies/ endpoint. To modify or add an audit policy, use the value of the returned key as the value of the audit policy.</td>
</tr>
</tbody>
</table>

"override_log_level": {
  "enabled": true,
  "log_level": 5
},
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>audit_policy</td>
<td>string</td>
<td>Cannot be null. References the identifier of the authentication policy. Note that currently you cannot create or modify Telnet Authentication Policies using the REST API. Use the web UI instead. To modify or add an authentication policy, use the value of the returned key as the value of the authentication_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>backup_policy</td>
<td>string</td>
<td>References the identifier of the backup policy. You can configure backup policies at the /api/configuration/policies/backup_policies/ endpoint. To modify or add a backup policy, use the value of the returned key as the value of the backup_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>channel_policy</td>
<td>string</td>
<td>References the identifier of the channel policy. The value of this option cannot be null. To modify or add a channel policy, use the value of the returned key as the value of the channel_policy element, and remove any child elements (including the key). You can configure Telnet channel policies at the /api/configuration/telnet/channel_policies/ endpoint.</td>
</tr>
<tr>
<td>credential_store</td>
<td>string</td>
<td>References the identifier of the credential store. You can configure credential stores at the /api/configuration/policies/credentialstores/ endpoint. To modify or add a credential store, use the value of the returned key as the value of the credential_store element, and</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ldap_server</td>
<td>string</td>
<td>References the identifier of the LDAP server. You can configure LDAP servers at the /api/configuration/policies/ldap_servers/ endpoint. To modify or add an LDAP server, use the value of the returned key as the value of the ldap_server element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>settings</td>
<td>string</td>
<td>References the identifier of the settings policy. The value of this option cannot be null. To modify or add a settings policy for this protocol, use the value of the returned key as the value of the settings element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>usermapping_policy</td>
<td>string</td>
<td>References the identifier of a Usermapping Policy. You can configure Usermapping Policies at the /api/configuration/policies/usermappi ng_policies/ endpoint. To modify or add a Usermapping Policy, use the value of the returned key as the value of the usermapping_policies element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>rate_limit</td>
<td>Top level element</td>
<td>Connection rate limit.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to provide a connection rate limit.</td>
</tr>
<tr>
<td>value</td>
<td>int</td>
<td>The number of connections (per minute) that are allowed in the connection policy.</td>
</tr>
<tr>
<td>server_address</td>
<td>Top level item</td>
<td>Defines the address where the clients connect to.</td>
</tr>
<tr>
<td>server_side_</td>
<td>Top</td>
<td>Defines the Transport Layer Security</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>transport_security</td>
<td>level item</td>
<td>(TLS) settings for the connection between SPS and the server. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;server_side_transport_security&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;disabled&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>source_address</td>
<td>Top level element</td>
<td>Allows you to configure Source Network Address Translation (SNAT) on the server side of SPS. SNAT determines the IP address SPS uses in the server-side connection. The target server will see the connection coming from this address.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures Source Network Address Translation. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- box_address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default. Uses the network address of the logical interface of SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- original</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses the IP address of the client, as seen by SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- fix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses a fixed address when connecting to the remote server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the address element.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if the value of the selection element is set to fix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The IP address to use as the source address in server-side connections.</td>
</tr>
<tr>
<td>web_gateway_authentication</td>
<td>Top level item</td>
<td>When gateway authentication is required for a connection, the user must authenticate on SPS as well. This additional authentication can be performed out-of-band on the SPS web interface for every protocol.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable additional gateway authentication on the SPS web interface.</td>
</tr>
<tr>
<td>groups</td>
<td>list,</td>
<td>By default, any user can perform</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>string</td>
<td>gateway authentication for the connections. You can restrict authentication to members of specific usergroups. Define the usergroups at the <code>/api/configuration/aaa/local_database/groups/</code> endpoint, and list the name of each group here.</td>
<td></td>
</tr>
<tr>
<td>require_same_ip</td>
<td>boolean</td>
<td>Set to true to only accept web gateway authentication from the same host that initiated the connection.</td>
</tr>
</tbody>
</table>

### Elements of access control

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizer</td>
<td>string</td>
<td>The usergroup (local or LDAP) who can authorize or audit the connection. Local usergroups can be added or modified at the <code>/api/configuration/aaa/local_database/groups/</code> endpoint.</td>
</tr>
<tr>
<td>permission</td>
<td>string</td>
<td>Defines the permissions of the authorizer usergroup. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• audit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• authorize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the authorize permission can authorize connection requests.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• audit_and_authorize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The usergroup with the audit_and_authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.</td>
</tr>
<tr>
<td>require_different_ip</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different IP addresses.</td>
</tr>
<tr>
<td>require</td>
<td>boolean</td>
<td>Set to true to require the authorizing user and its subject to have different IP addresses.</td>
</tr>
</tbody>
</table>
### Elements of access control

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>different_username</td>
<td>subject to have different usernames.</td>
</tr>
<tr>
<td>subject</td>
<td>Top level item Defines the subjects of the access control policy.</td>
</tr>
<tr>
<td>group</td>
<td>The usergroup (local or LDAP) that is subject to the access control policy. Local usergroups can be added or modified at the <code>/api/configuration/aaa/local_database/groups/</code> endpoint.</td>
</tr>
</tbody>
</table>
| selection             | Possible values:  
|                       |  - everybody  
|                       |    Every user is subject to the access control policy.  
|                       |  - only  
|                       |    Requires the group element. Members of the usergroup specified in the group element are subject to the access control policy. |

### Elements of client_side_transport_security

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>peer_certificate_check</td>
<td>Top level item Sets how SPS authenticates the peers. To permit connections from peers without requesting a certificate, set &quot;enabled&quot;: false, for example:</td>
</tr>
</tbody>
</table>
|                       | "peer_certificate_check": {  
|                       |    "enabled": false  
|                       | }  
|                       | To validate the certificate of the peer, set "enabled": true, and reference a trusted certificate authority list, for example: |

SPS 6.13.0 REST API Reference Guide  
Telnet connections
<table>
<thead>
<tr>
<th>Elements of client-side transport security</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
<td>Sets the encryption settings used between SPS and the client. When the connection is encrypted, SPS has to show a certificate to the client, so you must configure the sps_certificate option as well. The possible values of selection are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- starttls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enable encrypted connections that use the STARTTLS method. Note that the peer must use the STARTTLS method. Unencrypted connections will be terminated after a brief period.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- tls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Require encryption.</td>
</tr>
<tr>
<td>sps_certificate</td>
<td>JSON object</td>
<td>Sets the certificate that SPS shows to the peer when the communication is encrypted. SPS can either use the same certificate for every session, or generate a separate certificate for each session.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- To use the same certificate for every session, set selection: &quot;fix&quot; and reference the certificate to use in the x509_identity option, for example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sps_certificate&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;fix&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;x509_identity&quot;: &quot;'&lt;key' of an uploaded certificate&gt;'&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For details on uploading certificates to SPS, see Certificates stored on SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- To generate a certificate for every session, set selection: &quot;generate&quot; and reference the certificate authority to sign the generated certificates in the signing_ca option, for example:</td>
</tr>
</tbody>
</table>
## Elements of client-side transport security

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sps_certificate</td>
<td></td>
</tr>
</tbody>
</table>

```
"sps_certificate": {
  "selection": "generate",
  "signing_ca": "2221b768-0722-4298-9e16-ce67eb3723ad"
},
```

For details on using signing certificates, see Signing CA policies.

## Elements of server_address

### Elements of server_address

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>custom_dns</td>
<td>Configures a DNS server that is used to reverse-resolve the hostname if the Channel Policy contains the address of the target as a hostname instead of an IP address. By default, this is disabled and SPS uses the DNS server set in the /api/configuration/network/dns endpoint.</td>
</tr>
</tbody>
</table>

- To use the default DNS, disable this option:

  ```
  "server_address": {
    "custom_dns": {
      "enabled": false
    },
    ...
  },
  ```

- To use a custom DNS, enable this option and set the IP address of the domain name server to use:

  ```
  "server_address": {
    "custom_dns": {
      "enabled": true,
      "server": "192.168.1.1"
    }
  },
  ```
<table>
<thead>
<tr>
<th>Elements of server_address</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures the address where the clients connect to. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- original</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Connect to the same address specified by the client.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- nat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Perform a network address translation on the target address.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the network element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- fix</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the address and port elements.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- inband</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extract the address of the server from the username.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Must be used with the domains element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Optional elements: exception_domains, dns_server, and dns_suffixes.</td>
</tr>
<tr>
<td>network</td>
<td>string</td>
<td>Must be used if selection is set to nat. The target address in IP/prefix format. Example: &quot;10.20.30.40/24&quot;.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>Must be used if selection is set to fix. The IP address of the target server.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>Must be used if selection is set to fix. The port of the target server.</td>
</tr>
<tr>
<td>domains</td>
<td>Top level list</td>
<td>Must be used if selection is set to inband.</td>
</tr>
<tr>
<td>domain</td>
<td>Top level item</td>
<td>Lists the address ranges that are included in the connection policy.</td>
</tr>
<tr>
<td>Elements of server_address</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies if the target address range is provided as a domain or as an IP range. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the target address is an IP range.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- domain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the target address is a domain.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The address range of the target server(s).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port of the target server(s).</td>
</tr>
<tr>
<td>exception_domains</td>
<td>Top</td>
<td>Can only be used if selection is set to inband.</td>
</tr>
<tr>
<td></td>
<td>level</td>
<td>Lists the address ranges that are excluded from the connection policy.</td>
</tr>
<tr>
<td></td>
<td>list</td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>Top</td>
<td>Contains the excluded address range.</td>
</tr>
<tr>
<td></td>
<td>level</td>
<td></td>
</tr>
<tr>
<td></td>
<td>item</td>
<td></td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Specifies if the excluded address(es) are provided as a domain or as an IP range. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the excluded address is an IP range.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- domain</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the excluded address is a domain.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The excluded address(es).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use the selection element to specify if the address is an IP range, or a domain.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The excluded port.</td>
</tr>
<tr>
<td>dns_server</td>
<td>string</td>
<td>Can only be used if selection is set to inband.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IP address or the hostname of the</td>
</tr>
</tbody>
</table>
### Elements of server_address

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain name server</td>
<td>domain name server used to resolve the address of the target server.</td>
</tr>
</tbody>
</table>

**dns_suffixes**

- **Type**: list, string
- **Description**: Can only be used if selection is set to inband.

If the clients do not include the domain name when addressing the server (for example they use username@server instead of username@server.example.com), SPS can automatically add domain information (for example example.com).

You can add multiple domain names. SPS attempts to resolve the target address by appending the domain names in the provided order, and uses the first successfully resolved address to establish the connection.

### Elements of server_side_transport_security

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>peer_certificate_check</td>
<td>Sets how SPS authenticates the peers. To permit connections from peers without requesting a certificate, set &quot;enabled&quot;: false, for example:</td>
</tr>
</tbody>
</table>

```
"peer_certificate_check": {
  "enabled": false
}
```

To validate the certificate of the peer, set "enabled": true, and reference a trusted certificate authority list, for example:

```
"peer_certificate_check": {
  "enabled": true,
  "trusted_ca": "cfc815e5-dadb-4eb9-a628-12ae0c12d358"
}
```
### Elements of server-side transport security

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>selection</strong></td>
<td>Sets the encryption settings used between SPS and the server. If SPS has to show a certificate to the peer, so you must configure the sps_certificate option as well. The possible values of selection are:</td>
</tr>
<tr>
<td>string</td>
<td>- <strong>none</strong>&lt;br&gt;Do not use encryption.</td>
</tr>
<tr>
<td></td>
<td>- <strong>starttls</strong>&lt;br&gt;Enable encrypted connections that use the STARTTLS method. Note that the peer must use the STARTTLS method. Unencrypted connections will be terminated after a brief period.</td>
</tr>
<tr>
<td></td>
<td>- <strong>tls</strong>&lt;br&gt;Require encryption.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>sps_certificate</strong></th>
<th>JSON object</th>
<th>Sets the certificate that SPS shows to the peer when the communication is encrypted. SPS can either use the same certificate for every session, or generate a separate certificate for each session.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>- If the server does not require a certificate from SPS, set selection: &quot;none&quot;.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- To use the same certificate for every session, set selection: &quot;fix&quot; and reference the certificate to use in the x509_identity option, for example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sps_certificate&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;selection&quot;: &quot;fix&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;x509_identity&quot;: &quot;&lt;key' of an uploaded certificate&gt;&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
</tbody>
</table>

For details on uploading certificates to SPS, see **Certificates stored on SPS**.

- To generate a certificate for every session, set selection: "generate" and reference the certificate authority to sign the generated certificates in the signing_ca option, for example:
Elements of server-side transport security

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

```
"sps_certificate": {
  "selection": "generate",
  "signing_ca": "2221b768-0722-4298-9e16-ce67eb3723ad"
},
```

For details on using signing certificates, see Signing CA policies.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Add a Telnet connection policy

To add a Telnet connection policy, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 29.
2. **Create the JSON object for the new Telnet connection policy.**

   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/telnet/connections/` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

   If the POST request is successful, the response includes the key of the new Telnet connection policy. For example:

   ```json
   {
     "key": "a99be49b-b0a2-4cf9-b70d-f0a3f9ea188f",
     "meta": {
       "href": "/api/configuration/telnet/connections/a99be49b-b0a2-4cf9-b70d-f0a3f9ea188f",
       "parent": "/api/configuration/telnet/connections",
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Modify a Telnet connection policy**

To modify a Telnet connection policy, you have to:

1. **Open a transaction.**

   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the connection policy.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/telnet/connections/<key-of-the-object>` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Telnet channels**

The available Telnet channel types and their functionalities are described below.
<table>
<thead>
<tr>
<th>Channel</th>
<th>Special options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>telnet</td>
<td>Yes</td>
<td><strong>telnet</strong>: Enables access to the server’s terminal. This channel must be enabled for Telnet to work.</td>
</tr>
</tbody>
</table>

Channel-specific actions:

- content_policy reference: The ID of the Content policy to apply to the connection.

For example:

```
"actions": {
    "audit": true,
    "four_eyes": true,
    "content_policy": {
        "key": "43849548566ab327522e6"
        "meta": {
            "href": "/api/configuration/policies/content_policies/44287216854f482e7f2b24"
        }
    },
}
```

**Telnet authentication policies**

Lists the configured authentication methods that can be used in a connection. Each connection policy uses an authentication policy to determine how the client can authenticate on the SPS gateway.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/telnet/authentication_policies
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860.</td>
</tr>
</tbody>
</table>
Sample request

The following command lists Telnet authentication policies.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/telnet/authentication_policies
```

The following command retrieves the properties of a specific policy.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/telnet/authentication_policies<object-id>
```

Response

The following is a sample response received when listing Telnet authentication policies.

For details of the meta object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "-200",
      "meta": {
        "href": "/api/configuration/telnet/authentication_policies/-200"
      }
    },
    {
      "key": "-304002001",
      "meta": {
        "href": "/api/configuration/telnet/authentication_policies/-304002001"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/telnet/authentication_policies",
```
When retrieving the endpoint of a specific policy, the response is the following.

```
{
  "body": {
    "active_pattern_sets": [],
    "backend": {
      "selection": "ldap"
    },
    "name": "telnet_auth_policy_with_ldap"
  }
}
```

<table>
<thead>
<tr>
<th><strong>Element</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>key</strong></td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td><strong>body</strong></td>
<td>Top level</td>
<td>Contains the elements of the policy.</td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>string</td>
<td>The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td><strong>active_pattern_sets</strong></td>
<td>JSON list</td>
<td>The list of patterns to use to extract the username from the sessions. For details, see . For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;active_pattern_sets&quot;: [&quot;-8000&quot;,&quot;-8001&quot;,&quot;-8002&quot;]</td>
</tr>
<tr>
<td><strong>backend</strong></td>
<td>Top level</td>
<td>Client-side gateway authentication settings. The value of selection defines which authentication method is used.</td>
</tr>
<tr>
<td><strong>selection</strong></td>
<td>string</td>
<td>Defines the authentication method for client-side gateway authentication. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* none</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enables client-side gateway authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ldap</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses the LDAP server selected for the connection policy. LDAP servers can</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be configured in the /api/configuration/policies/ldap_servers endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• local</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses the local user database configured in the /api/configuration/policies.userIdatabases/endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To use this option, you must also configure the user_database element.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• radius</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uses one or more Radius servers for authentication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To use this option, you must also configure the authentication_protocol and servers elements.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of servers</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>Top level element</td>
<td>Defines the address of a RADIUS server.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Required child of the address element. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ip</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains the IP of the RADIUS server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fqdn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value element contains the FQDN of the RADIUS server.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The IP or the FQDN address of the RADIUS server.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number of the RADIUS server.</td>
</tr>
<tr>
<td>shared_secret</td>
<td>string</td>
<td>References the key of the shared secret for the RADIUS server. You can configure shared secrets at the /api/configuration/passwords/ endpoint.</td>
</tr>
</tbody>
</table>
To modify or add a shared secret, use the value of the returned key as the value of the shared_secret element, and remove any child elements (including the key).

Alternatively, you can include the new password as plain text.

"shared_secret": {
  "plain": "<new-password>"
}

Examples:

Querying base authentication policy without gateway authentication:

```
{
  "key": "-304002001",
  "body": {
    "name": "base",
    "backend": {
      "selection": "none"
    }
  }
}
```

Querying authentication policy with LDAP backend:

```
{
  "key": "telnet-auth-pol-2",
  "body": {
    "name": "telnet_ldap",
    "backend": {
      "selection": "ldap",
      "timeout": 3600,
      "keepalive": true
    }
  }
}
```

Querying authentication policy with local backend:
Querying authentication policy with RADIUS backend:

```json
{
    "key": "telnet-auth-pol-4",
    "body": {
        "name": "telnet_radius",
        "backend": {
            "selection": "radius",
            "servers": [
                {
                    "address": {
                        "selection": "ip",
                        "value": "1.2.3.4"
                    },
                    "port": 1812,
                    "shared_secret": {
                        "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXX",
                        "meta": { "href": "/api/configuration/passwords#XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXX" }
                    }
                }
            ],
            "authentication_protocol": "pap",
            "timeout": 3600,
            "keepalive": true
        }
    }
}
```
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Add a Telnet authentication policy

To add a Telnet authentication policy, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new policy.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/telnet/authentication_policies/ endpoint. You can find a detailed description of the available parameters listed in Telnet authentication policies.

   If the POST request is successful, the response includes the key of the new policy.
   
   For example:

   ```json
   {
     "key": "6f924f39-e4c9-4b0f-8018-8842e2115ebd",
     "meta": {
       "href": "/api/configuration/telnet/authentication_policies/6f924f39-
     }
   }
   ```
3. **Commit your changes.**
   For more information, see [Commit a transaction on page 31](#).

### Modify a Telnet authentication policy

To modify a Telnet authentication policy, you have to:

1. **Open a transaction.**
   For more information, see [Open a transaction on page 29](#).

2. **Modify the JSON object of the policy.**
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/telnet/authentication_policies/<key-of-the-object>` endpoint. You can find a detailed description of the available parameters listed in [Telnet authentication policies](#).

3. **Commit your changes.**
   For more information, see [Commit a transaction on page 31](#).

### Global Telnet options

List of options that affect all Telnet connections.

#### URL

```
GET https://<IP-address-of-SPS>/api/configuration/telnet/options
```
## Cookies

<table>
<thead>
<tr>
<th>Cookie Name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d03e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists global Telnet options.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/telnet/options
```

### Response

The following is a sample response received when listing global Telnet options.

For details of the `meta` object, see [Message format](#) on page 10.

```
{
  "body": {
    "audit": {
      "cleanup": {
        "enabled": false
      },
      "timestamping": {
        "selection": "local",
        "signing_interval": 30
      }
    },
    "service": {
      "enabled": true,
      "log_level": 4
    }
  },
  "key": "options",
```

SPS 6.13.0 REST API Reference Guide

Telnet connections
### Element | Type | Description
--- | --- | ---
key | Top level item | Contains the ID of the endpoint.

**body** | Top level item | Contains the elements of the global Telnet options.

**audit** | Top level item | Contains settings for timestamping and cleanup.

**service** | Top level item | Global setting to enable Telnet connections, and specify the logging detail.

| enabled | boolean | Set to true to enable Telnet connections.
| log_level | int | Defines the logging detail of Telnet connections.

### Elements of audit

**cleanup** | Top level item | Global retention settings for Telnet connection metadata. To configure retention time for a specific connection policy, use the `archive_cleanup_policy` element at the endpoint of the policy instead.

| channel_database_cleanup_days | int | Global retention time for the metadata of Telnet connections, in days. Must exceed the retention time of the archiving policy (or policies) used for Telnet connections, and the connection-specific database cleanup times (if configured).

| enabled | boolean | To enable the global cleanup of Telnet connection metadata, set
<table>
<thead>
<tr>
<th>Elements of audit</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamping</td>
<td>Top level item</td>
<td>Global timestamping settings for Telnet connections.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures local or remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Set local to use SPS for timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Set remote to configure a remote timestamping server.</td>
</tr>
<tr>
<td>server_url</td>
<td>string</td>
<td>Required for remote timestamping. The URL of the timestamping server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that HTTPS and password-protected connections are not supported.</td>
</tr>
<tr>
<td>oid</td>
<td>Top level item</td>
<td>The Object Identifier of the policy used for timestamping.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Required for remote timestamping.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set to true to configure the Object Identifier of the timestamping policy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on the timestamping remote server.</td>
</tr>
<tr>
<td>policy_oid</td>
<td>string</td>
<td>Required if the oid is enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The Object Identifier of the timestamping policy on the remote timestamping</td>
</tr>
<tr>
<td>signing_interval</td>
<td>int</td>
<td>Time interval for timestamping open connections, in seconds.</td>
</tr>
</tbody>
</table>

**Examples:**

Set SPS as the timestamping server:

```
{
    "audit": {
        "cleanup": {
            "enabled": false
        },
        "timestamping": {
            "selection": "local",
            "signing_interval": 30
        }
    }
```
Enable cleanup, and set it to occur every 10 days:

```
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "selection": "local",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": false
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, and specify the 1.2.3 timestamping policy:

```
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": false
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```
Modify global Telnet settings

To modify global Telnet settings, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the global Telnet settings endpoint.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/telnet/options` endpoint. You can find a detailed description of the available parameters listed in [Element](#). The elements of the audit item are described in [Elements of audit](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes](#) on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The <code>details</code> section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The <code>details</code> section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Telnet pattern sets**

List of Telnet pattern sets that help to extract the username from Telnet connections.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/telnet/pattern_sets
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API</strong> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
**Sample request**

The following command lists the available Telnet pattern sets.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/telnet/pattern_sets
```

**Response**

The following is a sample response received when listing the available Telnet pattern sets. For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "body": {
                "name": "Cisco devices",
                "description": "Cisco devices that require authentication"
            },
            "key": "-8000",
            "meta": { "href": "/api/configuration/telnet/pattern_sets/-8000" }
        },
        {
            "body": {
                "name": "Cisco devices without authentication",
                "description": "Cisco devices that do not require authentication"
            },
            "key": "-8001",
            "meta": { "href": "/api/configuration/telnet/pattern_sets/-8001" }
        },
        {
            "body": {
                "name": "General Telnet",
                "description": "General Telnet servers (for example, Linux telnetd)"
            },
            "key": "-8002",
            "meta": { "href": "/api/configuration/telnet/pattern_sets/-8002" }
        }
    ],
    "meta": {
        "first": "/api/configuration/telnet/authentication_policies",
        "href": "/api/configuration/telnet/pattern_sets",
        "last": "/api/configuration/telnet/pattern_sets",
        "next": null,
        "parent": "/api/configuration/telnet"
    }
}
```
To upload a telnet pattern set:

1. Open a transaction.
   For more information, see Open a transaction.
2. Upload the telnet pattern set file.
   POST the valid_pattern_set.tps file to the https://<IP-address-of-SPS>/api/upload/pattern_set endpoint, for example:

   curl -X POST --cookie cookies --insecure https://<IP-address-of-SPS>/api/upload/pattern_set --data-binary @<path-to-pattern_set.tps>

   The following is a sample response received:
   For details of the meta object, see Message format.

   ```
   {
     "body": {
       "description": "Test Pattern Set description",
       "name": "Test Pattern Set"
     },
     "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
     "meta": {
       "href": "/api/configuration/telnet/pattern_sets/XXXXXXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
       "parent": "/api/configuration/telnet/pattern_sets"
     }
   }
   ```
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>PatternSetValidationFailed</td>
<td>The validation of the telnet pattern set file failed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
</tbody>
</table>

3. Commit your changes.

For more information, see Commit a transaction.

Note the following points:

- Built-in telnet pattern sets cannot be deleted. The ID of built-in pattern sets begins with "-".
- Existing pattern sets cannot be re-loaded, only if you delete them beforehand.

Operations with the /pattern_sets endpoint

- /api/configuration/telNet/pattern_sets: GET
- /api/configuration/telNet/pattern_sets/<id>: GET, DELETE
- /api/upload/pattern_set: POST

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
<tr>
<td>405</td>
<td>DeleteNotAllowedOnBuiltInPatternSet</td>
<td>The deletion of built-in telnet pattern sets is not allowed.</td>
</tr>
</tbody>
</table>
VNC connections

List of endpoints for configuring the policies, options and connection rules of VNC connections.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/vnc

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command lists the available settings for configuring for VNC connections.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/vnc
```
Response

The following is a sample response received when listing the configuration settings.

For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "channel_policies",
            "meta": {
                "href": "/api/configuration/vnc/channel_policies"
            }
        },
        {
            "key": "connections",
            "meta": {
                "href": "/api/configuration/vnc/connections"
            }
        },
        {
            "key": "options",
            "meta": {
                "href": "/api/configuration/vnc/options"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/aaa",
        "href": "/api/configuration/vnc",
        "last": "/api/configuration/x509",
        "next": "/api/configuration/x509",
        "parent": "/api/configuration",
        "previous": "/api/configuration/troubleshooting",
        "transaction": "/api/transaction"
    }
}
```

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel_policies</td>
<td>List of the default and custom channel policies.</td>
</tr>
<tr>
<td>connections</td>
<td>List of the VNC connection policies.</td>
</tr>
<tr>
<td>options</td>
<td>List of global VNC options that affect all connections.</td>
</tr>
</tbody>
</table>

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**VNC connection policies**

Connection policies determine if a server can be accessed from a particular client. Connection policies reference other resources (policies, usergroups, keys) that must be configured and available before creating a connection policy.

⚠️ **CAUTION:**

The connection policies of this protocol are available in READ-ONLY mode on the REST API. Also, the returned data is incomplete, it does not include any protocol-specific settings, only the parameters that are common to every supported protocol.

To modify the connection policies of this protocol, you must use the SPS web interface.

Using the REST API, you can modify the connection policies of the RDP and SSH protocols.

**URL**

```
GET https://<IP-address-of-SPS>/api/configuration/vnc/connections/
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860.</td>
</tr>
</tbody>
</table>
### Cookie name

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
</table>
| session_id  | Contains the authentication token of the user | Required | The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists VNC connection policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/vnc/connections/
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/vnc/connections/<connection-key>
```

### Global VNC options

List of options that affect all VNC connections.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/vnc/options

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
</table>
| session_id  | Contains the authentication token of the user | Required | The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.
Sample request

The following command lists global VNC options.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/vnc/options
```

Response

The following is a sample response received when listing global VNC options.

For details of the meta object, see Message format on page 10.

```
{
    "body": {
        "audit": {
            "cleanup": {
                "enabled": false
            },
            "timestamping": {
                "selection": "local",
                "signing_interval": 30
            }
        },
        "service": {
            "enabled": true,
            "log_level": 4
        }
    },
    "key": "options",
    "meta": {
        "first": "/api/configuration/vnc/channel_policies",
        "href": "/api/configuration/vnc/options",
        "last": "/api/configuration/vnc/options",
        "next": null,
    }
}
```
"parent": "/api/configuration/vnc",
"previous": "/api/configuration/vnc/channel_policies",
"transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Top level item</td>
<td>Contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level item</td>
<td>Contains the elements of the global VNC options.</td>
</tr>
<tr>
<td>audit</td>
<td>Top level item</td>
<td>Contains settings for timestamping and cleanup.</td>
</tr>
<tr>
<td>service</td>
<td>Top level item</td>
<td>Global setting to enable VNC connections, and specify the logging detail.</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>Set to true to enable VNC connections.</td>
</tr>
<tr>
<td>log_level</td>
<td>int</td>
<td>Defines the logging detail of VNC connections.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of audit</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanup</td>
<td>Top level item</td>
<td>Global retention settings for VNC connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.</td>
</tr>
<tr>
<td>channel_databaseCleanup_days</td>
<td>int</td>
<td>Global retention time for the metadata of VNC connections, in days. Must exceed the retention time of the archiving policy (or policies) used for VNC connections, and the connection-specific database cleanup times (if configured).</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean</td>
<td>To enable the global cleanup of VNC connection metadata, set this element to true.</td>
</tr>
<tr>
<td>timestamping</td>
<td>Top level item</td>
<td>Global timestamping settings for VNC connections.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures local or remote VNC connections.</td>
</tr>
</tbody>
</table>
Elements of audit

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| timestamping  | - Set local to use SPS for timestamping.  
|               | - Set remote to configure a remote timestamping server. |

| server_url    | string | Required for remote timestamping.  
The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported. |
| oid           | Top level item | The Object Identifier of the policy used for timestamping. |
| enabled       | boolean | Required for remote timestamping.  
|               |          | Set to true to configure the Object Identifier of the timestamping policy on the timestamping remote server. |
| policy_oid    | string | Required if the oid is enabled.  
The Object Identifier of the timestamping policy on the remote timestamping server. |
| signing_interval | int | Time interval for timestamping open connections, in seconds. |

Examples:

Set SPS as the timestamping server:

```
{
    "audit": {
        "cleanup": {
            "enabled": false
        },
        "timestamping": {
            "selection": "local",
            "signing_interval": 30
        }
    },
}
```
Enable cleanup, and set it to occur every 10 days:

```json
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "selection": "local",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```json
{
    "audit": {
        "cleanup": {
            "channel_database_cleanup_days": 10,
            "enabled": true
        },
        "timestamping": {
            "oid": {
                "enabled": false
            },
            "selection": "remote",
            "server_url": "<url-of-timestamping-server>",
            "signing_interval": 30
        }
    },
    "service": {
        "enabled": true,
        "log_level": 4
    }
}
```

Change timestamping to a remote server, and specify the 1.2.3 timestamping policy:
Modify global VNC settings

To modify global VNC settings, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the global VNC settings endpoint.**
   
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/vnc/options` endpoint. You can find a detailed description of the available parameters listed in [Element](#). The elements of the audit item are described in [Elements of audit](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes](#) on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Search, download, and index sessions

Audited sessions

The api/audit/sessions endpoint lists the recorded sessions (active and closed).

URL

GET https://<IP-address-of-SPS>/api/audit/sessions

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the connections.

curl --cookie cookies https://<IP-address-of-SPS>/api/audit/sessions
The following command retrieves the properties of a specific connection.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/audit/sessions/<session-id>
```

**Response**

The following is a sample response received when listing connections. For details of the meta object, see [Message format](#) on page 10.

```json
{
  "items": [
    {
      "key": "2",
      "meta": {
        "href": "/api/audit/sessions/2"
      }
    },
    {
      "key": "1",
      "meta": {
        "href": "/api/audit/sessions/1"
      }
    }
  ],
  "meta": {
    "fields": [],
    "first": "/api/audit/sessions?limit=500&offset=0&fields=",
    "href": "/api/audit/sessions",
    "last": "/api/audit/sessions?limit=500&offset=0&fields=",
    "limit": 500,
    "match_count": 39,
    "next": null,
    "offset": 0,
    "parent": "/api/audit",
    "previous": null
  }
}
```

When retrieving the endpoint of a specific connection, the response is the following.

```json
{
  "body": {
    "active": false,
    "alerts": {
      "href": "/api/audit/sessions/rUhhQZ3jYs1NDWYp9DEpq/alerts"
    },
    "analytics": {
```
"interesting_events": [],
"scripted": false,
"scripted_results": {},
"similar_sessions": [],
"tags": []
},
"channels": {
  "href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq/channels"
},
"client": {
  "ip": "10.20.30.40",
  "name": "10.20.30.40",
  "port": 59125
},
"creation_time": "2018-11-14T12:26:59.244Z",
"duration": 57,
"end_time": "2018-09-15T14:22:00+05:00",
"events": {
  "href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq/events"
},
"hidden": false,
"indexing": {
  "href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq/indexing"
},
"node_id": "6fed7872-065e-41d2-9cfa-ba75e8cad901",
"origin": "RECORDING",
"phantom": false,
"protocol": "SSH",
"recording": {
  "archived": false,
  "audit_trail": {
    "archive": null,
    "download": {
      "href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq/audit_trail"
    }
  },
  "auth_method": "password",
  "channel_policy": "shell-only",
  "command_extracted": false,
  "connection_policy": "myconnectionpolicy",
  "connection_policy_id": "15682863055beac3c8d23bf",
  "content_reference_id": 30,
  "has_accepted_channel": true,
  "index_status": "INDEXED",
  "server_local": {
    "ip": "10.20.30.40",
    "name": "10.20.30.40",
    "port": 55386
  }
}
```
},
"session_id": "svc/rUhQZ3jYsY1NDWyp9DEpq/abcde:29",
"target": {
  "ip": "10.20.30.40",
  "name": "10.20.30.40",
  "port": 221
},
"verdict": "Accepted",
"window_title_extracted": false
},
"revision": 15,
"server": {
  "ip": "10.20.30.40",
  "name": "10.20.30.40",
  "port": 22
},
"start_time": "2018-09-15T15:53:00+05:00",
"user": {
  "id": "myid",
  "name": "myname",
  "server_username": "myserver"
},
"verdict": "ACCEPT"
},
"key": "rUhQZ3jYsY1NDWyp9DEpq",
"meta": {
  "href": "/api/audit/sessions/rUhQZ3jYsY1NDWyp9DEpq",
  "parent": "/api/audit/sessions",
  "remaining_seconds": 594
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the key of the connection or audit trail.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the properties of the connection.</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>If the returned value is true, the connection is ongoing.</td>
</tr>
<tr>
<td>alerts</td>
<td>Top level</td>
<td>Contains a link to the details of the alerts. For details, see Session alerts on page 750.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>item</td>
<td></td>
<td>An event is listed as alert only if the <strong>Actions &gt; Store in Connection Database</strong> option is selected in the <strong>Content Policy</strong> used to handle the session.</td>
</tr>
<tr>
<td>analytics</td>
<td>Top level</td>
<td>Contains analytics details of the connection.</td>
</tr>
<tr>
<td>channels</td>
<td>Top level</td>
<td>Contains a link to the details of the channel.</td>
</tr>
<tr>
<td>client</td>
<td>Top level</td>
<td>The IP address and port number of the client.</td>
</tr>
<tr>
<td>creation_time</td>
<td>date</td>
<td>The time this document was created. In optimal cases this is near equal to the session's original <strong>start_time</strong>. However, it can be later than <strong>start_time</strong>.</td>
</tr>
<tr>
<td>duration</td>
<td>int</td>
<td>The duration of the session in seconds. Computed value.</td>
</tr>
<tr>
<td>end_time</td>
<td>ISO 8601</td>
<td>The timestamp of the end of the connection. For ongoing connection, the value is null. Starting with SPS 5 LTS, the timestamp is in ISO 8601 format, for example, 2018-10-11T09:23:38.000+02:00. In earlier versions, it was in UNIX timestamp format.</td>
</tr>
<tr>
<td>events</td>
<td>Top level</td>
<td>Contains a link to the details of the events. For details, see <strong>Session events</strong> on page 754.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>hidden</td>
<td>boolean</td>
<td>True if this is a session that has not been displayed on the SPS GUI yet (due to fragmented data about the session).</td>
</tr>
<tr>
<td>indexer</td>
<td>Top level</td>
<td>Contains the details of indexing. For details on configuring indexing, see Local services: configuring the indexer on page 764.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;indexer&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;href&quot;: &quot;/api/audit/sessions/rUhhQZjYsY1NDWYp9DEpq/indexer&quot;</td>
</tr>
<tr>
<td>node_id</td>
<td>string</td>
<td>The node ID of the SPS machine where this session has been recorded.</td>
</tr>
<tr>
<td>origin</td>
<td>string</td>
<td>How SPA received this session. The following values are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PSM for sessions based on an audit trail recorded by SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LOG for sessions built from log data.</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>The protocol of the connection.</td>
</tr>
<tr>
<td>recording</td>
<td>Top level</td>
<td>Contains the properties of the audit trail.</td>
</tr>
<tr>
<td>archived</td>
<td>boolean</td>
<td>If the audit trail has been archived, this value is true, otherwise it is false. For details about thearchiving, see the archive object of</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the psm.audit_trail field.</td>
</tr>
<tr>
<td>audit_trail</td>
<td>Top level</td>
<td>The path to the audit trail file on SPS. If the session does not have an audit trail, this element is not used. To download the audit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>trail, see Download audit trails on page 716.</td>
</tr>
<tr>
<td>auth_method</td>
<td>Top level</td>
<td>Authentication method: The authentication method used in the connection. For example, password</td>
</tr>
<tr>
<td>channel_policy</td>
<td>string</td>
<td>References the name of the channel policy. You can find the list of channel policies for each protocol at</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>command_extracted</td>
<td>boolean</td>
<td>If commands have been extracted from this terminal session, this value is true, otherwise it is false. The extracted commands are available in the events object field.</td>
</tr>
<tr>
<td>connection_policy</td>
<td>string</td>
<td>The name of the Connection Policy that handled the session, for example, ssh_gateway_auth. This is the name displayed on the Control &gt; Connections page of the SPS web interface, and in the name field of the Connection Policy object. You can find the list of connection policies for each protocol at the /api/configuration/&lt;protocol&gt;/connections/ endpoint.</td>
</tr>
<tr>
<td>connection_policy_id</td>
<td>string</td>
<td>The key of the Connection Policy that handled the session, for example, 54906683158e768e727100. You can find the list of connection policies for each protocol at the /api/configuration/&lt;protocol&gt;/connections/ endpoint.</td>
</tr>
<tr>
<td>content_reference_id</td>
<td>long</td>
<td>The unique ID of the TCP connection.</td>
</tr>
<tr>
<td>has_accepted_channel</td>
<td>boolean</td>
<td>True, if at least the connection has been built successfully, the authentication was successful, and there was actual traffic.</td>
</tr>
<tr>
<td>index_status</td>
<td>string</td>
<td><strong>Channel's indexing status</strong>: Shows if the channel has been indexed. The following values are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CHANNEL_OPEN (0): The connection of the channel is still open (indexer is waiting for the connection to close).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NOT_INDEXED (1): All channels of the connection have been closed which belong to the connection. The channel is ready for indexing, unless the audit trail was placed in the skipped_connections queue.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• INDEXING_IN_PROGRESS (2): The channel is being indexed (indexing in progress). Note that SPS will return search results for the parts of the channel that are already indexed.</td>
</tr>
</tbody>
</table>

SPS 6.13.0 REST API Reference Guide
Search, download, and index sessions
- **INDEXED (3):** Indexing the channel is complete.
- **INDEXING NOT REQUIRED (4):** Indexing not required (indexing is not enabled for the connection).
- **INDEXING FAILED (5):** Indexing failed. The indexer service writes the corresponding error message in the error_message column of the indexer_jobs table. Note that SPS will return search results for the parts of the channel that were successfully indexed before the error occurred. For example, if the error occurred at the end of a long audit trail, you can still search for content from the first part of the audit trail.
- **NO TRAIL (6):** Auditing is not enabled for the channel.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network_id</td>
<td>string</td>
<td>The ID of the Linux network namespace where the session originated from.</td>
</tr>
<tr>
<td>server_local</td>
<td>Top level item</td>
<td>The IP address and port number of SPS.</td>
</tr>
<tr>
<td>session_id</td>
<td>string</td>
<td>The identifier of the session.</td>
</tr>
<tr>
<td>target</td>
<td>Top level item</td>
<td>The IP address and port number the client targeted for connection.</td>
</tr>
<tr>
<td>verdict</td>
<td>string</td>
<td>The connection verdict. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- accept</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The connection attempt was successful.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- accept-terminated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The connection violated a content policy, and was terminated by SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- auth-fail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authentication failure.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- deny</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The connection was denied.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- fail</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>The connection attempt failed.</td>
<td></td>
<td>The connection attempt failed.</td>
</tr>
<tr>
<td>gw-auth-fail</td>
<td></td>
<td>Gateway authentication failure.</td>
</tr>
<tr>
<td>key-error</td>
<td></td>
<td>The connection attempt failed due to a host key mismatch.</td>
</tr>
<tr>
<td>user-mapping-fail</td>
<td></td>
<td>The connection attempt failed due to a user mapping failure.</td>
</tr>
<tr>
<td>window_title_extracted</td>
<td>boolean</td>
<td>If window titles have been extracted from this graphical session, this value is true, otherwise it is false. The extracted window titles are available in the <code>events object</code> field.</td>
</tr>
<tr>
<td>revision</td>
<td>int</td>
<td>The revision number of the document. A newer document has a larger revision number than an older one. This helps you to determine which session version is newer.</td>
</tr>
<tr>
<td>server</td>
<td>Top level item</td>
<td>The IP address and port number of the remote server.</td>
</tr>
<tr>
<td>trail_downloads</td>
<td>Top level item</td>
<td>Contains a link to the details of the audit-trail downloads in this session (if any).</td>
</tr>
</tbody>
</table>
|                               |                 | "trail_downloads": {
|                               |                 |   "href": "/api/audit/sessions/rUhQZ3jYsY1NDWyp9DEpq/trail_downloads"
<p>|                               |                 | },                                                                          |
| start_time                    | ISO 8601 date   | The timestamp of the start of the connection. Starting with SPS 5 LTS, the timestamp is in ISO 8601 format, for example, 2018-10-11T09:23:38.000+02:00. In earlier versions, it was in UNIX timestamp format. |
| user                          | Top level item  | The details of the user authenticating on the remote server.               |
| id                            | string          | The ID of the user.                                                        |
| name                          | string          | The username used for authenticating against the gateway.                  |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_username</td>
<td>string</td>
<td>The username used for authenticating on the remote server.</td>
</tr>
<tr>
<td>verdict</td>
<td>string</td>
<td>Indicates what SPS decided about the session. A session verdict that originates from log events or other external events.</td>
</tr>
</tbody>
</table>

**Analytics elements**

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| analytics      | Top level element | Contains analytics details of the connection. For example:  

```
"analytics": {
  "interesting_events": [],
  "scripted": false,
  "scripted_results": {},
  "similar_sessions": [],
  "tags": []
},
```
| interesting_events | string | A list of commands and window titles from the session that could be interesting from a security point of view. |
| score.aggregated  | int     | The risk score that SPA assigned to the session. Values range from 0 to 100, with 100 representing the highest risk.          |
| score.details     | object  | This is an object where the keys are algorithm names and values are algorithm-specific details about the score result.             |
| scripted          | boolean | True if the SPA module marked the session as scripted because of non-human activity.                                                       |
| scripted_results  | object  | A key-value pair, where key=<algorithm-name>, value=<reason-of-the-decision>. The algorithm can be clockmaster or gapminder.        

Result: True/False. Reason: Either the reason behind the result, or if no result is available, an error message (for example, the baseline has not been built yet). |
| similar_sessions  | string  | Collection of similar sessions from different sources.                                                                                     |
### Analytics elements

| tags       | string | The Analytics tags section in Search > Details. |

### Audit trail elements

<table>
<thead>
<tr>
<th>archive</th>
<th>Top level element</th>
<th>Indicates whether the audit trail has been archived or not. If the audit trail has not been archived yet, the value of the element is null. For example:</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>ISO 8601 date</td>
<td>The date when the audit trail was archived in ISO 8601 date.</td>
</tr>
<tr>
<td>server</td>
<td>hostname or IP address</td>
<td>The address of the remote server where the audit trail was archived.</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>The path on the remote server where the audit trail was archived.</td>
</tr>
<tr>
<td>policy</td>
<td>string</td>
<td>The ID of the archiving policy that was used to archive the audit trail.</td>
</tr>
<tr>
<td>download</td>
<td>string</td>
<td>The download element allows downloading the audit trail.</td>
</tr>
</tbody>
</table>

### Channel elements

<table>
<thead>
<tr>
<th>key</th>
<th>string</th>
<th>Top level element, contains the ID of the channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Top level element (string)</td>
<td>The properties of the channel.</td>
</tr>
<tr>
<td>Channel elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>If the returned value is true, the session has not ended yet and the channel is active.</td>
</tr>
<tr>
<td>audit_stream_id</td>
<td>string</td>
<td>The identifier of the channel's audit stream. If the session does not have an audit trail, this element is not used.</td>
</tr>
<tr>
<td>channel_id</td>
<td>long</td>
<td>The unique ID of the channel.</td>
</tr>
<tr>
<td>client_x509_subject</td>
<td>string</td>
<td>The client's certificate in Telnet or VNC sessions. Available only if the <code>&lt;Protocol name&gt; Control &gt; Connections &gt; Client-side transport security settings &gt; Peer certificate validation</code> is enabled in SPS.</td>
</tr>
<tr>
<td>duration</td>
<td>int</td>
<td>The duration of the connection. Computed value.</td>
</tr>
<tr>
<td>end_time</td>
<td>ISO 8601 date</td>
<td>The ISO 8601 date of the end of the connection. For ongoing connections, the value is null.</td>
</tr>
<tr>
<td>rule_num</td>
<td>string</td>
<td>The number of the line in the Channel policy applied to the channel.</td>
</tr>
<tr>
<td>start_time</td>
<td>ISO 8601 date</td>
<td>The ISO 8601 date of the start of the connection.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The type of the channel. Additional elements might be used with certain ICA, SSH and RDP channel types.</td>
</tr>
<tr>
<td>verdict</td>
<td>string</td>
<td>The channel's connection verdict. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• accept&lt;br&gt;The connection attempt was successful.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• deny&lt;br&gt;The connection attempt was denied.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• four-eyes-deferred&lt;br&gt;Four-eyes authorization is unable to progress as it is waiting for a remote username.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• four-eyes-error&lt;br&gt;An internal error occurred during four-eyes authorization.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• four-eyes-reject</td>
</tr>
<tr>
<td>Channel elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>command</td>
<td>string</td>
<td>The executed command. Used with the session <code>exec</code> SSH channel type.</td>
</tr>
<tr>
<td>scp_path</td>
<td>string</td>
<td>The folder used for Secure Copy. Used with the session <code>exec scp</code> SSH channel type.</td>
</tr>
<tr>
<td>subsystem_name</td>
<td>string</td>
<td>The name of the used subsystem. Used with the session subsystem <code>sftp</code> SSH channel type.</td>
</tr>
<tr>
<td>originator.ip</td>
<td>string</td>
<td>The source address of the forwarded traffic. Used with the local forward and remote forward SSH channel types.</td>
</tr>
<tr>
<td>originator.name</td>
<td>string</td>
<td>The source host name of the forwarded traffic. If this information is not available, the value is the IP address instead.</td>
</tr>
<tr>
<td>originator.port</td>
<td>int</td>
<td>The source port of the forwarded traffic. Used with the local forward and remote forward SSH channel types.</td>
</tr>
<tr>
<td>connected.ip</td>
<td>string</td>
<td>The target address of the forwarded traffic. Used with the local forward and remote forward SSH channel types.</td>
</tr>
<tr>
<td>connected.name</td>
<td>string</td>
<td>The target host name of the forwarded traffic. If this information is not available, the value is the IP address instead.</td>
</tr>
<tr>
<td>connected.port</td>
<td>int</td>
<td>The target port of the forwarded traffic. Used with the local forward and remote forward SSH channel types.</td>
</tr>
<tr>
<td>dynamic_channel</td>
<td>string</td>
<td>The name of the dynamic channel. Used with the dynamic virtual RDP channel type.</td>
</tr>
</tbody>
</table>
| device_name      | string | Used with the serial redirect, parallel redirect, printer redirect, disk redirect,
<table>
<thead>
<tr>
<th>Channel elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>and scard redirect RDP channel types.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The name of the device.</td>
</tr>
<tr>
<td>application</td>
<td>string</td>
<td>Used with ICA connections.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The name of the application accessed in a seamless Citrix ICA connection.</td>
</tr>
<tr>
<td>four_eyes_authorizer</td>
<td>string</td>
<td>The username of the user who authorized the session.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available only if four-eyes authorization is required for the channel.</td>
</tr>
<tr>
<td>four_eyes_description</td>
<td>string</td>
<td>The description of the session submitted by the authorizer of the session.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available only if four-eyes authorization is required for the channel.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Client elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>client</td>
<td>Top level element</td>
<td>The IP address and port number of the client. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;client&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ip&quot;: &quot;10.20.30.40&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;port&quot;: 59125</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>ip</td>
<td>string</td>
<td>The IP address of the client.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The host name of the client. If this information is not available, the value is the IP address instead.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number of the client.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Server elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>Top level element</td>
<td>The IP address and port number of the remote server. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;server&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ip&quot;: &quot;10.20.30.40&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;port&quot;: 55386</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>ip</td>
<td>string</td>
<td>The IP address of the remote server.</td>
</tr>
<tr>
<td><strong>Server elements</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>---------------------</td>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The host name of the remote server. If this information is not available, the value is the IP address instead.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number of the remote server.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Server local elements</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>server_local</td>
<td>Top level element</td>
<td>The IP address and port number of SPS. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;server_local&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ip&quot;: &quot;10.20.30.40&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;port&quot;: 55386</td>
</tr>
<tr>
<td>ip</td>
<td>string</td>
<td>The IP address of SPS.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The host name of SPS. If this information is not available, the value is the IP address instead.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number of SPS.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Target elements</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>Top level element</td>
<td>The IP address and port number the client targeted for connection. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;target&quot;:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ip&quot;: &quot;10.20.30.40&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;port&quot;: 221</td>
</tr>
<tr>
<td>ip</td>
<td>string</td>
<td>The IP address the client targeted for connection.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The host name of the client targeted for connection. If this information is not available, the value is the IP address instead.</td>
</tr>
<tr>
<td>port</td>
<td>int</td>
<td>The port number the client targeted for connection.</td>
</tr>
</tbody>
</table>

**Examples:**

All possible SSH channel types:
"channels": [
  {
    "key": "1",
    "meta": {
      "href": "/api/audit/sessions/1/channels/1"
    },
    "body": {
      "type": "session shell",
      "verdict": "accept",
      "start_time": 1451901988,
      "end_time": 1451902145,
      "duration": 157
    }
  },
  {
    "key": "2",
    "meta": {
      "href": "/api/audit/sessions/1/channels/2"
    },
    "body": {
      "type": "session exec",
      "verdict": "accept",
      "start_time": 1451902141,
      "end_time": 1451902145,
      "duration": 4,
      "command": "ls"
    }
  },
  {
    "key": "3",
    "meta": {
      "href": "/api/audit/sessions/1/channels/3"
    },
    "body": {
      "type": "session exec scp",
      "verdict": "accept",
      "start_time": 1451902141,
      "end_time": 1451902145,
      "duration": 4,
      "scp_path": "<path-to-folder>"
    }
  },
  {
    "key": "4",
    "meta": {
      "href": "/api/audit/sessions/1/channels/4"
    },
    "body": {
      "type": "session exec scp",
      "verdict": "accept",
      "start_time": 1451902141,
      "end_time": 1451902145,
      "duration": 4,
      "scp_path": "<path-to-folder>"
    }
  }
]
"type": "session subsystem sftp",
"verdict": "accept",
"start_time": 1451902142,
"end_time": 1451902145,
"duration": 3,
"subsystem_name": "sftp"
},

{
"key": "5",
"meta": {
  "href": "/api/audit/sessions/1/channels/5"
},
"body": {
  "type": "local forward",
  "verdict": "accept",
  "start_time": 1451902145,
  "end_time": 1451902146,
  "duration": 1,
  "originator.address": "::1",
  "originator.port": 59578,
  "connected.address": "<server>",
  "connected.port": 22
}
},

{
"key": "6",
"meta": {
  "href": "/api/audit/sessions/1/channels/6"
},
"body": {
  "type": "remote forward",
  "verdict": "accept",
  "start_time": 1451902145,
  "end_time": 1451902146,
  "duration": 1,
  "originator.address": "::1",
  "originator.port": 42212,
  "connected.address": "localhost",
  "connected.port": 9898
}
},

{
"key": "7",
"meta": {
  "href": "/api/audit/sessions/1/channels/7"
},
"body": {
  "type": "remote forward",
  "verdict": "accept",
  "start_time": 1451902145,
  "end_time": 1451902146,
  "duration": 1,
  "originator.address": "::1",
  "originator.port": 59578,
  "connected.address": "<server>",
  "connected.port": 22
}
"type": "x11 forward",
"verdict": "deny",
"start_time": 1451902149,
"end_time": 1451902149,
"duration": 0
}
}
]

All possible RDP channel types:

"channels": [  
{
   "key": "1",
   "meta": {
      "href": "/api/audit/sessions/1/channels/1"
   },
   "body": {
      "type": "drawing",
      "verdict": "accept",
      "start_time": 1451901988,
      "end_time": 1451902145,
      "duration": 157
   }
  },  
{
   "key": "2",
   "meta": {
      "href": "/api/audit/sessions/1/channels/2"
   },
   "body": {
      "type": "sound",
      "verdict": "accept",
      "start_time": 1451902141,
      "end_time": 1451902145,
      "duration": 4
   }
  },  
{
   "key": "3",
   "meta": {
      "href": "/api/audit/sessions/1/channels/3"
   },
   "body": {
      "type": "clipboard",
      "verdict": "accept",
      "start_time": 1451902141,
      "end_time": 1451902145,
      "duration": 4
   }
}
"type": "serial redirect",
"verdict": "accept",
"start_time": 1451902149,
"end_time": 1451902150,
"duration": 1,
"device_name": "COM1"
},
{
"key": "8",
"meta": {
"href": "/api/audit/sessions/1/channels/8"
},
"body": {
"type": "parallel redirect",
"verdict": "accept",
"start_time": 1451902149,
"end_time": 1451902150,
"duration": 1,
"device_name": "LPT1"
}
},
{
"key": "9",
"meta": {
"href": "/api/audit/sessions/1/channels/9"
},
"body": {
"type": "printer redirect",
"verdict": "accept",
"start_time": 1451902149,
"end_time": 1451902150,
"duration": 1,
"device_name": "PRN22"
}
},
{
"key": "10",
"meta": {
"href": "/api/audit/sessions/1/channels/10"
},
"body": {
"type": "disk redirect",
"verdict": "accept",
"start_time": 1451902149,
"end_time": 1451902150,
"duration": 1,
"device_name": "J:"
}
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Download audit trails

You can download the audit trail of a session from the /api/audit/sessions/<session-id>/audit_trail endpoint. To find a specific audit trail, see Searching in the session database on page 717. You can download audit trails that are available on SPS, and also audit trails that have been archived (if SPS can access the archived audit trail).
To actually create a file, you must save the downloaded data into a file (use the .zat file extension), for example:

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/audit_trail" > my-downloaded-trail.zat
```

You can replay the downloaded audit trails with the Safeguard Desktop Player application. For details, see Safeguard Desktop Player User Guide.

If you want to replay an ongoing session in follow mode, you have to download the audit trail in .srs format. Use the ?format=srs option:

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/audit_trail?format=srs" > my-downloaded-trail.srs
```

For details, see "Replay audit files in follow mode" in the Safeguard Desktop Player User Guide.

### Searching in the session database

You can list, search, and filter the SPS session database at the /api/audit/sessions endpoint. You can use the following actions:

- **?start**
  Display sessions that started after the specified date. Use the ISO 8601 format for the date, for example, `2017-01-25T10:00`.
- **?q**
  Filter the list using one or more property (element) of the sessions.
- **?content**
  Search in the content of indexed sessions.
- **?end**
  Display sessions that ended before the specified date. Use the ISO 8601 format for the date, for example, `2017-01-25T10:00`.
- **?fields**
  Display the selected properties (elements and values) of the listed sessions.
- **?limit**
  Configure the pagination of the displayed results using the ?offset and ?limit parameters.
The `?limit` parameter allows you to configure the maximum number of results to display on a page at once.

The default value of `?limit` is 500.

**NOTE:** The default value of 500 is the maximum permitted value you can set for `?limit`. If you set the `?limit` parameter to a value bigger than 500, only the first 500 results will be displayed.

- **?offset**
  Configure the pagination of the displayed results using the `?offset` and `?limit` parameters.

  The `?offset` parameter allows you to configure the offset from the first result that is displayed. This can be useful if the number of items returned exceeds the number of items displayed on the first page, and you want to navigate to any of the subsequent items displayed on other pages.

  The default value of `?offset` is null.

  **NOTE:** The maximum number of search results in One Identity Safeguard for Privileged Sessions is 10000. As a result, any `?offset` values set to larger than 10000 will be ignored and the results exceeding the value of 10000 will not be displayed.

- **?sort**
  Sort the results based on the values of the fields.

- **?format**
  Configure the format of the displayed results.

  The default value of `?format` is `json`. If you do not configure the `?format` parameter, the results will be displayed in JSON format.

  To display search results in a CSV format, enter `csv` as a value.

To combine multiple expressions, use the & (ampersand) character, for example:

**Display the target server and port of each active session:**

```bash
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?fields=psm.target.address,psm.target.port&q=active:true"
```

**Display 10 sessions at once, and navigate to 31-40:**

```bash
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?limit=10&offset=31"
```

**Search in metadata and session content at the same time:**

```bash
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?q=protocol:ssh&content=sudo"
```

**NOTE:** If you use curl, use quotation marks for the URL to avoid problems with the & (ampersand) character.
Response

The response to search or filtering action contains a list of the matching sessions, as well as some additional meta fields. For example:

```
{
    "items": [
        {
            "body": {
                "duration": 0,
                "name": "myname",
                "start_time": "2017-01-25T11:52.000+01:00"
            },
            "key": "2",
            "meta": {
                "href": "/api/audit/sessions/2"
            }
        },
        {
            "body": {
                "duration": 34,
                "name": "myname",
                "start_time": "2017-01-25T11:11.000+01:00"
            },
            "key": "10",
            "meta": {
                "href": "/api/audit/sessions/10"
            }
        }
    ],
    "meta": {
        "fields": [
            "start_time",
            "name",
            "duration"
        ],
        "first": "/api/audit/sessions?limit=500&offset=0&fields=start_time,name,duration&q=name%3Amyname&duration",
        "href": "/api/audit/sessions",
        "last": "/api/audit/sessions?limit=500&offset=0&fields=start_time,name,duration&q=name%3Amyname&sort=duration",
        "limit": 500,
        "match_count": 2,
        "next": null,
        "offset": 0,
        "parent": "/api/audit",
        "previous": null
    }
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>list</td>
<td>Top level element, a list containing the details of the matching sessions.</td>
</tr>
<tr>
<td>body</td>
<td>JSON object</td>
<td>Contains the information returned about a session, that is, the fields selected with the ?fields expression. For example, if you used the fields=start_time,psm.gateway_username,duration expression in your query, then the body element contains these fields for each returned session:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;body&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;duration&quot;: 0,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;name&quot;: null,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;start_time&quot;: &quot;2017-01-25T11:11:52.000+01:00&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>A globally unique string that identifies the session. This session ID has the following format: svc/&lt;unique-random-hash&gt;/name-of-the-connection-policy:&lt;session-number-since-service-started&gt;/protocol, for example, svc/5tmEaM7xdNi1oscgVWpbZx/ssh_console:1/ssh. Log messages related to the session also contain this ID. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2015-03-20T14:29:15+01:00 demo.example zorp/scb_ssh[5594]: scb.audit(4): (svc/5tmEaM7xdNi1oscgVWpbZx/ssh_console:0/ssh): Closing connection; connection='ssh_console', protocol='ssh', connection_id='409829754550c1c7a27e7d', src_ip='10.40.0.28', src_port='39183', server_ip='10.10.20.35', server_port='22', gateway_username='', remote_username='example-username', verdict='ZV_ACCEPT'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that when using the session ID in a REST call, you must replace the special characters in the ID with the hyphen (-) character. For example, if the session ID in the log message is svc/fNLgRmAYf5EtycgUyNkC1B/ssh_demo2:2, use the svc-fNLgRmAYf5EtycgUyNkC1B-ssh_demo2-2 ID in REST calls.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In addition to the usual meta elements of other endpoints, search results can contain the following additional elements.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>meta</td>
<td>JSON</td>
<td>Top level element, a list containing meta information about the response.</td>
</tr>
</tbody>
</table>
| fields    | list   | Contains the list of data fields returned about each session, that is, the fields selected with the ?fields expression. For example, if you used the fields=start_time, psm.gateway_username, duration expression in your query, then the body element contains these fields for each returned session:  
```
"fields": [
  "start_time",
  "name",
  "duration"
],
```
| limit     | integer| The maximum number of sessions returned in a the response (by default, 500).  |
| match_count| integer| The number of results matching the query.                                  |
| next      | string | A query to retrieve the next set of search results, if match_count is higher than limit. |
| offset    | integer| Indicates the position of the results in this response, relative to the total number of results (match_count). Otherwise, its value is null. |
| previous  | string | A query to retrieve the previous set of search results, if match_count is higher than limit, and offset is higher than 0. Otherwise, its value is null. |

**Filtering**

You can use the ?q option to filter the list using one or more property (element) of the sessions.

```
?q=protocol:ssh
```

You can escape special characters using the backslash character.

```
?q=server_username:"Windows User"
```

To add multiple elements to the filter, you can use the AND, AND NOT, and OR operators.

```
?q=protocol:ssh AND verdict:accept AND NOT name:admin
```
You can create groups using () (parentheses).

?q=(client.address:10.20.30.40 OR target.address:10.20.30.40) AND verdict:accept

You can also use () (parentheses) to add multiple possible values for a property.

?q=protocol:(ssh rdp)

You can use the * (asterisk) and ? (question mark) wildcards for string-type values.

?q=name:?dmi*

You can define ranges using [] (brackets) or {} (braces) and the TO operator. This only works for numeric (int) values.

- [ means equal or higher than the following value
- ] means equal or lower than the preceding value
- { means higher than the following value
- } means lower than the preceding value

For example, the following range resolves to 22:

?q=port:{21 TO 23}

You can also use the * (asterisk) wildcard in the range.

?q=start_time:[* TO 1461654799]

Note that not all connection data can be used for filtering. The available elements are:

- active
  Boolean, true means the session is ongoing (it is still active).
- auth_method
  String, the authentication method used.
- channel_policy
  String, the key of the channel policy.
- client.address
  String, the IP address of the client.
- client.port
  Integer, the port of the client.
- psm.connection_policy
  String, the key of the connection policy.
- end_time
The date of the end of the session in ISO 8601 format.

- **name**
  String, the username used for authenticating against the gateway.

- **protocol**
  String, the protocol of the session.

- **server.address**
  String, the IP of the remote server.

- **psm.server_local.address**
  String, the IP of SPS.

- **psm.server_local.port**
  String, the port of SPS.

- **server.port**
  String, the port of the remote server.

- **server_username**
  String, the username used for authenticating on the remote server.

- **session_id**
  String, the identifier of the session.

- **start_time**
  The date of the start of the session in ISO 8601 format.

- **target.address**
  String, the IP the client targeted in the session.

- **target.port**
  Integer, the port the client targeted in the session.

- **verdict**
  String, the connection verdict. Possible values are:
  - **accept**
    The connection attempt was successful.
  - **accept-terminated**
    The connection violated a content policy, and was terminated by SPS.
  - **auth-fail**
    Authentication failure.
  - **deny**
    The connection was denied.
  - **fail**
    The connection attempt failed.
• gw-auth-fail
  Gateway authentication failure.
• key-error
  The connection attempt failed due to a host key mismatch.
• user-mapping-fail
  The connection attempt failed due to a user mapping failure.

**Content search in indexed audit trails**

You can use the `?content` option to search for keywords that appear in the content of the audit trails. Such content is any text that appeared on the screen in terminal or graphical sessions, or commands that the user typed in terminal sessions. Note that content search works only if:

- Indexing was enabled in the connection policy related to the audit trail during the session, and
- the audit trail has already been indexed.

`?content="my-search-expression"`

You can use the [Apache Lucene query syntax](https://lucene.apache.org/lucene-core/core-FAQ.html) to create the search expression, but note the following points.

- You must format the search expression as an URL, and escape special characters accordingly. For example, if your search expression is `man iptables`, you must escape the whitespace: `man%20iptables`
  
  For a list of special (reserved) URL characters, see [RFC3986](https://tools.ietf.org/html/rfc3986).
- Do not begin the expression with the * wildcard.

**Examples:**

Search for the word example

`?content=example`

Search for the words example, examples, and so on:

`?content=example%3F`

Search for the words example, examine, and so on:

`?content=exam%2A`
Search in metadata and session content at the same time:

```bash
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?q=protocol:ssh&content=sudo"
```

For further details and examples, see "Searching in the contents of audit trails" in the Administration Guide.

**Displaying session data**

You can use the `?fields` option to display the selected data (body elements) of each session.

```bash
?fields=protocol
```

To list multiple elements, use the `,` (comma) character. Note that the response includes the selected fields in alphabetic order, not in the order they were specified.

```bash
?fields=protocol,name
```

To list all possible elements, use the `fields=*` expression.

```bash
?fields=* 
```

Note that not all connection data can be displayed in the generated list. The available elements are:

- **active**
  
  Boolean, true means the connection is ongoing.

- **archived**
  
  Boolean, true means the session has been archived.

- **auth_method**
  
  String, the authentication method used.

- **channel_policy**
  
  String, the key of the channel policy.

- **client.address**
  
  String, the IP address of the client.

- **client.port**
  
  Integer, the port of the client.

- **connection_policy**
String, the key of the connection policy.

- duration
  Integer, the duration of the session. Computed value.

- end_time
  The date of the end of the session in ISO 8601 format.

- name
  String, the username used for authenticating against the gateway.

- protocol
  String, the protocol of the session.

- server.address
  String, the IP of the remote server.

- server_local.address
  String, the IP of SPS.

- server_local.port
  Integer, the port of SPS.

- server.port
  Integer, the port of the remote server.

- server_username
  String, the username used for authenticating on the remote server.

- session_id
  String, the identifier of the session.

- start_time
  The date of the start of the session in ISO 8601 format.

- target.address
  String, the IP the client targeted in the session.

- target.port
  Integer, the port the client targeted in the session.

**Date-specific search**

To display search results only for specific date intervals, you can use the ?start and ?end options.

- The ?start option selects the sessions that started after the specified date (based on the value of the start_time field).
- The ?end option selects the sessions that ended before the specified date (based on the value of the end_time field).
- Both options accept the date in ISO 8601 format.
### Examples:

Select sessions that started on January 20, 2017, or later:

```
?start=2017-01-20
```

Select sessions that started on 11:00 January 20, 2017, or later:

```
?start=2017-01-20T11:00
```

Select sessions that ended on January 20, 2017:

```
?end=2017-01-20
```

Select sessions started and ended on January 20, 2017:

```
?start=2017-01-20&end=2017-01-20
```

Select sessions started after 11:00, January 20, 2017, and ended before 09:00, January 21, 2017:

```
?start=2017-01-20T11:00&end=2017-01-21T09:00
```

### Changing the display limit

You can use the ?limit option to change the number of items displayed at once. The default limit is 500.

```
?limit=1000
```

To navigate beyond the displayed set, use the offset option.

### Navigating large datasets

You can use the ?offset option to navigate data sets that extend beyond the display limit. The default value of the offset is 0, this is the initially displayed set. To move to other items beyond the initial set, increase the value to a number that corresponds to the item where you want to start displaying results from.

Example: the display limit is the default 500, and the number of sessions is 1012. The initial 500 sessions are listed at:
To view sessions from 501 to 1000, change the offset to 501:

?offset=501

To display the remaining 12 sessions, change the offset to 1001:

?offset=1001

Sort the results

You can sort the search results using the sort expression, for example, based on the length of the sessions:

?sort=duration

You can use any field to sort the results. By default, sorting returns the results in ascending order, if you use ?sort=duration, then the shortest session is at the beginning of the list. To sort the results in descending order, add the minus sign (-) before the field name. For example, the response to the following expression starts with the longest session:

?sort=-duration

You can specify multiple fields to order the list. In this case, the list is first ordered using the first field, then the second, and so on. For example, to order the list first by duration, then by start time, use the following expression.

?sort=duration,start_time

The following example sorts the results by duration, and displays the start time, gateway username, and duration fields.

curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?sort=duration&fields=start_time,psm.gateway_username,duration"

Configure the format of the displayed results

The default value of ?format is json. If you do not configure the ?format parameter, the results will be displayed in JSON format.

?format=json

To display search results in a CSV format, enter csv as a value.

?format=csv
Example: querying sessions in CSV result format

Given that the following sessions were recorded:

```json
{
  "1": {
    "channel": [
      {"channel_id": 1},
      {"channel_id": 2}
    ],
    "recording": {
      "session_id": 1,
      "archived": false,
      "channel_policy": "policy1",
      "content_reference_id": 1,
      "connection_policy": "connection1",
      "auth_method": "password",
      "target": {
        "port": 2222,
        "ip": "1.1.1.1",
        "name": "1.1.1.1"
      },
      "server_local": {
        "port": 46,
        "ip": "1.1.1.1",
        "name": "1.1.1.1"
      }
    },
    "user": {
      "server_username": "user1",
      "gateway_username": "user1"
    },
    "client": {
      "port": 48679,
      "ip": "2.2.2.2",
      "name": "2.2.2.2"
    },
    "active": false,
    "start_time": 1,
    "duration": 4,
    "server": {
      "port": 22,
      "ip": "2.2.2.2",
      "name": "2.2.2.2"
    },
    "end_time": 5,
  }
}``

SPS 6.13.0 REST API Reference Guide
Search, download, and index sessions
"protocol": "ssh",
"2": {
  "channel": [
    {
      "channel_id": 3
    },
    {
      "channel_id": 4
    }
  ],
  "recording": {
    "session_id": 2,
    "archived": false,
    "channel_policy": "policy2",
    "content_reference_id": 2,
    "connection_policy": "connection2",
    "auth_method": "password",
    "target": {
      "port": 2222,
      "ip": "1.1.1.1",
      "name": "1.1.1.1"
    },
    "server_local": {
      "port": 46,
      "ip": "1.1.1.1",
      "name": "1.1.1.1"
    }
  },
  "user": {
    "server_username": "user2",
    "gateway_username": "user2"
  },
  "client": {
    "port": 48680,
    "ip": "3.3.3.3",
    "name": "3.3.3.3"
  },
  "active": false,
  "start_time": 1,
  "duration": 4,
  "server": {
    "port": 24,
    "ip": "2.2.2.2",
    "name": "2.2.2.2"
  },
  "end_time": 7,
  "protocol": "ssh"}
}
When the query is the following:

```bash
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?format=csv&fields=protocol,end_time,user.gateway_username,server.ip,client.ip,client.port"
```

The response is the following:

```
"Key","Protocol","End time","Gateway username","Server IP","Client IP","Client port"
"2","ssh","7","user2","2.2.2.2","3.3.3.3","48680"
"1","ssh","5","user1","2.2.2.2","2.2.2.2","48679"
```

Example: querying sessions in CSV result format with interesting events

Given that the following sessions were recorded:

```json
{
  "1":{
    "origin": "RECORDING",
    "protocol": "SSH",
    "analytics": {
      "interesting_events": ["ssh", "sudo"],
      "similar_sessions": []
    },
    "recording": {
      "session_id": "1",
      "verdict": "ACCEPT",
      "audit_trail": "/var/lib/zorp/audit/532078660569910c6542b2/01/audit-scb_ssh-1451900800-1.zat",
      "connection_policy": "ssh1",
      "content_reference_id": 1
    }
  },
  "2":{
    "origin": "RECORDING",
    "protocol": "SSH",
```
When the query is the following:

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?sort=recording.session_id&format=csv&fields=recording.session_id,analytics.interesting_events,analytics.similar_sessions"
```

The response is the following:

```
"Key","Recording Session ID","Analytics Interesting events","Similar Sessions"
"1","1","ssh",""
"1","1","sudo",""
"2","2","sudo",""
"2","2","systemctl",""
```

Example: querying sessions in CSV result format with audit trail link

Given that the following sessions were recorded:

```
{
    "svc-paKzcMjwXghEFJ9UvvdqFU-sid-1": {
        "origin": "RECORDING",
        "protocol": "SSH",
        "recording": {
            "session_id": "1",
            "verdict": "ACCEPT",
```
"audit_trail": "/var/lib/zorp/audit/532078660569910c6542b2/01/audit-scb_ssh-1451900800-1.zat",
"connection_policy": "ssh1",
"content_reference_id": 1
}
},
"svc-paKzcMJwXghEFJ9UvsdqFU-sid-2": {
"origin": "RECORDING",
"protocol": "SSH",
"recording": {
"session_id": "2",
"verdict": "ACCEPT",
"connection_policy": "ssh2",
"content_reference_id": 2
}
}
}

When the query is the following:

```
   curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions?format=csv&fields=trail_download_link"
```

The response is the following:

```
   "Key","Audit trail download link"
   "svc-paKzcMJwXghEFJ9UvsdqFU-sid-2",""
   "svc-paKzcMJwXghEFJ9UvsdqFU-sid-1","https://127.0.0.1/api/audit/sessions/svc-paKzcMJwXghEFJ9UvsdqFU-sid-1/audit_trail"
```

### Searching in connection content

You can search in the contents of individual connections at the `api/audit/sessions/<session-id>/content/?q=<my-search-expression>` endpoint.

**URL**

GET `https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/content/?q=<my-search-expression>`
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <strong>Authenticate to the SPS REST API on page 19.</strong></td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command retrieves those events in the contents of a specific connection that match the search expression(s).

```
curl --cookie cookies https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/content/?q=<my-search-expression>
```

**NOTE:** Make sure that you use the ?q option and that when you use it, you do not leave it empty. Not using the ?q option or an empty ?q will result in an empty "items" list returned in the response.

You can use the **Apache Lucene query syntax** to create the search expression, but note the following points.

- You must format the search expression as a URL, and escape special characters accordingly. For example, if your search expression is `man iptables`, you must escape the whitespace: `man%20iptables`
- Do not begin the expression with the * wildcard.

Response

The response contains a list of those events in the contents of the connection that match the search expression(s). The response also contains some meta fields.

If you specified a search expression using the ?q option and the response returns an empty "items" list, that can indicate that:

- The search returned no results.
- There is no content recorded for the connection.

The following is an example response:
```json
{
  "items": [
    {
      "channel.id": 5,
      "end_time": "2017-08-14T10:35:43.957000",
      "rank": 2.4756217002868652,
      "record_id": {
        "begin": 158,
        "end": 160,
        "for_screenshot": 158
      },
      "start_time": "2017-08-14T10:35:19.098000",
      "trail_id": "12"
    }
  ],
  "meta": {
    "href": "/api/audit/sessions/2a620c1cfeb39c537a5e80280283d741/content",
    "parent": "/api/audit/sessions/2a620c1cfeb39c537a5e80280283d741",
    "remaining_seconds": 599
  }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>list</td>
<td>Top-level element, a list containing the details of the matching session.</td>
</tr>
<tr>
<td>channel.id</td>
<td>integer</td>
<td>A reference to the ID of the channel in the session where the event occurred.</td>
</tr>
<tr>
<td>end_time</td>
<td>string</td>
<td>The timestamp of when the content disappeared from the screen. Starting with SPS 5 LTS, the timestamp is in ISO 8601 format, for example, 2018-10-11T09:23:38.000+02:00. In earlier versions, it was in UNIX timestamp format.</td>
</tr>
<tr>
<td>rank</td>
<td>float</td>
<td>Indicates the relevance of the match. If there are several results, the order of them is based on their relevance.</td>
</tr>
<tr>
<td>record_id</td>
<td>integer</td>
<td>The content element's exact position in the audit trail file.</td>
</tr>
<tr>
<td>begin</td>
<td>integer</td>
<td>The identifier of the screenshot in the</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>end</td>
<td>integer</td>
<td>The identifier of the screenshot in the audit trail file where the content element last appeared.</td>
</tr>
<tr>
<td>for_screenshot</td>
<td>integer</td>
<td>The identifier of the most relevant screenshot in the audit trail file. This is the screenshot on which the event in question is the most clearly visible. For details on how to generate and retrieve the screenshot, see Generate and retrieve screenshot for content search.</td>
</tr>
<tr>
<td>start_time</td>
<td>string</td>
<td>The timestamp of when the content first appeared on the screen and recording started.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Starting with SPS 5 LTS, the timestamp is in ISO 8601 format, for example, 2018-10-11T09:23:38.000+02:00. In earlier versions, it was in UNIX timestamp format.</td>
</tr>
<tr>
<td>trail_id</td>
<td>integer</td>
<td>The unique identifier of the trail that contains the event.</td>
</tr>
</tbody>
</table>

In addition, search results can contain the usual meta elements of other endpoints:

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meta</td>
<td>JSON object</td>
<td>Top-level element, a list containing meta information about the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For details about the type of information returned, see Message format on page 10.</td>
</tr>
</tbody>
</table>

### Generate and retrieve screenshot for content search

To generate and download screenshots for a specific content search result, complete the following steps. For details on searching in the content of a session, see Searching in connection content.
1. **Perform a content search in a session.**

   Use a GET request on the endpoint of a specific session, for example:

   ```
   GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/content/?q=<my-search-expression>
   ```

   For details, see *Searching in connection content*. If there are search results for the search keywords in the session, the response includes a `record_id` block, for example:

   ```json
   "record_id": {
      "begin": 158,
      "end": 160,
      "for_screenshot": 158
   },
   ```

2. **Generate a screenshot for the search result.**

   Note the value of the `for_screenshot` key in the search response, and use it to generate a screenshot for that particular record_id. POST the value of the `for_screenshot` key to the `https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/_generate?record_ids=<value-of-for_screenshot>` endpoint.

3. **Download the screenshot.**

   To download the screenshot in PNG format, GET the value of the `for_screenshot` key to the `https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/screenshots/<value-of-for_screenshot>` endpoint.

**Session statistics**

The `api/audit/sessions/stats` endpoint provides statistics about recorded sessions (active and closed).

**URL**

```
GET https://<IP-address-of-SPS>/api/audit/sessions/stats?field=<field-name>
```
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following request retrieves statistical data about sessions.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/audit/sessions/stats?field=<field-name>
```

Request parameters

Use the following parameters to fine-tune your request for statistics:

- `?q`: Narrow down the scope of statistics using one or more properties (elements) of the sessions.
- `?field`: Request statistics for the selected properties (elements and values) of sessions (for example, protocol).
  
  Using this parameter is mandatory.
- `?sub_fields`: Request sub statistics for the selected properties (elements and values) of sessions (for example, protocol).
  
  This parameter only accepts a single parameter. If more than one parameter is listed, only the first will be considered.
- `?size`: Limit the range of values displayed in the statistics for a given field. Statistics will be shown only for the top size number of most frequently occurring values (that is, values with the highest number of counts).

Take the following example. If you query `/api/audit/sessions/stats?field=protocol&size=2`, and the following sessions were recorded:
The response contains:

```json
...
{  "Alpha": {    "protocol": "http"  },  "Bravo": {    "protocol": "ssh"  },  "Charlie": {    "protocol": "rdp"  },  "Delta": {    "protocol": "rdp"  },  "Echo": {    "protocol": "rdp"  },  "Foxtrot": {    "protocol": "http"  },  "Golf": {    "protocol": "http"  } }
...
```

And the response items look like the snippet below. That is, in this example, there will be no statistics for "protocol": "ssh". The top 2 values are "rdp" and "http", with a count of 3 each. "ssh" occurred only once, so it did not make it to the top 2 most frequent values.
Statistics are returned for sessions that started after the specified date. Use the ISO 8601 format for the date, for example, 2017-01-25T10:00.

**?end:** Statistics are returned for sessions that ended before the specified date. Use the ISO 8601 format for the date, for example, 2017-01-25T11:00.

**?content:** Statistics are returned for indexed sessions that contain the type of content specified.

**NOTE:** When performing a content query, the maximum number of results returned is 3000. When this limit is exceeded, the scope of statistics is limited to the first 3000 sessions (even if there are more than 3000 sessions that match your criteria).

### Response

The following snippet is a sample response received when retrieving statistics about the protocol field.

For details of the *meta* object, see Message format on page 10.

Those fields of the *meta* object that are specific to statistics are collected in table **Element**.

```json
{
   "items": [
   {
      "count": 7,
      "value": "ssh"
   }
   ],
   "meta": {
      "field": "protocol",
      "href": "/api/audit/sessions/stats",
      "others": 0,
      "parent": "/api/audit/sessions",
      "remaining_seconds": 600,
      "size": 10
   }
}
```

<table>
<thead>
<tr>
<th><strong>Element</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>body, or items when a</td>
<td>Top-level element (string)</td>
<td>Contains the properties that are in the scope of the requested statistics.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>list is returned</td>
<td></td>
<td></td>
</tr>
<tr>
<td>count</td>
<td>integer</td>
<td>Indicates the number of sessions included in the scope of statistics.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Contains the value of the field that you requested statistics about.</td>
</tr>
<tr>
<td>meta</td>
<td>Top-level element</td>
<td>Contains links to different parts of the REST service.</td>
</tr>
<tr>
<td>field</td>
<td>string</td>
<td>Contains the name of the field that you requested statistics about.</td>
</tr>
<tr>
<td>sub_fields</td>
<td>string</td>
<td>Contains the name of the sub field that you requested statistics about.</td>
</tr>
<tr>
<td>others</td>
<td>integer</td>
<td>Some values of the field that you specified in your query are not included in the scope of statistics. This happens when a specific value occurs fewer times in the examined sessions than the aggregation size. The others field indicates the number of those distinct values that are not included in the statistics. For a detailed explanation with an example, see ?size.</td>
</tr>
<tr>
<td>size</td>
<td>integer</td>
<td>The size that you specified in your query.</td>
</tr>
</tbody>
</table>

**Example 1:**

If you query "/api/audit/sessions/stats?field=protocol", and the following sessions were recorded:

```json
...
 {
   "Alpha": {
     "protocol": "ssh"
   },
   "Bravo": {
     "protocol": "ssh"
   },
   "Charlie": {
     "protocol": "rdp"
   },
```
"Delta": {
    "protocol": "rdp"
},
"Echo": {
    "protocol": "rdp"
},
"Foxtrot": {
    "protocol": "ssh"
},
"Golf": {
    "protocol": "ssh"
}

The response contains:

...

{  
    "meta": {
        "href": "/api/audit/sessions/stats",
        "parent": "/api/audit/sessions",
        "others": 0,
        "field": "protocol"
    }
}

The response items contain:

...

[  
    {"count": 4, "value": "ssh"},
    {"count": 3, "value": "rdp"}
...

Example 2:

If you query
"/api/audit/sessions/stats?field=protocol&content=login&start=2017-01-
and the following sessions were recorded:

```json
{
  "Alpha": {
    "protocol": "ssh",
    "start_time": "2017-01-01",
    "end_time": "2017-01-02",
    "recording": {
      "content_reference_id": 1
    }
  },
  "Bravo": {
    "protocol": "ssh",
    "start_time": "2017-01-01",
    "end_time": "2017-01-02",
    "recording": {
      "content_reference_id": 2
    }
  },
  "Charlie": {
    "protocol": "rdp",
    "start_time": "2017-01-01",
    "end_time": "2017-01-02",
    "recording": {
      "content_reference_id": 3
    }
  },
  "Delta": {
    "protocol": "rdp",
    "start_time": "2017-01-03",
    "end_time": "2017-01-04",
    "psm": {
      "content_reference_id": 4
    }
  },
  "Echo": {
    "protocol": "rdp",
    "start_time": "2017-01-03",
    "end_time": "2017-01-04",
    "recording": {
      "content_reference_id": 5
    }
  },
  "Foxtrot": {
```
```
"protocol": "ssh",
"start_time": "2017-01-04",
"end_time": "2017-01-06",
"recording": {
  "content_reference_id": 6
},
"Golf": {
  "protocol": "ssh",
  "start_time": "2017-01-02",
  "end_time": "2017-01-10",
  "recording": {
    "content_reference_id": 7
  }
}
}

And the following sessions match when running the content query:

<table>
<thead>
<tr>
<th>trail_id</th>
<th>rank</th>
<th>hits_count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.555</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1.555</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1.555</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1.555</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1.555</td>
<td>1</td>
</tr>
</tbody>
</table>

The response contains:

...
{
  "meta": {
    "href": "/api/audit/sessions/stats",
    "parent": "/api/audit/sessions",
    "others": 0,
    "field": "protocol"
  }
}
...

The response items contain:

...
[
  {"count": 2, "value": "rdp"}
...

SPS 6.13.0 REST API Reference Guide
Search, download, and index sessions
ONE IDENTITY by Quest
Example 3:

If you query "/api/audit/sessions/stats?field=user.gateway_username&?sub_fields=protocol&?size=1", and the following sessions were recorded:

```json
...
  "Alpha": {
    "protocol": "ssh",
    "user": {
      "gateway_username": "user-Alpha"
    }
  },
  "Bravo": {
    "protocol": "ssh",
    "user": {
      "gateway_username": "user-Bravo"
    }
  },
  "Charlie": {
    "protocol": "rdp",
    "user": {
      "gateway_username": "user-Charlie"
    }
  },
  "Delta": {
    "protocol": "rdp",
    "user": {
      "gateway_username": "user-Alpha"
    }
  },
  "Echo": {
    "protocol": "rdp",
    "user": {
      "gateway_username": "user-Alpha"
    }
  },
  "Foxtrot": {
    "protocol": "ssh",
    "user": {
      "gateway_username": "user-Alpha"
    }
  },
  "Golf": {
```
"protocol": "ssh",
"user": {
  "gateway_username": "user-Alpha"
},
"Hotel": {
  "protocol": "ssh",
  "user": {
    "gateway_username": "user-Delta"
  }
}
...

The response contains:

...
{
  "meta": {
    "href": "/api/audit/sessions/stats",
    "parent": "/api/audit/sessions",
    "others": 3
  }
}
...

The response items contain:

...
[{
  "buckets": [
    
  ]
}]
...
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>The query was well-formed and statistics have been successfully retrieved.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQueryValue</td>
<td>The query is invalid, for example, it has an invalid value.</td>
</tr>
<tr>
<td>500</td>
<td>SearchUnavailable</td>
<td>The search backend is inaccessible.</td>
</tr>
</tbody>
</table>

Session histogram

The api/audit/sessions/histogram endpoint provides a histogram about the recorded sessions.

URL

GET https://<IP-address-of-SPS>/api/audit/sessions/histogram

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command retrieves statistical data about sessions.
curl --cookie cookies https://<IP-address-of-SPS>/api/audit/sessions/histogram

**Request parameters**

Use the following query parameters to fine-tune your request for statistics:

- **?q**: Narrow down the scope of the histogram using one or more properties (elements) of the sessions.
- **?field**: Create a histogram for the selected properties (elements and values) of sessions (for example, protocol).
  
  Using this parameter is mandatory.
- **?bin-size**: Determines the size of the unit for the histogram, for example, hour. SPS splits the queried period to intervals of this unit, and returns the number of sessions to each interval. For example, if you query an histogram from 2018-02-12:14:40 to 2018-02-16:14:40, and you set the bin-size to day, then SPS will return five datasets (one for each day). If you set the bin-size to week, then SPS will return only one dataset.
- **?start**: Create a histogram from the sessions that started after the specified date. Use the ISO 8601 format for the date, for example, 2017-01-25T10:00. By default, this is the one month before the date of the request.
- **?end**: Create a histogram from the sessions that ended before the specified date. Use the ISO 8601 format for the date, for example, 2017-01-25T11:00. By default, this is the date of the request.
- **?size**: Limit the range of values displayed in the histogram for a given field. The histogram will only be created for the top size number of most frequently occurring values (that is, values with the highest number of counts).

**Response**

The following snippet is a sample response received when retrieving a histogram about the audited sessions.

For details of the meta object, see Message format on page 10.

Those fields of the meta object that are specific to histograms are described in table Element.

```json
{
    "body": {
        "buckets": [
            {
                "active_count": 61, "id": "2018-01-15T12:00:00.000Z", "start_count": 61
            },
            {
                "active_count": 99, "id": "2018-01-15T13:00:00.000Z", "start_count": 89
            },
            {
                "active_count": 39, "id": "2018-01-15T14:00:00.000Z", "start_count": 24
            },
            {
                "active_count": 62, "id": "2018-01-15T15:00:00.000Z", "start_count":
```
```json
[{
    "active_count": 92,
    "id": "2018-01-15T16:00:00.000Z",
    "start_count": 81
},
{
    "active_count": 27,
    "id": "2018-01-15T17:00:00.000Z",
    "start_count": 19
}]
```

**Element** | **Type** | **Description**
--- | --- | ---
`body` | Top-level element (string) | Contains the properties that are in the scope of the requested histogram.

`buckets` | list | Contains the details of the histogram.

`active_count` | integer | The number of sessions that were active in this interval.

`id` | date | The starting date of the interval in ISO 8601 format.

`start_count` | integer | The number of sessions that were started in this interval.

**meta** | Top-level element (JSON object) | Contains metadata about the endpoint and the histogram.

`bin_size` | string | The size of the intervals used to create the histogram. You can change this using the `?bin_size` parameter of the request. Default value: `month`. Possible values: `second`, `minute`, `hour`, `day`, `week`, `month`, `year`.

`field` | string | Contains the name of the field that you requested statistics about.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>end</td>
<td>date</td>
<td>The date set in the <code>?end</code> parameter of the request. By default, this is the date of the request.</td>
</tr>
<tr>
<td>start</td>
<td>date</td>
<td>The date set in the <code>?start</code> parameter of the request. By default, this is one month before the date of the request.</td>
</tr>
<tr>
<td>time_zone</td>
<td>string</td>
<td>The time zone to use when calculating the intervals of the histogram, for example, Etc/UTC. By default, SPS uses UTC+0 (Zulu Time Zone). For the list of available time zones, see Element.</td>
</tr>
<tr>
<td>size</td>
<td>integer</td>
<td>The size that you specified in your query.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>The query was well-formed and the histogram has been successfully retrieved.</td>
</tr>
<tr>
<td>400</td>
<td>TooMuchBucketsInResult</td>
<td>Using the requested bin_size would result in too many intervals for the queried period.</td>
</tr>
<tr>
<td>400</td>
<td>NotSupportedContentOption</td>
<td>This endpoint does not support filtering in the content of sessions.</td>
</tr>
</tbody>
</table>

### Session alerts

The api/audit/sessions/<session-id>/alerts endpoint lists the alerts triggered in a session (if any). For details on configuring alerts, see Real-time content monitoring with Content Policies.

An event is listed as alert only if the Actions > Store in Connection Database option is selected in the Content Policy used to handle the session.

**URL**

```
GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/alerts
```
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the alerts of a session.

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/alerts"
```

Response

The following is a sample response received when listing the alerts of a session.

For details of the meta object, see Message format on page 10.

```
{
   "items": [
   {
   "alert_type": "adp.event.command",
   "channel_id": "0",
   "matched_action": "ls",
   "matched_content": "[myuser@examplehost ~]$ ls",
   "matched_regexp": "ls",
   "record_id": 94,
   "rule_name": "PatternMatcherRule",
   "time": "2017-04-25T13:26:39.144356"
   },
   {
   "alert_type": "adp.event.command",
   "channel_id": "0",
   "matched_action": "man man",
   "matched_content": "[myuser@examplehost ~]$ man man",
   ...
```

SPS 6.13.0 REST API Reference Guide
Search, download, and index sessions 751
"matched_regexp": "man",
"record_id": 197,
"rule_name": "PatternMatcherRule",
"time": "2017-04-25T13:34:15.265411"
}
],
"meta": {
  "first":
  "/api/audit/sessions/c7e51cebad1a3e2ade480909f7687b16/alerts?limit=500&offset=0",
  "href": "/api/audit/sessions/c7e51cebad1a3e2ade480909f7687b16/alerts",
  "last":
  "/api/audit/sessions/c7e51cebad1a3e2ade480909f7687b16/alerts?limit=500&offset=0",
  "limit": 500,
  "match_count": 3,
  "next": null,
  "offset": 0,
  "parent": "/api/audit/sessions/c7e51cebad1a3e2ade480909f7687b16",
  "previous": null,
  "remaining_seconds": 600
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>list</td>
<td>Top level element, a list containing the alerts of the session.</td>
</tr>
<tr>
<td>alert_type</td>
<td>string</td>
<td>The type of the event that triggered the alert. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- adp.event.command: A command entered in SSH or Telnet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- adp.event.screen.content: Alert triggered by the screen content.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- adp.event.screen.creditcard: Credit card numbers detected. Displayed only as an alert, not visible in the events.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- adp.event.screen.windowtitle: The title of the window in graphic protocols.</td>
</tr>
<tr>
<td>channel_id</td>
<td>string</td>
<td>The regular expression that matched the command line without prompt.</td>
</tr>
<tr>
<td>matched_action</td>
<td>integer</td>
<td>A reference to the ID of the channel in the session where the event occurred.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>matched_content</td>
<td>text</td>
<td>The content that occurred in the session and triggered the alert. Note that this value contains the context of the match as well. For example, if a Content Policy triggers an alert if a user types the <code>sudo</code> command, then the <code>psm.alerts.matched_content</code> value contains the entire command line, including the command prompt, for example, <code>myuser@examplehost:~$ man sudo</code></td>
</tr>
<tr>
<td>matched_regexp</td>
<td>text</td>
<td>The regular expression (match field) of the Content Policy that matched a part of the content and triggered the alert. For details, see Real-time content monitoring with Content Policies.</td>
</tr>
<tr>
<td>record_id</td>
<td>integer</td>
<td>The ID number of the alert within the session.</td>
</tr>
<tr>
<td>rule_name</td>
<td>string</td>
<td>The name of the content policy rule that triggered the alert. Note that this is not the name of the Content Policy.</td>
</tr>
<tr>
<td>time</td>
<td>string</td>
<td>The timestamp when the alert was triggered, for example, <code>2017-04-25T13:26:39.144356</code>.</td>
</tr>
</tbody>
</table>

**Changing the display limit**

You can use the `?limit` option to change the number of items displayed at once. The default limit is 500.

```
?limit=1000
```

To navigate beyond the displayed set, use the `offset` option.

**Navigating large datasets**

You can use the `?offset` option to navigate data sets that extend beyond the display limit. The default value of the offset is 0, this is the initially displayed set. To move to other items beyond the initial set, increase the value to a number that corresponds to the item where you want to start displaying results from.

Example: the display limit is the default 500, and the number of sessions is 1012. The initial 500 sessions are listed at:

```
?offset=0
```

To view sessions from 501 to 1000, change the offset to 501:

```
?offset=501
```

To display the remaining 12 sessions, change the offset to 1001:
Sorting and filtering

Sorting and filtering alerts is currently not supported. The items are automatically sorted by the record ID. The response includes every available field.

Session events

The api/audit/sessions/<session-id>/events endpoint lists the events extracted from a session (if any). Events are available only if the session is indexed. For details on configuring indexing, see Local services: configuring the indexer on page 764.

URL

GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/events

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the events of a session.

curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/events"
Response

The following is a sample response received when listing the events of a session. For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "action": "ls",
            "channels_id": "0",
            "content": "myuser@examplehost:~$ ls",
            "record_id": 46,
            "time": "2017-04-11T09:21:10.832",
            "type": "command"
        },
        {
            "action": "cd",
            "channels_id": "0",
            "content": "myuser@examplehost:~$ cd /cd",
            "record_id": 64,
            "time": "2017-04-11T09:21:15.488",
            "type": "command"
        },
        {
            "action": "cat 24hrs.txt",
            "channels_id": "0",
            "content": "myuser@examplehost:/var$ cat 24hrs.txt",
            "record_id": 78,
            "time": "2017-04-11T09:21:18.017",
            "type": "command"
        },
        {
            "action": "ls -la",
            "channels_id": "0",
            "content": "myuser@examplehost:/var$ ls -la",
            "record_id": 95,
            "type": "command"
        },
        {
            "action": "echo example.txt",
            "channels_id": "0",
            "content": "myuser@examplehost:/var$ echo example.txt",
            "record_id": 113,
            "time": "2017-04-11T09:21:23.353",
            "type": "command"
        },
        {
            "action": "ls",
            ...
```json
{
    "channels_id": "0",
    "content": "myuser@examplehost:/var$ man sudo",
    "record_id": 148,
    "time": "2017-04-11T09:21:27.017",
    "type": "command"
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>list</td>
<td>Top level element, a list containing the alerts of the session.</td>
</tr>
<tr>
<td>action</td>
<td>string</td>
<td>The command line without prompt in commands.</td>
</tr>
<tr>
<td>channels_id</td>
<td>integer</td>
<td>A reference to the ID of the channel in the session where the event occurred.</td>
</tr>
<tr>
<td>content</td>
<td>text</td>
<td>The event that occurred in the session. Note that this value contains the context of the event as well. For example, for command events in terminal sessions, the value contains the entire command line, including the command prompt. For example, <code>myuser@examplehost:~$ man sudo</code></td>
</tr>
<tr>
<td>record_id</td>
<td>integer</td>
<td>The ID number of the event within the session.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>The type of the event. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- command: A command entered in SSH or TelNet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- file_transfer: A file transfer event.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- http_request: An HTTP request initiated during the session.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- window_title: The title of the window in graphic</td>
</tr>
</tbody>
</table>

---

**Element Type Description**

- **items**: list - Top level element, a list containing the alerts of the session.
- **action**: string - The command line without prompt in commands.
- **channels_id**: integer - A reference to the ID of the channel in the session where the event occurred.
- **content**: text - The event that occurred in the session. Note that this value contains the context of the event as well. For example, for command events in terminal sessions, the value contains the entire command line, including the command prompt. For example, `myuser@examplehost:~$ man sudo`.
- **record_id**: integer - The ID number of the event within the session.
- **type**: string - The type of the event. Possible values:
  - command: A command entered in SSH or TelNet.
  - file_transfer: A file transfer event.
  - http_request: An HTTP request initiated during the session.
  - window_title: The title of the window in graphic.
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>string</td>
<td>The timestamp when the event occurred, for example, 2017-04-25T13:26:39.144356.</td>
</tr>
</tbody>
</table>

**Changing the display limit**

You can use the `?limit` option to change the number of items displayed at once. The default limit is 500.

```
?limit=1000
```

To navigate beyond the displayed set, use the `offset` option.

**Navigating large datasets**

You can use the `?offset` option to navigate data sets that extend beyond the display limit. The default value of the offset is 0, this is the initially displayed set. To move to other items beyond the initial set, increase the value to a number that corresponds to the item where you want to start displaying results from.

Example: the display limit is the default 500, and the number of sessions is 1012. The initial 500 sessions are listed at:

```
?offset=0
```

To view sessions from 501 to 1000, change the offset to 501:

```
?offset=501
```

To display the remaining 12 sessions, change the offset to 1001:

```
?offset=1001
```

**Filtering**

You can filter events at the `/api/audit/sessions/<session-id>/events` endpoint. Use the `?q` option to filter the list using one or more properties (elements) of the sessions.

```
?q=content:sudo
```

You can escape special characters using the backslash character.

```
?q=content:"Copying Files"
```

To add multiple elements to the filter, you can use the AND, AND NOT, and OR operators.
content:ls AND content:cp AND NOT content:mv

You can create groups using () (parentheses).

?q=(content:rm OR content:mv) AND channels_id:5

You can also use () (parentheses) to add multiple possible values for a property.

?q=content:(sudo rm)

You can use the * (asterisk) and ? (question mark) wildcards for string-type values.

?q=content:?dmi*

You can define ranges using [] (brackets) or {} (braces) and the TO operator. This only works for numeric (int) values.

- [ means equal or higher than the following value
- ] means equal or lower than the preceding value
- { means higher than the following value
- } means lower than the preceding value

For example, the following range resolves to 2:

?q=channels_id:{1 TO 3}

You can also use the * (asterisk) wildcard in the range.

?q=channels_id:{* TO 5}

Note that not all connection data can be used for filtering. The available elements are:

- channels_id: [integer] The channel in the session where the event occurred.
- content: [text] The event that occurred in the session.
- record_id: [integer] The identifier of the event in the session.
- time: [string] The timestamp when the event occurred.
- type: [string] The type of the event:
  - command: A command entered in SSH or Telnet.
  - screen.content: Screen content.
  - screen.creditcard: Credit card numbers detected. Displayed only as an alert, not visible in the events.
  - screen.windowtitle: The title of the window in graphic protocols.
Indexing sessions

The api/audit/sessions/<session-id>/indexing endpoint lists the indexing-related details in this session (if any). For details on configuring indexing, see Local services: configuring the indexer on page 764.

URL

GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/indexers

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the indexing-related details of a session.

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/indexing"
```

Response

The following is a sample response received when listing the indexing-related details of a session.

For details of the meta object, see Message format on page 10.
{  
  "items": [  
   {  
     "config": {  
       "command": {  
         "enabled": true  
       },  
       "keyboard": {  
         "buffer_interval": 3,  
         "enabled": false  
       },  
       "mouse": {  
         "buffer_interval": 1,  
         "enabled": false  
       },  
       "near_realtime": false,  
       "ocr_languages": [],  
       "screen": {  
         "enabled": true,  
         "omnipage_trade_off": "TO_ACCURATE"  
       },  
       "title": {  
         "enabled": true  
       }  
     },  
     "statistics": {  
       "cpu_time": 5,  
       "duration": 149,  
       "start_time": 1542116524143  
     },  
     "status": "COMPLETED",  
     "version": {  
       "adp": "6.0.20",  
       "worker": "4.0.26"  
     }  
   }  
  ],  
  "meta": {  
   "first":  
   "href": 
   "$ref": "$ref"  
  },  
  "last":  
  "limit": 500,  
  "match_count": 1,  
  "next": null,  
  "offset": 0}
"offset": 0,
"parent": "/api/audit/sessions/rUhhQZ3jYsY1NDwYp9DEpq",
"previous": null,
"remaining_seconds": 599
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>list</td>
<td>Top level element, a list containing the indexing-related details of the session. For details, see indexer_info section in &quot;List of available search queries&quot; in the Administration Guide.</td>
</tr>
</tbody>
</table>

**Changing the display limit**

You can use the ?limit option to change the number of items displayed at once. The default limit is 500.

?limit=1000

To navigate beyond the displayed set, use the offset option.

**Navigating large datasets**

You can use the ?offset option to navigate data sets that extend beyond the display limit. The default value of the offset is 0, this is the initially displayed set. To move to other items beyond the initial set, increase the value to a number that corresponds to the item where you want to start displaying results from.

Example: the display limit is the default 500, and the number of sessions is 1012. The initial 500 sessions are listed at:

?offset=0

To view sessions from 501 to 1000, change the offset to 501:

?offset=501

To display the remaining 12 sessions, change the offset to 1001:

?offset=1001
Session audit trail downloads

The api/audit/sessions/<session-id>/trail_downloads endpoint lists the details of audit-trail downloads in this session (if any). For details on downloading audit trails, see Local services: configuring the indexer on page 764.

URL

GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/trail_downloads

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the indexing-related details of a session.

```bash
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/trail_downloads"
```

Response

The following is a sample response received when listing the indexing-related details of a session.

For details of the meta object, see Message format on page 10.
{
    "items": [
        {
            "from_api": false,
            "ip_address": "10.20.30.40",
            "time": "2018-11-20T11:19:00Z",
            "username": "admin"
        },
        {
            "from_api": false,
            "ip_address": "10.20.30.40",
            "time": "2018-11-20T11:38.000Z",
            "username": "admin"
        }
    ],
    "meta": {
        "first": "/api/audit/sessions/c7e51cebad1a3e2ade480909f7687b16/indexer?limit=500&offset=0",
        "href": "/api/audit/sessions/c7e51cebad1a3e2ade480909f7687b16/indexer",
        "last": "/api/audit/sessions/c7e51cebad1a3e2ade480909f7687b16/indexer?limit=500&offset=0",
        "limit": 5,
        "match_count": 2,
        "next": null,
        "offset": 0,
        "parent": "/api/audit/sessions/rUhhQZ3jYs1NDWyp9DEpq",
        "previous": null,
        "remaining_seconds": 599
    }
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>list</td>
<td>Top level element, a list containing the indexing-related details of the session.</td>
</tr>
<tr>
<td>from_api</td>
<td>boolean</td>
<td>True, if the audit trail was not downloaded from the GUI, but through SOAP or REST API.</td>
</tr>
<tr>
<td>ip_address</td>
<td>string</td>
<td>The IP address of the client that downloaded the audit trail.</td>
</tr>
<tr>
<td>time</td>
<td>boolean</td>
<td>The exact time when the user downloaded the audit trail file.</td>
</tr>
<tr>
<td>username</td>
<td>string</td>
<td>The user name of the user who downloaded the audit trail.</td>
</tr>
</tbody>
</table>
Changing the display limit

You can use the \( \text{?limit} \) option to change the number of items displayed at once. The default limit is 500.

\( \text{?limit=1000} \)

To navigate beyond the displayed set, use the \( \text{offset} \) option.

Navigating large datasets

You can use the \( \text{?offset} \) option to navigate data sets that extend beyond the display limit. The default value of the offset is 0, this is the initially displayed set. To move to other items beyond the initial set, increase the value to a number that corresponds to the item where you want to start displaying results from.

Example: the display limit is the default 500, and the number of sessions is 1012. The initial 500 sessions are listed at:

\( \text{?offset=0} \)

To view sessions from 501 to 1000, change the offset to 501:

\( \text{?offset=501} \)

To display the remaining 12 sessions, change the offset to 1001:

\( \text{?offset=1001} \)

Local services: configuring the indexer

Indexing is a resource intensive (CPU and hard disk) operation, and depending on the number of processed audit trails and parallel connections passing SPS, may affect the performance of SPS. Test it thoroughly before enabling it in a production environment that is under heavy load. If your SPS appliance cannot handle the connections and the indexing, consider using external indexers (see "Configuring external indexers" in the Administration Guide) to decrease the load on SPS. For sizing recommendations, ask your One Identity partner or contact our Support Team.

**NOTE:** Only those audit trails will be processed that were created after full-text indexing had been configured for the connection policy. It is not possible to process already existing audit trails.

**NOTE:** Using content policies significantly slows down connections (approximately 5 times slower), and can also cause performance problems when using the indexer service.
## URL

GET https://<IP-address-of-SPS>/api/configuration/local_services/indexer

## Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d830e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

## Sample request

The following command lists the configuration options.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/local_services/indexer
```

## Response

The following is a sample response received when external indexers are disabled.

For details of the meta object, see Message format on page 10.

```json
{
    "body": {
        "decryption_keys": [
            {
                "key": "e38d47bd-5374-4d7c-b683-e26ea77142e2",
                "meta": {
                    "href": "/api/configuration/x509/e38d47bd-5374-4d7c-b683-e26ea77142e2"
                }
            }
        ],
```
A sample response when external indexers are enabled:

```json
{
    "body": {
        "decryption_keys": [],
        "number_of_workers": 1,
        "number_of_workers": 0,
        "remote_access": {
            "access_restriction": {
            "allowed_from": [
                "10.40.0.0/16",
            ],
            "enabled": true,
            "enable": true,
            "listen": [
                "address": {
                    "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1",
                    "meta": {
                        "href": "/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/1"
                    }
                },
                "port": 12345
            ],
            "ssl_config": {
```
"ca": {
    "key": "52735ce4-4a43-458d-8803-c23c715640a5",
    "meta": {
        "href": "/api/configuration/x509/52735ce4-4a43-458d-8803-c23c715640a5"
    }
},
"service": {
    "key": "60eacdba-d889-4cb4-bdb0-cbbd4054f01c",
    "meta": {
        "href": "/api/configuration/x509/60eacdba-d889-4cb4-bdb0-cbbd4054f01c"
    }
},
"worker": {
    "key": "93198544-1e82-4661-90b7-e01b0b1e2ed9",
    "meta": {
        "href": "/api/configuration/x509/93198544-1e82-4661-90b7-e01b0b1e2ed9"
    }
},
"selection": "integrated"
},
"key": "indexer",
"meta": {
    "first": "/api/configuration/local_services/admin_web",
    "href": "/api/configuration/local_services/indexer",
    "last": "/api/configuration/local_services/user_web",
    "next": "/api/configuration/local_services/postgresql",
    "parent": "/api/configuration/local_services",
    "previous": "/api/configuration/local_services/admin_web",
    "remaining_seconds": 599,
    "transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the endpoint.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration options of the indexer service.</td>
</tr>
<tr>
<td>decryption_keys</td>
<td>list</td>
<td>Indexing encrypted audit trails requires the SPS 6.13.0 REST API Reference Guide</td>
</tr>
</tbody>
</table>
### X.509 certificates and the matching private keys.

X.509 certificates and the matching private keys. The certificates must be in PEM format, and use RSA keys. This parameter lists the reference IDs of the configured decryption keys. When configuring the indexer, you must first upload the keys before you can configure the decryption keys. For details, see [Private keys stored on SPS](#) on page 267.

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>The ID of the referenced decryption key. You can upload private keys at the /api/configuration/private_key endpoint. For details, see <a href="#">Private keys stored on SPS</a> on page 267.</td>
</tr>
</tbody>
</table>

### number_of_near_realtime_workers

The number of indexer workers configured to perform near-realtime indexing. For details, see "Configuring the external indexer" in the Administration Guide.

### number_of_workers

This option determines the maximum number of parallel indexing tasks that the SPS appliance performs. The default value is set to the number of detected CPU cores. Note that indexing audit trails require 50-100 Mbytes of memory for terminal sessions (SSH, Telnet, TN3270), and 150-300 Mbytes for graphical sessions (RDP, ICA, VNC, X11). Consider the memory usage of your SPS host before modifying this value.

### remote_access

Enables external indexers to access the SPS host, and configures access restrictions and other parameters.

### selection

The value of this option must be integrated.

### access restriction

Enables and configures limitations on the clients that can access the web interface, based on the IP address of the clients.

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>access_restriction</td>
<td>JSON object</td>
<td>Enables and configures limitations on the clients that can access the web interface, based on the IP address of the clients.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allowed_from</td>
<td>list</td>
<td>The list of IP networks from where the administrators are permitted to access this management interface. To specify the IP addresses or networks, use the IPv4-Address/prefix format, for example, 10.40.0.0/16.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>boolean</td>
<td>Set it to true to restrict access to the specified client</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>enabled</td>
<td>boolean-boolean</td>
<td>Enables the remote access for the external indexers. That way, indexer services running on external hosts can access the audit trails, index them, and upload the indexed data to SPS. If this option is set to False, SPS ignores every other option of this object. For details on installing and configuring external indexers, see &quot;Configuring external indexers&quot; in the Administration Guide.</td>
</tr>
</tbody>
</table>

**CAUTION:**
Disabling an already configured remote indexer access causes SPS to delete every related certificate. If you re-enable remote indexer access, SPS generates new certificates, and you have to import them to the external indexer hosts.

<table>
<thead>
<tr>
<th>listen</th>
<th>list</th>
<th>Selects the network interface, IP address, and port where the clients can access the web interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>JSON object</td>
<td>A reference to a configured network interface and IP address where this local service accepts connections. For example, if querying the interface /api/configuration/network/nics/nic1#interfaces/ff754025754b3df1647001/addresses/ returns the following response:</td>
</tr>
</tbody>
</table>

```json
{
   "body": {
      "interfaces": {
         "@order": [
            "ff754025754b3df1647001"
        ],
        "ff754025754b3df1647001": {
           "addresses": {
              "1": "10.40.255.171/24",
              "@order": [
                 "1"
              ]
           },
           "name": "default",
           "vlantag": 0
        }
      }
   }
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;: &quot;eth0&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;speed&quot;: &quot;auto&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;key&quot;: &quot;nic1&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;meta&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;first&quot;: &quot;/api/configuration/network/nics/nic1&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;href&quot;: &quot;/api/configuration/network/nics/nic1&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;last&quot;: &quot;/api/configuration/network/nics/nic3&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;next&quot;: &quot;/api/configuration/network/nics/nic2&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;parent&quot;: &quot;/api/configuration/network/nics&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;previous&quot;: null,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;transaction&quot;: &quot;/api/transaction&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Then the listening address of the local service is the following.

```
nic1.interfaces.ff7574025754b3df1647001.addresses.1
```

This is the format you have to use when configuring the address of the local service using REST:

```
"address": "nic1.interfaces.ff7574025754b3df1647001.addresses.1"
```

When querying a local services endpoint, the response will contain a reference to the IP address of the interface in the following format:

```
"address": { |
| "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1", |
| "meta": { |
| "href": "/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/1"
```

SPS 6.13.0 REST API Reference Guide
Search, download, and index sessions
Element | Type | Description
--- | --- | ---
port | integer | The port number where this local service accepts connections.
ssl_config | JSON object | Contains references to the certificates used to encrypt the communication between SPS and the external indexer hosts. SPS generates these certificates automatically when you enable the indexer service.
ca | reference | The ID of the CA certificate used to sign the certificates used to communicate between SPS and the external indexers.
service | reference | The ID of the certificate that SPS shows to the external indexer hosts.
worker | reference | The ID of the certificate that the external indexer hosts must show to SPS.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Updating the indexer configuration**

To update the configuration of the indexer, you have to PUT the updated configuration in JSON format to the endpoint, for example:
Indexer policies

Indexer policies allow you to configure the Optical Character Recognition (OCR) engine of SPS, and specify which languages it should use. Only graphical protocols (RDP, Citrix ICA, VNC) are affected.

**NOTE:** In the case of graphical protocols, the default Optical Character Recognition (OCR) configuration is automatic language detection. This means that the OCR engine will attempt to detect the languages of the indexed audit trails automatically. However, if you know in advance what language(s) will be used, create a new Indexer Policy.

If you specify the languages manually, note the following:

- Specifying only one language provides the best results in terms of performance and precision.
- The English language is always detected along with the non-English languages that you have configured. However, if you want the OCR to only recognize the English language, you have to select it from the list of languages.
There are certain limitations in the OCR engine when recognizing languages with very different character sets. For this reason, consider the following:

- When selecting Asian languages (Simplified Chinese, Traditional Chinese, Korean), avoid adding languages that use the Latin alphabet.
- When selecting the Arabic language, avoid selecting any other languages.
- The Thai language is currently not supported. If you are interested in using SPS to index Thai texts, contact our Sales Team.

### URL

```
GET https://<IP-address-of-SPS>/api/configuration/policies/indexing
```

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists the available indexer policies.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/indexing
```

The following command displays a specific indexer policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/indexing/<id-of-the-policy>
```

### Response

The following is a sample response received when querying the /api/configuration/policies/indexing/ endpoint.
For details of the meta object, see Message format on page 10.

```
{
   "items": [
      {
         "key": "-50000",
         "meta": {
            "href": "http://example.com/api/configuration/policies/indexing/-50000"
         }
      },
      {
         "key": "13442970955825a89b55e46",
         "meta": {
            "href": "http://example.com/api/configuration/policies/indexing/13442970955825a89b55e46"
         }
      }
   ],
   "meta": {
      "first": "http://example.com/api/configuration/policies/audit_policies",
      "href": "http://example.com/api/configuration/policies/indexing",
      "last": "http://example.com/api/configuration/policies/usermapping_policies",
      "next": "http://example.com/api/configuration/policies/ldap_servers",
      "parent": "http://example.com/api/configuration/policies",
      "previous": "http://example.com/api/configuration/policies/credentialstores",
      "remaining_seconds": 599,
      "transaction": "http://example.com/api/transaction"
   }
}
```

A sample response when querying a specific indexer policy:

```
{
   "body": {
      "index": {
         "command": true,
         "keyboard": false,
         "mouse": false,
         "screen_content": false,
         "window_title": true
      },
      "name": "english-german-russian",
      "ocr": {
         "accuracy": "accurate",
         "custom_languages": {
            "enabled": true,
            "languages": ["eng",
```
```
{"deu",
"rus"
}
}

"key": 
"-50000",

"meta": {
"first": "/api/configuration/policies/indexing/-50000",
"href": "/api/configuration/policies/indexing/-50000",
"last": "/api/configuration/policies/indexing/-50000",
"next": null,
"parent": "/api/configuration/policies/indexing",
"previous": null,
"remaining_seconds": 599,
"transaction": "/api/transaction"
}
```

<table>
<thead>
<tr>
<th><strong>Element</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the policy.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the configuration options of the indexer policy.</td>
</tr>
<tr>
<td>index</td>
<td>Top level element</td>
<td>Contains the indexed events of the indexer policy. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- command: A command entered in SSH or Telnet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- keyboard: Keyboard-related events, for example, pressing Enter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- mouse: Mouse-related events, for example, mouse clicks.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- screen_content: Screen content elements, for example, commands, window titles, IP addresses, user names, and so on.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the indexer policy.</td>
</tr>
<tr>
<td>ocr</td>
<td>JSON object</td>
<td>Configuration of the OCR engine.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>accuracy</td>
<td>string</td>
<td>Accuracy level for Optical Character Recognition. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>fast</strong>: The fastest option with potentially less accurate results. Select this option if speed is more important to you than getting the most accurate results possible.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>balanced</strong>: Fairly accurate option with less than optimum speed. Select this option if you want results to be fairly accurate but you have more than a few sessions to process and processing time is less of a concern.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>accurate</strong>: The most accurate option with less optimal speed. Select this option if you must have the most accurate results possible and speed is less important or you only have a few sessions to process.</td>
</tr>
</tbody>
</table>

### Custom languages

**Elements**

| Custom languages elements | Type       | Description                                                                 ||
|----------------------------|------------|-----------------------------------------------------------------------------|
| custom_languages          | Top level element | Configures what languages to detect.                                       |

| enabled | boolean | If false, the OCR engine detects the language of the text automatically. This is the default behavior. To specify which languages to use, set the custom_languages element to true, and list the abbreviation of the languages in the languages element (for example, "eng", "ger"). |

| languages | list | The list of languages the OCR engine should use to process graphical protocols. To specify which languages to use, set the custom_languages element to true, and list the abbreviation of the languages in the languages element (for example, "eng", "ger"). |
|           |      | - Specifying only one language provides the best results in terms of performance and precision. |
|           |      | - The English language is always detected |
Custom languages elements

<table>
<thead>
<tr>
<th>Description</th>
<th>Type</th>
<th>Custom languages elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>along with the non-English languages that you have configured. However, if you want the OCR to only recognize the English language, you have to select it from the list of languages.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>There are certain limitations in the OCR engine when recognizing languages with very different character sets. For this reason, consider the following:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- When selecting Asian languages (Simplified Chinese, Traditional Chinese, Korean), avoid adding languages that use the Latin alphabet.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- When selecting the Arabic language, avoid selecting any other languages.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- The Thai language is currently not supported. If you are interested in using SPS to index Thai texts, contact our Sales Team.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Custom languages elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict</td>
<td>No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction on page 29.</td>
</tr>
</tbody>
</table>

### Add an indexing policy

To add an indexing policy, you have to:
1. **Open a transaction.**
   
   For more information, see *Open a transaction* on page 29.

2. **Create the JSON object for the new indexing policy.**
   
   You can find a detailed description of the available parameters listed in *Element*. POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/policies/indexing` endpoint. If the POST request is successful, the response includes the key of the new ticketing policy. For example:

   ```json
   {
     "key": "aa423b72-0d0f-4275-be30-494e9a9ffad",
     "meta": {
       "href": "/api/configuration/policies/indexing/aa423b72-0d0f-4275-be30-494e9a9ffad",
       "parent": "/api/configuration/policies/indexing",
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes.**
   
   For more information, see *Commit a transaction* on page 31.
Reporting

List of endpoints for configuring reporting, and accessing the generated reports.

URL

GET https://<IP-address-of-SPS>/api/configuration/reporting

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Sample request

The following command lists the available endpoints.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting
Response

The following is a sample response received when listing the available endpoints.

For details of the meta object, see Message format on page 10.

```json
{
    "meta": {
        "first": "/api/configuration/aaa",
        "href": "/api/configuration/reporting",
        "last": "/api/configuration/x509",
        "next": "/api/configuration/ssh",
        "parent": "/api/configuration",
        "previous": "/api/configuration/rdp",
        "transaction": "/api/transaction"
    },
    "items": [
        {
            "key": "content_subchapters",
            "meta": {
                "href": "/api/configuration/reporting/content_subchapters"
            }
        },
        {
            "key": "custom_subchapters",
            "meta": {
                "href": "/api/configuration/reporting/custom_subchapters"
            }
        },
        {
            "key": "predefined_reports",
            "meta": {
                "href": "/api/configuration/reporting/predefined_reports"
            }
        },
        {
            "key": "reports",
            "meta": {
                "href": "/api/configuration/reporting/reports"
            }
        },
        {
            "key": "statistics_subchapters",
            "meta": {
                "href": "/api/configuration/reporting/statistics_subchapters"
            }
        }
    ]
}
```
### Endpoint Description

<table>
<thead>
<tr>
<th>Endpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content_subchapters</td>
<td>List of the reporting subchapters created from audit trail content (statistics of search keywords, and screenshots).</td>
</tr>
<tr>
<td>custom_subchapters</td>
<td>List of the reporting subchapters created from custom queries to the SPS connection database.</td>
</tr>
<tr>
<td>predefined_reports</td>
<td>List of the pre-defined reports available on SPS.</td>
</tr>
<tr>
<td>reports</td>
<td>List of the configured reports.</td>
</tr>
<tr>
<td>statistics_subchapters</td>
<td>List of the reporting subchapters created from connection statistics.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Reports

List of the configured reports.

### URL

```
GET https://<IP-address-of-SPS>/api/configuration/reporting/reports
```
## Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19. <strong>NOTE:</strong> This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

## Sample request

The following command lists the configured reports.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/reports
```

The following command retrieves the properties of a specific report.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/reports/<key_value>
```

## Response

The following is a sample response received when listing reports.

For details of the meta object, see [Message format](#) on page 10.

```
{
   "meta": {
      "first": "/api/configuration/reporting/content_subchapters",
      "href": "/api/configuration/reporting/reports",
      "last": "/api/configuration/reporting/statistics_subchapters",
      "next": "/api/configuration/reporting/statistics_subchapters",
      "parent": "/api/configuration/reporting",
      "previous": "/api/configuration/reporting/predefined_reports",
      "transaction": "/api/transaction"
   },
   "items": [  
```

---

SPS 6.13.0 REST API Reference Guide Reporting 783
When retrieving the endpoint of a specific report, the response is the following.

```
{
  "body": {
    "access": [
      "report"
    ],
    "chapters": [
      {
        "name": "System health",
        "subchapters": [
          {
            "name": "system_health_network_connections",
            "selection": "builtin"
          },
          {
            "name": "system_health_load_average",
            "selection": "builtin"
          }
        ]
      },
      {
        "name": "All connections",
        "subchapters": [
          {
            "name": "connection_each_scb_top10_channel_types_each",
            "selection": "builtin"
          },
          {
            "name": "connection_each_scb_top10_portforward_targets_each",
            "selection": "builtin"
          }
        ]
      }
    ]
  }
}
```
{
  "name": "Search statistics",
  "subchapters": [  
    {
      "reference": "21111736175707f1df8bea1",
      "selection": "custom"
    }
  ],
  "name": "Misc",
  "subchapters": [  
    {
      "reference": "13869311625707e0a3e0892",
      "selection": "custom"
    }
  ],
  "name": "Advanced statistics",
  "subchapters": [  
    {
      "reference": "5983143445707eb740fee3",
      "selection": "custom"
    }
  ]
},
"email_recipients": {  
  "recipients": [  
    "admin@company.com"
  ],
  "selection": "other"
},
"frequency": {  
  "day": false,
  "month": true,
  "week": false
},
"logo_id": "logoC890jH",
"name": "all-options",
"send_report_in_email": true
},
"key": "8292675195707f19d932af",
"meta": {
  "first": "/api/configuration/reporting/reports/7798770004e472c8576912",
  "href": "/api/configuration/reporting/reports/8292675195707f19d932af"}
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the report</td>
</tr>
<tr>
<td>body</td>
<td>Top level</td>
<td>The elements of the report.</td>
</tr>
<tr>
<td>access</td>
<td>list</td>
<td>Required. List of access control groups whose members can access the subchapter. To deny access to the report, use &quot;admin&quot; as the only value for the element.</td>
</tr>
<tr>
<td>chapters</td>
<td>Top level</td>
<td>A chapter of the report.</td>
</tr>
<tr>
<td>email_recipients</td>
<td>Top level</td>
<td>Contains the list of e-mails where the generated report is sent.</td>
</tr>
<tr>
<td>recipients</td>
<td>list</td>
<td>Custom list of e-mails where the generated report is sent. To use a custom list, the selection element must be set to other.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>This element can have two values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- default uses the e-mail address configured in the reporting_address element of the https://&lt;IP-address-of-SPS&gt;/api/configuration/management/email endpoint (or the Basic Settings &gt; Management &gt; Mail settings &gt; Send reports to field on the web UI).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- other uses the e-mails listed in the recipients element.</td>
</tr>
<tr>
<td>frequency</td>
<td>Top</td>
<td>Contains the list of options for defining the</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>day</td>
<td>boolean</td>
<td>Set it to <code>true</code> to generate the report each day.</td>
</tr>
<tr>
<td>month</td>
<td>boolean</td>
<td>Set it to <code>true</code> to generate the report each month.</td>
</tr>
<tr>
<td>week</td>
<td>boolean</td>
<td>Set it to <code>true</code> to generate the report each week.</td>
</tr>
<tr>
<td>logo_id</td>
<td>string</td>
<td>The ID of the custom logo. The null value means the report is generated using the default logo. You can upload a custom logo on the web UI of SPS, using the Reporting &gt; Configuration &gt; &lt;report&gt; &gt; Choose new logo button.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the report.</td>
</tr>
<tr>
<td>send_report_in_email</td>
<td>boolean</td>
<td>Set it to <code>false</code> if you do not want to include the generated report in the e-mail.</td>
</tr>
</tbody>
</table>

### Chapters elements

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the chapter.</td>
</tr>
<tr>
<td>subchapters</td>
<td>list</td>
<td>List of subchapters included in the chapter.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the built-in subchapter included in the chapter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For the list of the built-in subchapters, see Built-in subchapters on page 792.</td>
</tr>
<tr>
<td>reference</td>
<td>string</td>
<td>The key of the custom, content, or statistics subchapter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- For the keys of the reporting subchapters created from custom queries to the SPS connection database, see the custom_subchapters endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- For the keys of the reporting subchapters created from audit trail</td>
</tr>
<tr>
<td>Chapters elements</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>content</td>
<td></td>
<td>(statistics of search keywords, and screenshots), see the reporting/content_subchapters endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- For the keys of the reporting subchapters created from connection statistics, see the reporting/statistics_subchapters endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To include a custom, content, or statistics subchapter, use the value of its key element, not the name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>selection</th>
<th>string</th>
<th>This element can have two values:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>- Set builtin for the default reporting subchapters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Set custom for all other subchapters (custom, content or statistics).</td>
</tr>
</tbody>
</table>

**Examples:**

Set the e-mail recipients to the default (as configured in the reporting_address element of the /api/configuration/management/email endpoint):

```json
"email_recipients": {
  "selection": "default"
}
```

Create a custom set of e-mail recipients:

```json
"email_recipients": {
  "recipients": [
    "<email-1>",
    "<email-2>"
  ],
  "selection": "other"
}
```

Add a reporting chapter with built-in subchapters:
"chapters": [
  {
    "name": "<custom-name>",
    "subchapters": [
      {
        "name": "system_health_filesystem_usage",
        "selection": "builtin"
      },
      {
        "name": "system_health_network_connections",
        "selection": "builtin"
      },
      {
        "name": "system_health_load_average",
        "selection": "builtin"
      }
    ]
  }
]

Add a reporting chapter with custom, content, or statistics subchapters:

"chapters": [
  {
    "name": "<custom-name>",
    "subchapters": [
      {
        "reference": "<key-of-subchapter>",
        "selection": "custom"
      },
      {
        "reference": "<key-of-subchapter>",
        "selection": "custom"
      }
    ]
  }
]

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>400</td>
<td>IncompleteConfigurationSubtreeError</td>
<td>Possible cause: PUT operation on the reports endpoint, instead of POST.</td>
</tr>
<tr>
<td>400</td>
<td>IncompleteConfigurationSubtreeError &quot;missing_paths&quot;: [ &quot;email_recipients/recipients&quot; ]</td>
<td>You have selected other for the selection element under email_recipients, but did not provide a list using recipients.</td>
</tr>
<tr>
<td>400</td>
<td>IncompleteConfigurationSubtreeError Syntax error: &quot;No such property; property='recipients'&quot;</td>
<td>Do not provide recipients if you set the selection element under email_recipients to default.</td>
</tr>
<tr>
<td>400</td>
<td>IncompleteConfigurationSubtreeError &quot;missing_paths&quot;: [ &quot;chapters/7/subchapters/0/name&quot; ]</td>
<td>Verify that the selection element of the subchapter is correctly set to builtin or custom.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Add a report

To add a report, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.
2. **Create the JSON object for the new report.**

   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/reporting/reports endpoint. You can find a detailed description of the available parameters listed in [Element](#).

   If the POST request is successful, the response includes the key of the new report.

   ```json
   {
     "key": "26ddf648-9a21-4a7f-af56-9cea575785a9",
     "meta": {
       "href": "https://<IP-address-of-SPS>/api/configuration/reporting/reports/26ddf648-9a21-4a7f-af56-9cea575785a9",
       "parent": "https://<IP-address-of-SPS>/api/configuration/reporting/reports",
       "transaction": "https://<IP-address-of-SPS>/api/transaction"
     }
   }
   ```

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Modify a report**

To modify a report, you have to:

1. **Open a transaction.**

   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the report.**

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/reporting/reports/<key-of-the-report> endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

**Generate a report for a custom time period**

To generate a report for a custom time period, you have to:

1. **Define the custom time period for the report.**

   GET https://<IP-address-of-SPS>/api/configuration/reporting/reports. Search for the name of the report that you want to run on a custom time period. Copy the
2. POST the parameters to the https://<IP-address-of-SPS>/api/reports endpoint. The following parameter is required:
   - configuration_id

The following parameters are optional.
   - start: start timestamp in the format of either YYYY-MM-DD or YYYY-MM-DDTHH:MM.
   - end: end timestamp in the format of either YYYY-MM-DD or YYYY-MM-DDTHH:MM.
   - force: By default: False. If you set it to True, you can regenerate a report that has already been generated before.

If you do not enter the optional parameters, the start timestamp defaults to 1970.01.01 and the end timestamp defaults to the timestamp of when the report was generated.

**NOTE:** Timestamps are according to UTC.

This means that for example, if you are located in an UTC+1 region, a report that has the end parameter configured as 2020-01-18 will actually have an end date/time of 2020-01-18 01:00.

**Example: Generate a report for a custom time period**

```
https://198.51.100.0/api/reports?configuration_id=8292675195707f19d932af&start=2020-02-01&end=2020-02-18
```

3. You will receive a response similar to the following:

```json
{
   "message": "Report generation started.",
   "meta": {
      "href": "/api/reports",
      "parent": "/api"
   }
}
```

### Built-in subchapters

To create reports, you can use a number of predefined reporting subchapters. The following sections list the short description of each subchapter, as displayed on the web UI of SPS, and its name you can use to configure reports using the REST API.
**Configuration changes**

- Configuration changes - Changes by pages:
  configuration_changes_changes_by_pages
- Configuration changes - Changes by users:
  configuration_changes_changes_by_users
- Configuration changes - Changes in time:
  configuration_changes_changes_in_time
- Configuration changes - Special events:
  configuration_changes_special_events
- Configuration changes - Password change:
  configuration_changes_password_change

**Connection summary**

- Channels table
  connection_aggregate_scb_channels
- Distribution of channels
  connection_aggregate_scb_channeldist
- Channels history
  connection_aggregate_scb_channelshist
- Verdicts history by channels
  connection_aggregate_scb_verdicthist
- Usernames
  connection_aggregate_scb_usernames
- Accepted usernames
  connection_aggregate_scb_accepted_usernames
- Remote usernames
  connection_aggregate_scb_remote_usernames
- Accepted remote usernames
  connection_aggregate_scb_accepted_remote_usernames
- Four-eyes authorizers
  connection_aggregate_scb_4eyes_authorizers
- Source addresses
  connection_aggregate_scb_source_addresses
- Server addresses
  connection_aggregate_scb_server_addresses
- Top 10 usernames in denied channels
  connection_aggregate_scb_top10_users_in_denied_channels
- Top 10 denied usernames in channels
  connection_aggregate_scb_top10_denied_users
- Top 10 denied servers in channels
  connection_aggregate_scb_top10_denied_servers
- Top 10 denied channel types
  connection_aggregate_scb_top10_denied_channeltypes
- Top 10 longest sessions
  connection_aggregate_scb_top10_longest_sessions
- Top 10 shortest sessions
  connection_aggregate_scb_top10_shortest_sessions

**System health**

- System health - Filesystem usage
  system_health_filesystem_usage
- System health - Network connections
  system_health_network_connections
- System health - Load average
  system_health_load_average

**All connections**

- Top 10 usernames in each connection
  connection_each_scb_top10_users_each
- Top 10 accepted usernames in each connection
  connection_each_scb_top10_accepted_users_each
- Top 10 remote usernames in each connection
  connection_each_scb_top10_remote_users_each
- Top 10 username/four-eyes authorizer in each connection
  connection_each_scb_top10_4eyes_authorizers_each
- Top 10 servers in each connection
  connection_each_scb_top10_servers_each
- Top 10 username/server in each connection
  connection_each_scb_top10_username_server_connection_each
- Top 10 username/remote user in each connection
connection_each_scb_top10_remoteusers_each

- Top 10 commands over SSH session-exec channel in each connection
  connection_each_scb_top10_exec_commands_each
- Top 10 channel types in each connection
  connection_each_scb_top10_channel_types_each
- Top 10 Port forward targets in each connection
  connection_each_scb_top10_portforward_targets_each

**Specific connections**

You can also use subchapters for a specific connection. You have to use the protocol and the key of the connection.

The following examples assume that the connection's protocol is SSH, and its key is 8348340645707e2575e3c6.

- Top 10 usernames in "<connection_name>"
  connection_<protocol>_scb_top10_users_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_users_ssh-8348340645707e2575e3c6

- Top 10 accepted usernames in "<connection_name>"
  connection_<protocol>_scb_top10_accepted_users_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_accepted_users_ssh-8348340645707e2575e3c6

- Top 10 remote usernames in "<connection_name>"
  connection_<protocol>_scb_top10_remote_users_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_remote_users_ssh-8348340645707e2575e3c6

- Top 10 username/four-eyes authorizer in "<connection_name>"
  connection_<protocol>_scb_top10_4eyes_authorizers_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_4eyes_authorizers_ssh-8348340645707e2575e3c6

- Top 10 servers in "<connection_name>"
  connection_<protocol>_scb_top10_servers_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_servers_ssh-8348340645707e2575e3c6

- Top 10 username/server in "<connection_name>"
  connection_<protocol>_scb_top10_username_server_connection_<protocol>-<key>
  Example:
connection_ssh_scb_top10_username_server_connection_ssh-8348340645707e2575e3c6

- Top 10 username/remote user in "<connection_name>"
  connection_<protocol>_scb_top10_remoteusers_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_remoteusers_ssh-8348340645707e2575e3c6

- Top 10 commands over SSH session-exec channel in "<connection_name>"
  connection_<protocol>_scb_top10_exec_commands_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_exec_commands_ssh-8348340645707e2575e3c6

- Top 10 channel types in "<connection_name>"
  connection_<protocol>_scb_top10_channel_types_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_channel_types_ssh-8348340645707e2575e3c6

- Top 10 Port forward targets in "<connection_name>"
  connection_<protocol>_scb_top10_portforward_targets_<protocol>-<key>
  Example:
  connection_ssh_scb_top10_portforward_targets_ssh-8348340645707e2575e3c6

**Pre-defined reports**

You can configure the compliance reports of SPS using the predefined_reports endpoint.

To help you comply with the regulations of the Payment Card Industry Data Security Standard (PCI DSS), One Identity Safeguard for Privileged Sessions (SPS) can generate reports on the compliance status of SPS. Note that this is not a fully-featured compliance report: it is a tool to enhance and complement your compliance report by providing information available in SPS. The report corresponds with the document *Payment Card Industry (PCI) Data Security Standard, Requirements and Security Assessment Procedures, Version 3.0*, published by the PCI Security Standards Council.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/reporting/predefined_reports
Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19. NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).</td>
</tr>
</tbody>
</table>

Sample request

The following request command lists the pre-defined reports available on SPS.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/predefined_reports
```

The following command retrieves the properties of a specific report.

```
curl --cookie cookies https://<IP-address-of-SPS>/api//configuration/reporting/predefined_reports/<report-key>
```

Response

The following is a sample response received when listing pre-defined reports.

For details of the meta object, see Message format on page 10.

```
{
    "meta": {
        "first": "/api/configuration/reporting/content_subchapters",
        "href": "/api/configuration/reporting/predefined_reports",
        "last": "/api/configuration/reporting/statistics_subchapters",
        "next": "/api/configuration/reporting/reports",
        "parent": "/api/configuration/reporting",
        "previous": "/api/configuration/reporting/custom_subchapters",
        "transaction": "/api/transaction"
    },
    "items": []
}
```
When retrieving the endpoint of a specific report, the response is the following.

```
{
    "key": "pcidss",
    "meta": {
        "first": "/api/configuration/reporting/predefined_reports/pcidss",
        "href": "/api/configuration/reporting/predefined_reports/pcidss",
        "last": "/api/configuration/reporting/predefined_reports/pcidss",
        "next": null,
        "parent": "/api/configuration/reporting/predefined_reports",
        "previous": null,
        "transaction": "/api/transaction"
    },
    "pcidss": {
        "access": ["report"],
        "email_recipients": {
            "selection": "default"
        },
        "name": "PCI-DSS",
        "send_report_in_email": true
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the report.</td>
</tr>
<tr>
<td>&lt;id-of-the-report&gt;</td>
<td>Top level item</td>
<td>The elements of the pre-defined report.</td>
</tr>
<tr>
<td>access</td>
<td>list</td>
<td>List of access control groups whose members can access the report.</td>
</tr>
<tr>
<td>email_recipients</td>
<td>Top level item</td>
<td>Contains the list of e-mails where the generated report is sent.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>recipient</td>
<td>list</td>
<td>Custom list of e-mails where the generated report is sent. To use a custom list, the selection element must be set to other.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>This element can have two values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- default uses the e-mail address configured in the reporting_address element of the https://&lt;IP-address-of-SPS&gt;/api/configuration/management/email endpoint (or the Basic Settings &gt; Management &gt; Mail settings &gt; Send reports to field on the web UI).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- other uses the e-mails listed in the recipients element.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the report.</td>
</tr>
<tr>
<td>send_report_in_email</td>
<td>boolean</td>
<td>Set it to false if you do not want to include the generated report in the e-mail.</td>
</tr>
</tbody>
</table>

**Examples:**

Set the e-mail recipients to the default (as configured in the reporting_address element of the /api/configuration/management/email endpoint):

```
"email_recipients": {  
    "selection": "default"  
}
```

Create a custom set of e-mail recipients:

```
"email_recipients": {  
    "recipients": [  
        "<email-1>",  
        "<email-2>"  
    ],  
    "selection": "other"  
}
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>400</td>
<td>IncompleteConfigurationSubtreeError</td>
<td>Do not provide recipients if you set the selection element under email_recipients to default.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>Error when committing your transaction. Creating new pre-defined reports is not allowed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Modify a pre-defined report**

To modify a report, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the report.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/reporting/predefined_reports/<report-key> endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.
Content subchapters

Reporting subchapters created from audit trail content (statistics of search keywords, and screenshots). You have to provide a list of keywords, and create the appropriate filters to narrow down the scope of the search. SPS searches the indexed content of all audit trails that fit the filter criteria, and provide the resulting statistics and screenshots in the report.

Configure and enable indexing for all connections that you want to include in the reports.

URL

GET https://<IP-address-of-SPS>/api/configuration/reporting/content_subchapters

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available content subchapters.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/content_subchapters
```

The following command retrieves the properties of a specific subchapter.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/content_subchapters/<subchapter-key>
```

Response

The following is a sample response received when listing content subchapters.
For details of the meta object, see Message format on page 10.

```json
{
  "meta": {
    "first": "/api/configuration/reporting/content_subchapters",
    "href": "/api/configuration/reporting/content_subchapters",
    "last": "/api/configuration/reporting/statistics_subchapters",
    "next": "/api/configuration/reporting/custom_subchapters",
    "parent": "/api/configuration/reporting",
    "previous": null,
    "transaction": "/api/transaction"
  },
  "items": [
    {
      "key": "13869311625707e0a3e0892",
      "meta": {
        "href": "/api/configuration/reporting/content_subchapters/13869311625707e0a3e0892"
      }
    }
  ]
}
```

When retrieving the endpoint of a specific content subchapter, the response is the following.

```json
{
  "body": {
    "access": [
      "search"
    ],
    "filter": {
      "channel_policy": {
        "key": "+10200",
        "meta": {
          "href": "/api/configuration/ssh/channel_policies/+10200"
        }
      },
      "connection_policy": "8348340645707e2575e3c6",
      "protocol": "ssh",
      "server_address": "192.168.56.102",
      "server_port": 22,
      "source_address": "192.168.56.101",
      "source_port": 22,
      "username": "admin"
    },
    "name": "API_test_subchapter",
    "search_words": [
      "logout"
    ]
  }
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the subchapter.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the subchapter.</td>
</tr>
<tr>
<td>access</td>
<td>list</td>
<td>Required. List of access control groups whose members can access the subchapter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To deny access to the subchapter, use &quot;admin&quot; as the only value for the element.</td>
</tr>
<tr>
<td>filter</td>
<td>Top level element.</td>
<td>Filter options for narrowing the scope of the keyword search. See the corresponding table for more details.</td>
</tr>
<tr>
<td>channel_policy</td>
<td>string</td>
<td>References the key of the channel policy. You can configure channel policies at the &quot;/api/configuration/&lt;protocol&gt;/channel_policies/&lt;policy-ID&gt;&quot; endpoint.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that the path is different for each protocol.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To modify or add a channel policy, use the value of the returned key as the value of the channel_policy element, and remove any child elements (including the key).</td>
</tr>
<tr>
<td>connection_policy</td>
<td>string</td>
<td>The key of the connection policy specified for the search.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>The protocol of the connection or channel policy specified for the search.</td>
</tr>
<tr>
<td>server_address</td>
<td>string</td>
<td>The target server's address. Use an IPv4 address.</td>
</tr>
<tr>
<td>server_port</td>
<td>int</td>
<td>The port of the target server's address.</td>
</tr>
<tr>
<td>source_address</td>
<td>string</td>
<td>The address from where the connection is initiated.</td>
</tr>
<tr>
<td>source_port</td>
<td>int</td>
<td>The port of the address from where the connection is initiated.</td>
</tr>
<tr>
<td>username</td>
<td>string</td>
<td>The username used to connect to the target server.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the subchapter.</td>
</tr>
<tr>
<td>search_words</td>
<td>list</td>
<td>The list of search keywords to generate statistics and screenshots for in the subchapter.</td>
</tr>
</tbody>
</table>

**Examples:**

Create a content subchapter for the occurrences of the "logout" keyword in SSH connections. Make the subchapter accessible to the search and report usergroups.

- Search connections where the "shell-only" channel policy is used.

```json
{
    "access": [
        "search",
        "report"
    ],
    "filter": {
        "channel_policy": "-10000",
        "connection_policy": null,
        "protocol": "ssh",
        "server_address": null,
        "server_port": null,
        "source_address": null,
        "source_port": null,
        "username": null
    }
}  ```
Search connections of a specific connection policy. Provide the protocol of the connection. The key of the connection policy is available at the /api/configuration/<protocol>/connections/ endpoint.

```json
{
  "access": [
    "search",
    "report"
  ],
  "filter": {
    "channel_policy": null,
    "connection_policy": "<key-of-connection-policy>",
    "protocol": "ssh",
    "server_address": null,
    "server_port": null,
    "source_address": null,
    "source_port": null,
    "username": null
  },
  "name": "Controlled_access",
  "search_words": [
    "logout"
  ]
}
```

Search connections where the "admin" username was used.

```json
{
  "access": [
    "search",
    "report"
  ],
  "filter": {
    "channel_policy": null,
    "connection_policy": null,
    "protocol": "ssh",
    "server_address": null,
    "server_port": null,
    "source_address": null,
    "source_port": null,
    "username": "admin"
  }
}
```
Search connections made to a specific server address and port.

```json
{
  "name": "Server_access",
  "search_words": [
    "logout"
  ]
}
```

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>400</td>
<td>Path: &lt;endpoint&gt;/filter/channel_policy</td>
<td>You have included the key and meta elements of a channel_policy in a PUT or POST request.</td>
</tr>
<tr>
<td></td>
<td>Type: SyntacticError</td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

**Add a content subchapter**

To add a content subchapter, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new content subchapter.**

   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/reporting/content_subchapters/` endpoint. You can find a detailed description of the available parameters listed in Element.
   
   - To use a channel policy for filtering, use the key of the policy. You must also set the `protocol` element to the corresponding protocol.
     
     For example, to use the shell-only channel policy, which is a default SSH policy provided by SPS, you have to configure both the `channel_policy` element...
     
     ```json
     "channel_policy": "-10000"
     ```
     
     ...and the `protocol` element:
     
     ```json
     "protocol": "ssh"
     ```

   If the POST request is successful, the response includes the key of the new subchapter. For example:
3. **Commit your changes.**

   For more information, see *Commit a transaction* on page 31.

**Modify a content subchapter**

To modify a content subchapter, you have to:

1. **Open a transaction.**

   For more information, see *Open a transaction* on page 29.

2. **Modify the JSON object of the subchapter.**

   You can find a detailed description of the available parameters listed in [Element](#).

   To use a channel policy for filtering, do not include the returned key and meta elements of the channel policy in your PUT request. Instead, set the value of the channel_policy to the value of its key.

   For example, if a GET request for the subchapter returns the following channel_policy filter:

   ```json
   "channel_policy": {
     "key": "-10200",
     "meta": {
       "href": "/api/configuration/ssh/channel_policies/-10200"
     }
   }
   ``

   You have to change it in your PUT request to:

   ```json
   "channel_policy": "-10200"
   ```

   You must also configure the protocol element to the protocol of the channel policy.
3. **Upload the modified configuration**

   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/reporting/content_subchapters/<subchapter-key> endpoint.

4. **Commit your changes.**

   For more information, see [Commit a transaction](#) on page 31.

## Custom subchapters

List of the reporting subchapters created from custom queries to the SPS connection database. The list of tables and fields you can query are described in "Database tables available for custom queries" in the Administration Guide.

### URL

GET https://<IP-address-of-SPS>/api/configuration/reporting/custom_subchapters

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API</a> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

### Sample request

The following command lists the available custom subchapters.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/custom_subchapters
```
The following command retrieves the properties of a specific subchapter.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/custom_subchapters/<object-id>

**Response**

The following is a sample response received when listing custom subchapters.

For details of the `meta` object, see Message format on page 10.

```json
{
  "meta": {
    "first": "/api/configuration/reporting/content_subchapters",
    "href": "/api/configuration/reporting/custom_subchapters",
    "last": "/api/configuration/reporting/statistics_subchapters",
    "next": "/api/configuration/reporting/predefined_reports",
    "parent": "/api/configuration/reporting",
    "previous": "/api/configuration/reporting/content_subchapters",
    "transaction": "/api/transaction"
  },
  "items": [
    {
      "key": "5983143445707eb740fee3",
      "meta": {
        "href": "/api/configuration/reporting/custom_subchapters/5983143445707eb740fee3"
      }
    }
  ]
}
```

When retrieving the endpoint of a specific subchapter, the response is the following.

```json
{
  "body": {
    "access": [
      "search"
    ],
    "chart": {
      "column_titles": [
        "col1",
        "col2"
      ],
      "type": "list"
    },
    "name": "API_test_adv_stats",
    "query": "select\n      to_timestamp(audit_trail_downloads.download_time),
      audit_trail_downloads.username,
      channels.channel_type,
      ..."
  }
}
```
```
channels.connection, 
 audit_trail_downloads.ip
from audit_trail_downloads, 
 channels
where channels.connection_channel_id = audit_trail_downloads.id
and audit_trail_downloads.download_time <= :range_start
and audit_trail_downloads.download_time > :range_end
and audit_trail_downloads.username != 'admin'
order by audit_trail_downloads.download_time;
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the custom subchapter.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the custom subchapter.</td>
</tr>
<tr>
<td>access</td>
<td>list</td>
<td>Required. List of access control groups whose members can access the subchapter.</td>
</tr>
<tr>
<td>chart type</td>
<td>string</td>
<td>Defines the chart type.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Use bar to generate a bar chart.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You have to provide the y_axis_title element for bar charts (it can be null).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use pie to generate pie a chart.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use list to generate a list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You have to provide the column_titles element for lists (it can be null).</td>
</tr>
</tbody>
</table>
### Element Table

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y_axis_title</td>
<td>string</td>
<td>Required if the type element is set to bar. The name of the y axis for the generated bar chart.</td>
</tr>
<tr>
<td>column_titles</td>
<td>list</td>
<td>Required if the type element is set to list. The column titles for the generated list.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the subchapter.</td>
</tr>
<tr>
<td>query</td>
<td>string</td>
<td>The SQL database query for creating the subchapter.</td>
</tr>
</tbody>
</table>

⚠️ **CAUTION:**

Generating a report that includes an Advanced statistics chapter that returns several thousands of entries requires significant CPU and memory resources from One Identity Safeguard for Privileged Sessions (SPS). While generating such a partial report, the web interface of SPS can become slow or unresponsive.

### Examples:

Create a bar chart with a custom title for the y-axis:

```json
"chart": {
  "type": "bar",
  "y_axis_title": "Y_axis"
}
```

Create a pie chart:

```json
"chart": {
  "type": "pie"
}
```

Create a list with custom column names:

```json
"chart": {
  "column_titles": [
```

```json
```
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Add a custom subchapter

To add a custom subchapter, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new subchapter.**
   
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/reporting/custom_subchapters endpoint. You can find a detailed description of the available parameters listed in Element. If the POST request is successful, the response includes the key of the new subchapter. For example:
3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**Modify a custom subchapter**

To modify a subchapter, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the subchapter.**
   
   PUT the modified JSON object to the https://IP-address-of-SPS/api/configuration/reporting/custom_subchapters/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in Element.

1. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**Connection statistics subchapters**

List of the reporting subchapters created from connection statistics.

**URL**

GET https://IP-address-of-SPS/api/configuration/reporting/statistics_subchapters
Cookies

<table>
<thead>
<tr>
<th>Cookie Name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following request command lists the available subchapters.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/statistics_subchapters
```

The following command retrieves the properties of a specific subchapter.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/reporting/statistics_subchapters/<subchapter-id>
```

Response

The following is a sample response received when listing connection statistics subchapters. For details of the meta object, see Message format on page 10.

```json
{
   "meta": {
      "first": "/api/configuration/reporting/content_subchapters",
      "href": "/api/configuration/reporting/statistics_subchapters",
      "last": "/api/configuration/reporting/statistics_subchapters",
      "next": null,
      "parent": "/api/configuration/reporting",
      "previous": "/api/configuration/reporting/reports",
      "transaction": "/api/transaction"
   },
   "items": [
```

SPS 6.13.0 REST API Reference Guide

Reporting

815

ONE IDENTITY by Quest
When retrieving the endpoint of a specific subchapter, the response is the following.

```json
{
    "body": {
        "access": [
            "search",
            "reporting"
        ],
        "chart": {
            "type": "list"
        },
        "name": "stats_simple",
        "query": {
            "column": "username",
            "filter": [
                {
                    "column": "protocol",
                    "is_exact": false,
                    "is_inverted": false,
                    "value": "ssh"
                },
                {
                    "column": "username",
                    "is_exact": false,
                    "is_inverted": false,
                    "value": "admin"
                }
            ],
            "limit": 15,
            "order": "top"
        }
    },
    "key": "496444806570e9c7e32c30",
    "meta": {
        "first": "/api/configuration/reporting/statistics_subchapters/21111736175707f1df8bea1",
        "href": "/api/configuration/reporting/statistics_subchapters/496444806570e9c7e32c30",
        "last": "/api/configuration/reporting/statistics_subchapters/496444806570e9c7e32c30"
    }
}
```
"next": null,
"parent": "/api/configuration/reporting/statistics_subchapters",
"previous": "/api/configuration/reporting/statistics_subchapters/1539306268570e9442cab6c",
"transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the subchapter.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The elements of the subchapter.</td>
</tr>
<tr>
<td>access</td>
<td>list</td>
<td>Required. List of access control groups whose members can access the subchapter. To deny access to the subchapter, use &quot;admin&quot; as the only value for the element.</td>
</tr>
<tr>
<td>chart</td>
<td>Top level element</td>
<td>Defines the properties of the chart generated from the database query.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Defines the chart type.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the subchapter.</td>
</tr>
<tr>
<td>query</td>
<td>string</td>
<td>The search query that defines the connections to use for creating statistics. For details on using the search, see Searching in the session database on page 717.</td>
</tr>
</tbody>
</table>

**Examples:**
Create statistics about the 15 most common usernames used in SSH connections.

- Create a bar chart.

```json
{
   "access": [
      "reporting",
      "search"
   ],
```
Create a bar chart.

```
"chart": {
    "type": "bar"
},
"name": "stats_bar",
"query": {
    "column": "username",
    "filter": [
        {
            "column": "protocol",
            "is_exact": false,
            "is_inverted": false,
            "value": "ssh"
        }
    ],
    "limit": 15,
    "order": "top"
}
```

Create a pie chart.

```
{
    "access": [
        "reporting",
        "search"
    ],
    "chart": {
        "type": "pie"
    },
    "name": "stats_pie",
    "query": {
        "column": "username",
        "filter": [
            {
                "column": "protocol",
                "is_exact": false,
                "is_inverted": false,
                "value": "ssh"
            }
        ],
        "limit": 15,
        "order": "top"
    }
}
```

Create a list.
{  
  "access": [  
    "reporting",
    "search"
  ],  
  "chart": {  
    "type": "list"
  },  
  "name": "stats_list",
  "query": {  
    "column": "username",
    "filter": [  
      {  
        "column": "protocol",
        "is_exact": false,
        "is_inverted": false,
        "value": "ssh"
      }
    ],  
    "limit": 15,
    "order": "top"
  }
}

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Add a connection statistics subchapter

To add a connection statistics subchapter, you have to:

1. **Open a transaction.**
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new subchapter.**
   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/reporting/statistics_subchapters/` endpoint. You can find a detailed description of the available parameters listed in [Element](#).
   If the POST request is successful, the response includes the key of the new subchapter. For example:
   ```json
   {
     "key": "769e627d-515d-4d26-a03e-cb2ed0bb0e04",
     "meta": {
       "href": "https://<IP-address-of-SPS>/api/configuration/reporting/statistics_subchapters/769e627d-515d-4d26-a03e-cb2ed0bb0e04",
       "parent": "https://<IP-address-of-SPS>/api/configuration/reporting/statistics_subchapters",
       "transaction": "https://<IP-address-of-SPS>/api/transaction"
     }
   }
   ```

3. **Commit your changes.**
   For more information, see [Commit a transaction](#) on page 31.

Modify a connection statistics subchapter

To modify a subchapter, you have to:

1. **Open a transaction.**
   For more information, see [Open a transaction](#) on page 29.

2. **Modify the JSON object of the subchapter.**
   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/reporting/statistics_subchapters//<key-of-the-subchapter>` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

3. **Commit your changes.**
   For more information, see [Commit a transaction](#) on page 31.
Health and maintenance

Monitor appliance health status

To monitor the health status of an appliance, query the /api/health-status endpoint.

```shell
curl --cookie cookies https://<IP-address-of-SPS>/api/health-status
```

The following is a sample response received.

For details of the meta object, see Message format on page 10.

For details of the other objects, see tables Cluster status details and "issues" object details.

```
{
    "health_status": {
        "cpu": 5.4,
        "disk": 10.3,
        "load1": 1.93,
        "load15": 1.98,
        "load5": 2.01,
        "memory": 46.3,
        "sessions": {
            "http": 0,
            "ica": 0,
            "mssql": 0,
            "rdp": 0,
            "ssh": 0,
            "telnet": 0,
            "vnc": 0
        },
        "sessions_total": 0,
        "swap": 0,
        "system_details": {
            "cpu": {
                "guest": 0.0,
                "guest_nice": 0.0,
                "idle": 94.6,
                "iowait": 0.0,
                "irq": 0.0,
                "nice": 0.5,
                "softirq": 0.0,
                "steal": 1.0,
                "system": 1.0,
                "user": 3.0
            }
        }
    }
}
```
"disk": {
    "free": 26850131968,
    "percent": 10.3,
    "total": 31571550208,
    "used": 3094085632
},
"memory": {
    "active": 4459466752,
    "available": 4492849152,
    "buffers": 456245248,
    "cached": 3229765632,
    "free": 1336004608,
    "inactive": 1984532480,
    "percent": 46.3,
    "shared": 249368576,
    "total": 8364044288,
    "used": 3342028800
},
"swap": {
    "free": 0,
    "percent": 0,
    "sin": 0,
    "sout": 0,
    "total": 0,
    "used": 0
},
"pipeline-queues": {
    "session-events": 64.3,
    "indexer-events": 40.0
}
},
"meta": {
    "href": "/api/health-status",
    "parent": "/api",
    "remaining_seconds": 600
}

<table>
<thead>
<tr>
<th>Elements</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>health_status</td>
<td>null or object</td>
<td>The health status of a node. When queried, it lists data related to the given node's health (in the case of HA, this means the current master node).</td>
</tr>
<tr>
<td>memory</td>
<td>floating point number</td>
<td>Memory usage (percent)</td>
</tr>
</tbody>
</table>
## Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>disk</td>
<td>floating point number</td>
<td>Hard disk usage (percent)</td>
</tr>
<tr>
<td>swap</td>
<td>floating point number</td>
<td>Swap usage (percent)</td>
</tr>
<tr>
<td>cpu</td>
<td>floating point number</td>
<td>Overall CPU usage (percent)</td>
</tr>
<tr>
<td>load1</td>
<td>floating point number</td>
<td>The average system load during the last one minute.</td>
</tr>
<tr>
<td>load5</td>
<td>floating point number</td>
<td>The average system load during the last five-minute period.</td>
</tr>
<tr>
<td>load15</td>
<td>floating point number</td>
<td>The average system load during the last fifteen-minute period.</td>
</tr>
<tr>
<td>sessions</td>
<td>string</td>
<td>The protocol type and the number of ongoing sessions. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sessions&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ssh&quot;: 3,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;rdp&quot;: 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td>total_sessions</td>
<td>integer (number of)</td>
<td>The total number of ongoing sessions.</td>
</tr>
<tr>
<td>system_details</td>
<td>JSON object</td>
<td>Various details about the CPU, disk, memory and swap usage of the appliance. Note that the exact set of metrics is determined by the underlying kernel and system libraries, therefore it might change between different versions of Safeguard for Privileged Sessions without notice.</td>
</tr>
<tr>
<td>pipeline-queues</td>
<td>JSON object</td>
<td>Represents the fullness of the processing pipelines in percentages.</td>
</tr>
</tbody>
</table>

The number of CPUs determine the load a system can handle without causing the processes having to wait. As a generic rule of thumb, if the load is less than the number of processor cores of the appliance, the overall system load can be considered normal, otherwise it might be an indication of performance issues.
Advanced authentication and authorization

Usermapping policy

For SSH, RDP, Telnet, and Citrix ICA connections, usermapping policies can be defined. A usermapping policy describes who can use a specific username to access the remote server: only members of the specified local or LDAP usergroups (for example, administrators) can use the specified username (for example, root) on the server.

URL

GET https://<IP-address-of-SPS>/api/configuration/policies/usermapping_policies

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command lists the existing usermapping policies.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/usermapping_policies

The following command retrieves the properties of a specific usermapping policy.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/usermapping_policies<object-id>

Response

The following is a sample response received when listing usermapping policies.

For details of the meta object, see Message format on page 10.

```
{
  "meta": {
    "first": "/api/configuration/policies/audit_policies",
    "href": "/api/configuration/policies/usermapping_policies",
    "last": "/api/configuration/policies/usermapping_policies",
    "next": null,
    "parent": "/api/configuration/policies",
    "previous": "/api/configuration/policies/userlists",
    "transaction": "/api/transaction"
  },
  "items": [
    {
      "key": "11581153055704544883f77",
      "meta": {
        "href": "/api/configuration/policies/usermapping_policies/11581153055704544883f77"
      }
    },
    {
      "key": "9328731525704545f5e3de",
      "meta": {
        "href": "/api/configuration/policies/usermapping_policies/9328731525704545f5e3de"
      }
    }
  ]
}
```

When retrieving the endpoint of a specific host key, the response is the following.
Element | Type | Description
--- | --- | ---
key | string | Top level element, contains the ID of the policy.

body | Top level element (string) | The elements of the usermapping policy.

allow_other_remote_users_without_mapping | boolean | Default value: true. To allow access the remote servers for users who are not explicitly listed in the Usermapping Policy, configure true. Note that these users must use the same username on the SPS gateway and the SPS 6.13.0 REST API Reference Guide Advanced authentication and authorization 826
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappings</td>
<td>Top level list</td>
<td>Contains the list of user groups and the corresponding remote usernames the group members can use to log in.</td>
</tr>
<tr>
<td>allowed_groups</td>
<td>list</td>
<td>The usergroups allowed to log in as the remote_user on the remote server. Required element. Empty means all users.</td>
</tr>
<tr>
<td>remote_user</td>
<td>string</td>
<td>The username on the remote server that the users configured in allowed_groups can use to log in. Required element. Must have a value.</td>
</tr>
</tbody>
</table>

**Example mappings:**

Anyone can log in to the remote server as the test user:

```
"mappings": [  
  {  
    "allowed_groups": [],  
    "remote_user": "test"  
  }
```

Only the members of the admin group can log in to the remote server as the root user:

```
"mappings": [  
  {  
    "allowed_groups": [  
      "admins"
    ],  
    "remote_user": "root"
  }
```

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidQuery</td>
<td>The requested filter or its value is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the</td>
</tr>
</tbody>
</table>

SPS 6.13.0 REST API Reference Guide
Advanced authentication and authorization
### Code Description Notes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

#### Add a usermapping policy

To add a usermapping policy, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.

2. **Create the JSON object for the new usermapping policy.**
   
   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/policies/usermapping` endpoint. You can find a detailed description of the available parameters listed in [Element](#).

   If the POST request is successful, the response includes the key of the new usermapping policy. For example:

   ```json
   {
     "key": "2e8692fa-7fda-4753-8363-37e8244f6b80",
     "meta": {
       "href": "/api/configuration/policies/usermapping_policies/2e8692fa-7fda-4753-8363-37e8244f6b80",
       "parent": "/api/configuration/policies/usermapping_policies",
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes.**
   
   For more information, see [Commit a transaction](#) on page 31.

#### Modify a usermapping policy

To modify a usermapping policy, you have to:
1. **Open a transaction.**
   For more information, see *Open a transaction* on page 29.

2. **Modify the JSON object of the usermapping policy.**
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/usermapping/<key-of-the-object> endpoint. You can find a detailed description of the available parameters listed in *Element*.

3. **Commit your changes.**
   For more information, see *Commit a transaction* on page 31.

## Plugins

Contains the endpoints for configuring plugins.

### URL

**GET** https://<IP-address-of-SPS>/api/configuration/plugins

### Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <em>Authenticate to the SPS REST API</em> on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format). 

### Sample request

The following command lists endpoints for configuring plugins.
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/plugins

Response

The following is a sample response received when listing endpoints for configuring plugins. For details of the meta object, see Message format on page 10.

```json
{
  "items": [
    {
      "key": "aa",
      "meta": {
        "href": "/api/configuration/plugins/aa"
      }
    },
    {
      "key": "configuration_sync",
      "meta": {
        "href": "/api/configuration/plugins/configuration_sync"
      }
    },
    {
      "key": "credentialstore",
      "meta": {
        "href": "/api/configuration/plugins/credentialstore"
      }
    },
    {
      "key": "signingca",
      "meta": {
        "href": "/api/configuration/plugins/signingca"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa",
    "href": "/api/configuration/plugins",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/policies",
    "parent": "/api/configuration",
    "previous": "/api/configuration/passwords",
    "remaining_seconds": 600,
    "transaction": "/api/transaction"
  }
}
```
### Element Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa</td>
<td>Endpoint for configuring authentication and authorization plugins.</td>
</tr>
<tr>
<td>configuration_</td>
<td>Endpoint for configuring plugins that synchronize the configuration of SPS</td>
</tr>
<tr>
<td>sync</td>
<td>clusters that receive their configuration from the Central Management node.</td>
</tr>
<tr>
<td>credentialstore</td>
<td>Endpoint for configuring credential store plugins.</td>
</tr>
<tr>
<td>signingca</td>
<td>Endpoint for configuring plugins to sign certificates.</td>
</tr>
</tbody>
</table>

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

### Upload a plugin

To upload or update a plugin, complete the following steps. To update a plugin, upload a new version. Starting with version 6.4, you can also delete plugins using the REST API. For details, see Delete a plugin.

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Upload a plugin**
   
   POST the plugin as a zip file (application/zip) to the `https://<IP-address-of-SPS>/api/upload/plugins` endpoint, for example:
curl -X POST -H "Content-Type: application/zip" --cookie cookies https://<IP-address-of-SPS>/api/upload/plugins --data-binary @<path-to-plugin.zip>

If the POST request is successful, the response includes the key of the new plugin, as well as information about the uploaded plugin. For example:

```json
{
    "meta": {
        "href": "/api/configuration/plugins/aa/aa423b72-0d0f-4275-be30-494e9a9ffad",
        "parent": "/api/configuration/plugins/aa"
    },
    "key": "aa423b72-0d0f-4275-be30-494e9a9ffad",
    "body": {
        "name": "Sample-Authentication-Plugin",
        "description": "My custom authentication plugin",
        "version": "1.12",
        "path": "/opt/scb/var/plugins/aa/Sample-Authentication-Plugin",
        "api": "1.0"
    }
}
```

3. **Commit your changes.**

For more information, see Commit a transaction on page 31.

Note the following points.

- Re-uploading an already existing plugin overwrites the existing plugin.
- Uploading a newer version of an already existing plugin overwrites the existing plugin.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The plugin has been successfully uploaded. The response should include the key of the created object.</td>
</tr>
<tr>
<td>400</td>
<td>The plugin does not support this version of SPS.</td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>InvalidPlugin</td>
<td>The type or some other value in the Manifest file of the plugin is invalid, or this version of SPS is not supported.</td>
</tr>
</tbody>
</table>
### Delete a plugin

Starting with version 6.4, you can also delete plugins using the REST API.

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Delete a plugin**
   
   DELETE the https://<IP-address-of-SPS>/api/configuration/plugins/aa/<ID-of-the-plugin-to-delete> endpoint. For details, see Delete an object on page 43. If the DELETE request is successful, the response includes only the meta object, for example:
   
   ```json
   {
     "meta": {
       "href": "/api/configuration/plugins/aa/b080b1ba546232548bb1a9",
       "parent": "/api/configuration/plugins/aa"
     }
   }
   ```

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>422</td>
<td>TransactionProcessingError</td>
<td>The plugin was uploaded but deploying the plugin failed for some reason.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>200</td>
<td></td>
<td>The plugin has been successfully deleted</td>
</tr>
<tr>
<td>400</td>
<td>SemanticError</td>
<td>The plugin cannot be deleted, because there is reference to it in the configuration (For example, AA plugin delete fails because there is an AA Plugin Configuration for it).</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td></td>
<td>There is no plugin with the given key.</td>
</tr>
</tbody>
</table>

### Check the integrity of a plugin

The authentication and authorization (AA) plugins used on SPS. To upload or update a plugin, see [Upload a plugin](#).

**URL**

```
GET https://<IP-address-of-SPS>/api/plugin/integrity?key=<key-value-from-the-response-of-the-last-creation>&plugin_type=<type-of-the-plugin>&ops=zip_checksum&ops=zip_content&ops=unregistered
```

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API on page 19</a>.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request

The following command retrieves the results of the integrity check.

curl --cookie cookies https://<IP-address-of-SPS>/api/plugin/integrity?key=<key-value-from-the-response-of-the-last-creation>&plugin_type=aa&ops=zip_checksum&ops=zip_content&ops=unregistered

- To retrieve the <key-value-from-the-response-of-the-last-creation> of the plugin that you have uploaded earlier, enter the following command:

curl https://<IP-address-of-SPS>/api/configuration/plugins/<plugin_type>

This will display all plugins that you have uploaded earlier, that belong to the specified plugin type. The value will be the value of the key parameter of the response.

- The following plugin_type values are available:
  - Authentication and authorization: aa
  - Configuration synchronization: configuration_sync
  - Credential Store: credentialstore
  - Signing CA: signingca

Response

The following is a sample response received when querying the results of the integrity check.

For details of the meta object, see Message format on page 10.

```json
{
  "body": {
    "zip_checksum": {
      "verdict": "passed",
      "reason": "Plugin .zip checksums match"
    },
    "zip_content": {
      "verdict": "passed",
      "reason": "The plugin runtime files are the same since you have uploaded the plugin .zip"
    },
    "unregistered": {
      "verdict": "unavailable",
      "reason": "Cannot find checker. Make sure that you use an existing checker: unregistered"
    }
  }
}
```
### Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>MissingMandatoryParameter</td>
<td>One of the following keys is missing: key, plugin_type, ops.</td>
</tr>
<tr>
<td>400</td>
<td>InvalidFormat</td>
<td>The key is not valid plugin key.</td>
</tr>
<tr>
<td>404</td>
<td>MissingPlugin</td>
<td>The plugin is not found in the configuration.</td>
</tr>
</tbody>
</table>

### Authentication and authorization plugins

The authentication and authorization (AA) plugins used on SPS. To upload or update a plugin, see Upload a plugin.
URL

GET https://<IP-address-of-SPS>/api/configuration/plugins/aa

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d830e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command queries the list of AA plugins used on SPS.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/plugins/aa
```

The following command retrieves the properties of a specific plugin.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/plugins/aa/<plugin-id>
```

Response

The following is a sample response received when querying the list of AAA plugins used on SPS.

For details of the meta object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "2080160955734bb2a1ddf9",
            "meta": {
                "href": "/api/configuration/plugins/aa/2080160955734bb2a1ddf9"
            }
        }
    ]
}
```
When retrieving the endpoint of a specific plugin, the response is the following.

```
{
    "body": {
        "api": "1.0",
        "description": "test1",
        "name": "AAPluginExample",
        "version": "1.1",
        "path": "/opt/scb/var/plugins/customgwauthplugin",
        "scb_max_version": "",
        "scb_min_version": "",
        "default_configuration": "",
        "entry_point": null,
        "sha256sum": "c4bb901de6b2274dcb94f1eec429fd0f3565ac792a856b07b8895e56ca2d8f42"
    },
    "key": "208016095734bb2a1ddf9",
    "meta": {
        "first": "/api/configuration/plugins/aa/208016095734bb2a1ddf9",
        "href": "/api/configuration/plugins/aa/208016095734bb2a1ddf9",
        "last": "/api/configuration/plugins/aa/208016095734bb2a1ddf9",
        "next": null,
        "parent": "/api/configuration/plugins/aa",
        "previous": null,
        "transaction": "/api/transaction"
    }
}
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the plugin.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the properties of the plugin.</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>api</td>
<td>string</td>
<td>The API version of the plugin.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the plugin. This description is also displayed on the SPS web interface.</td>
</tr>
<tr>
<td>default_configuration</td>
<td>string</td>
<td>The default configuration of the plugin (an INI file as a string). For details, see the documentation of the particular plugin.</td>
</tr>
<tr>
<td>entry_point</td>
<td>string</td>
<td>The entry point of the plugin, for example, main.py</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the plugin. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>The path where the plugin is stored on SPS.</td>
</tr>
<tr>
<td>scb_max_version</td>
<td>string</td>
<td>The version number of the latest SPS release that is compatible with the plugin.</td>
</tr>
<tr>
<td>scb_min_version</td>
<td>string</td>
<td>The version number of the earliest SPS release that is compatible with the plugin.</td>
</tr>
<tr>
<td>sha256sum</td>
<td>string</td>
<td>The SHA-256 checksum of the plugin.</td>
</tr>
<tr>
<td>version</td>
<td>string</td>
<td>The version number of the plugin.</td>
</tr>
</tbody>
</table>

To configure a particular instance of a plugin, use the /api/configuration/policies/aa_plugin_instances/<key-of-the-plugin-instance> endpoint.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Configuring Authentication and Authorization plugin instances

You can configure instances of Authentication and Authorization (AA) plugins to use in your Connection Policies. To configure an instance of a plugin you must first upload the plugin to SPS. To upload or update a plugin, see Upload a plugin.

**URL**

GET https://<IP-address-of-SPS>/api/configuration/policies/aa_plugin_instances

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

**Sample request**

The following command queries the list of AA plugin instances available on SPS.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/aa_plugin_instances
```

The following command retrieves the properties of a specific instance.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/aa_plugin_instances/<plugin-id>
```

**Response**

The following is a sample response received when querying the list of AA plugins used on SPS.
For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "name": "new_plugin_instance",
            "configuration": "test configuration",
            "plugin": "8876228625d67aa91e2253"
        }
    ],
    "meta": {
        "first": "/api/configuration/policies/aa_plugin_instances",
        "href": "/api/configuration/policies/aa_plugin_instances",
        "last": "/api/configuration/policies/usermapping_policies",
        "next": "/api/configuration/policies/analytics",
        "parent": "/api/configuration/policies",
        "previous": null,
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
    }
}
```

When retrieving the endpoint of a specific plugin instance, the response is the following.

```json
{
    "body": {
        "configuration": "[starling]\n# ..... disable_echo=yes\n",
        "name": "Demo_starling_plugin",
        "plugin": {
            "key": "8876228625d67aa91e2253",
            "meta": {
                "href": "/api/configuration/plugins/aa/8876228625d67aa91e2253"
            }
        }
    },
    "key": "8114402005d67adbeb38b6",
    "meta": {
        "first": "/api/configuration/policies/aa_plugin_instances/8114402005d67adbeb38b6",
        "href": "/api/configuration/policies/aa_plugin_instances/8114402005d67adbeb38b6",
        "last": "/api/configuration/policies/aa_plugin_instances/8114402005d67adbeb38b6",
        "next": null,
        "parent": "/api/configuration/policies/aa_plugin_instances",
```

SPS 6.13.0 REST API Reference Guide
Advanced authentication and authorization
Create a new plugin instance

To create a new instance of a plugin, you have to:

1. **Open a transaction.**
   
   For more information, see [Open a transaction](#) on page 29.
2. **Create the JSON object of the plugin instance.**
   
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/aa_plugin_instances endpoint. You can find a detailed description of the available parameters listed in Configuring Authentication and Authorization plugin instances.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>SemanticError</td>
<td>The configuration of the instance is invalid. Check the error key in the response for details.</td>
</tr>
</tbody>
</table>

**Modify a plugin instance**

To modify an instance of a plugin, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the policy.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/aa_plugin_instances/<key-of-the-instance> endpoint.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

**Delete a plugin instance**

To delete an instance of a plugin, you have to:

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.
2. Remove any references to the plugin instance from your Connection Policies. You cannot delete a plugin instance that other parts of the configuration actively use.

3. **Delete the endpoint of the plugin instance.**

   DELETE the https://<IP-address-of-SPS>/api/configuration/policies/aa_plugin_instances/<key-of-the-instance> endpoint.

4. **Commit your changes.**

   For more information, see Commit a transaction on page 31.

**Credential store plugins**

The credential store plugins used on SPS. To upload or update a plugin, see Upload a plugin.

**URL**

| GET https://<IP-address-of-SPS>/api/configuration/plugins/credentialstore |

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).|

**Sample request**

The following command lists the credential store plugins stored on SPS.
The following command retrieves the properties of a specific plugin.

```bash
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/plugins/credentialstore/plugins/credentialstore/<plugin-id>
```

Response

The following is a sample response received when listing the credential store plugins used on SPS.

For details of the meta object, see Message format on page 10.

```json
{
   "items": [
   {
      "key": "2534221015734bb18aaf32",
      "meta": {
         "href": "/api/configuration/plugins/credentialstore/2534221015734bb18aaf32"
      }
   },
   
   "meta": {
      "first": "/api/configuration/plugins/aa",
      "href": "/api/configuration/plugins/credentialstore",
      "last": "/api/configuration/plugins/ticketing",
      "next": "/api/configuration/plugins/ticketing",
      "parent": "/api/configuration/plugins",
      "previous": "/api/configuration/plugins/aa",
      "transaction": "/api/transaction"
   }
}
```

When retrieving the endpoint of a specific plugin, the response is the following.

```json
{
   "body": {
      "api": "1.0",
      "description": "Demo credentialstore plugin for demonstration purposes",
      "name": "DemoCredentialStorePlugin",
      "path": "/opt/scb/var/plugins/credentialstore/DemoCredentialStorePlugin",
      "version": "1.1",
      "scb_max_version": "",
      "scb_min_version": "",
      "default_configuration": "",
      "entry_point": null,
```
```json
{"sha256sum": "c4bb901de6b2274dcb94f1eec429fd0f3565ac792a856b07b8895e56ca2d8f42",
"key": "2534221015734bb18aaf32",
"meta": {
  "first": "/api/configuration/plugins/credentialstore/2534221015734bb18aaf32",
  "href": "/api/configuration/plugins/credentialstore/2534221015734bb18aaf32",
  "last": "/api/configuration/plugins/credentialstore/2534221015734bb18aaf32",
  "next": null,
  "parent": "/api/configuration/plugins/credentialstore",
  "previous": null,
  "transaction": "/api/transaction"
}
}
```

### Element Description Table

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the plugin.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>Contains the properties of the plugin.</td>
</tr>
<tr>
<td>api</td>
<td>string</td>
<td>The API version of the plugin.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the plugin. This description is also displayed on the SPS web interface.</td>
</tr>
<tr>
<td>default_configuration</td>
<td>string</td>
<td>The default configuration of the plugin (an INI file as a string). For details, see the documentation of the particular plugin.</td>
</tr>
<tr>
<td>entry_point</td>
<td>string</td>
<td>The entry point of the plugin, for example, main.py</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the plugin. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>The path where the plugin is stored on SPS.</td>
</tr>
<tr>
<td>scb_max_version</td>
<td>string</td>
<td>The version number of the latest SPS release that is compatible with the plugin.</td>
</tr>
<tr>
<td>scb_min_version</td>
<td>string</td>
<td>The version number of the earliest SPS release that is compatible with the plugin.</td>
</tr>
<tr>
<td>sha256sum</td>
<td>string</td>
<td>The SHA-256 checksum of the plugin.</td>
</tr>
<tr>
<td>version</td>
<td>string</td>
<td>The version of the plugin.</td>
</tr>
</tbody>
</table>
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

Credential stores

Credential Stores offer a way to store user credentials (for example, passwords, private keys, certificates) and use them to login to the target server, without the user having access to the credentials. That way, the users only have to perform gateway authentication on SPS with their usual password (or to an LDAP database), and if the user is allowed to access the target server, SPS automatically logs in using the Credential Store.

URL

GET https://<IP-address-of-SPS>/api/configuration/policies/credentialstores

Cookies

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see Authenticate to the SPS REST API on page 19.</td>
</tr>
<tr>
<td>Cookie name</td>
<td>Description</td>
<td>Required</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
</tr>
</tbody>
</table>

NOTE: This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the credential stores.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/credentialstores
```

The following command retrieves the properties of a specific credential store.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/policies/credentialstores/<policy-id>
```

Response

The following is a sample response received when listing credential stores.

For details of the meta object, see Message format on page 10.

```json
{
    "items": [
        {
            "key": "1580973975727acedd51b2",
            "meta": {
                "href": "/api/configuration/policies/credentialstores/1580973975727acedd51b2"
            }
        },
        {
            "key": "935272738572bc2ec1dbdd",
            "meta": {
                "href": "/api/configuration/policies/credentialstores/935272738572bc2ec1dbdd"
            }
        }
    ],
    "meta": {
        "first": "/api/configuration/policies/audit_policies",
        "href": "/api/configuration/policies/credentialstores",
        "last": "/api/configuration/policies/usermapping_policies"
    }
}
```
When retrieving the endpoint of a specific credential store, the response is the following.

```json
{
  "body": {
    "name": "API_LOCAL",
    "type": {
      "authenticator_name": "auth_server_name",
      "default_namespace": "{HOST}",
      "dns_servers": {
        "primary": "192.168.56.1",
        "secondary": "192.168.56.2"
      },
      "domain_mappings": [
        {
          "domain": "domain",
          "host": {
            "selection": "fqdn",
            "value": "host"
          }
        }
      ],
      "login_mode": {
        "password": {
          "key": "e0ecbe98-bd17-4805-ba5d-17fb789f3971",
          "meta": {
            "href": "/api/configuration/passwords/e0ecbe98-bd17-4805-ba5d-17fb789f3971"
          }
        },
        "selection": "fixed",
        "username": "fixed_username"
      },
      "proxy_server": "http://192.168.56.201:9999",
      "selection": "local",
      "server_certificate_check": {
        "enabled": true,
        "trusted_ca": {
          "key": "12269547065727ad6e79d9e",
          "meta": {
            "href": "/api/configuration/policies/trusted_ca_lists/12269547065727ad6e79d9e"
          }
        }
      }
    }
  }
}```
{ }
,
"web_interface_url": "http://erpm_address"
},
"key": "935272738572bc2ec1dbdd",
"meta": {
  "first": "/api/configuration/policies/credentialstores/1580974097527acedd51b2",
  "href": "/api/configuration/policies/credentialstores/935272738572bc2ec1dbdd",
  "last": "/api/configuration/policies/credentialstores/935272738572bc2ec1dbdd",
  "next": null,
  "parent": "/api/configuration/policies/credentialstores",
  "previous": "/api/configuration/policies/credentialstores/1580974097527acedd51b2",
  "transaction": "/api/transaction"
}

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>string</td>
<td>Top level element, contains the ID of the credential store.</td>
</tr>
<tr>
<td>body</td>
<td>Top level element (string)</td>
<td>The configuration elements of the credential store.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the credential store. This name is also displayed on the SPS web interface. It cannot contain whitespace.</td>
</tr>
<tr>
<td>type</td>
<td>Top level item</td>
<td>All elements for the configured type of credential store.</td>
</tr>
<tr>
<td>authenticator_name</td>
<td>string</td>
<td>If your ERPM setup is configured to use an external authentication method, enter the name of the Authentication Server (Authenticator Source) set on your ERPM server. If empty, SPS uses the [Explicit] authenticator.</td>
</tr>
<tr>
<td>default_namespace</td>
<td>string</td>
<td>The default namespace of the accounts (for example, [Linux], [LDAP], [IPMI], W2003DOMAIN).</td>
</tr>
<tr>
<td>dns_servers</td>
<td>Top level</td>
<td>The IP addresses of the DNS servers to use</td>
</tr>
<tr>
<td>Element</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>domain_mappings</td>
<td>Top level item</td>
<td>Use for RDP connections only. In a domainless environment, use default_namespace.</td>
</tr>
<tr>
<td>encryption</td>
<td>Top level item</td>
<td>Configures the encryption key for the local credential store.</td>
</tr>
<tr>
<td>login_mode</td>
<td>Top level item</td>
<td>Configures the account SPS uses to login to the ERPM server.</td>
</tr>
<tr>
<td>plugin</td>
<td>string</td>
<td>Must be used if the selection element is set to external_plugin. References the Credential Store plugin. You can find the list of available plugins at the /api/configuration/plugins/credentialstore endpoint. To modify or add a plugin, use the value of the returned key as the value of the plugin element, and remove any child elements (including the key). Plugins can only be uploaded using the web interface of SPS.</td>
</tr>
<tr>
<td>proxy_server</td>
<td>string</td>
<td>The IP address and port of the proxy server. Use the http:// or https:// prefix.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Configures the type of the credential store. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- local Local credential store. Can only be configured using the web interface of SPS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- external_plugin Credential Store Plug-in. To upload or update a plugin, see Upload a plugin.</td>
</tr>
<tr>
<td>server_certificate_check</td>
<td>Top level item</td>
<td>To verify the certificate of the ERPM server, configure server_certificate_check.</td>
</tr>
<tr>
<td>web_interface_url</td>
<td>string</td>
<td>Name of the DN of the ERPM server. Use the http:// or https:// prefix.</td>
</tr>
<tr>
<td><strong>Elements of dns_servers</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>primary</td>
<td>string</td>
<td>The IP address of the primary DNS server.</td>
</tr>
<tr>
<td>secondary</td>
<td>string</td>
<td>The IP address of the secondary DNS server.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Elements of domain_mappings</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>string</td>
<td>The domain name used for Domain/Host mapping.</td>
</tr>
<tr>
<td>host</td>
<td>Top level item</td>
<td>The host name or address of the domain controller used for Domain/Host mapping.</td>
</tr>
<tr>
<td>selection</td>
<td>string</td>
<td>Declares if the value element contains an IP or an FQDN. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- fqdn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ip</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The IP address or hostname of the domain controller.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Elements of encryption</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>selection</td>
<td>string</td>
<td>Defines the encryption of the local credential store. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- basic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- password</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Elements of login_mode</strong></th>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>password</td>
<td>string</td>
<td>Must be used if the selection element is set to fixed_username. References the password SPS uses to authenticate on the ERPM server. You can configure passwords at the /api/configuration/passwords/ endpoint. To modify or add a password, use the value of the returned key as</td>
</tr>
</tbody>
</table>
### Elements of login_mode

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the value of the password element, and remove any child elements (including the key).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>selection</th>
<th>string</th>
<th>Possible values are:</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixed_username</td>
<td>SPS uses a fix username and password. Requires the username and password elements.</td>
<td></td>
</tr>
<tr>
<td>gateway_auth_credentials</td>
<td>SPS uses the username and password that the user provided during the gateway authentication process. Can only be used for SSH connections.</td>
<td></td>
</tr>
</tbody>
</table>

| username | string | Must be used if the selection element is set to fixed_username. The username SPS uses to authenticate on the ERPM server. |

### Elements of server_certificate_check

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set to true to verify the ERPM server's certificate.</td>
</tr>
</tbody>
</table>

| trusted_ca | string | Must be used if server certificate checking is enabled. References the list of trusted Certificate Authorities. You configure trusted CAs at the /api/configuration/policies/trusted_ca_lists/ endpoint. To reference a trusted CA list, use the value of the returned key as the value of the trusted_ca element, and remove any child elements (including the key). |

---

**Example:**

| NOTE: The following example is response only. Credential stores can only be configured using the web interface of SPS. Use a credential store plugin. |
Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
Completing the Welcome Wizard using REST

The Welcome Wizard helps you complete the initial configuration of SPS. Starting with version 5 F4, you can complete the Welcome Wizard using REST as well.

**URL**

```
GET https://<IP-address-of-SPS>/api/setup
```

**Prerequisites**

You can complete the Welcome Wizard only if it has not been already completed. To verify this, access the /api/setup endpoint. If the value of the status field is uninitialized, you can complete the Welcome Wizard.

**Sample request**

The following command completes the Welcome Wizard. The data content of the request is read from the file body.json. For the details of the body of the request, see Request body.

```
curl -H "Content-Type: application/json" -d @body.json -X POST https://<IP-address-of-SPS>/api/setup/
```

**NOTE:** The request automatically fails if there are any other clients connected to the REST or the web interface of SPS.

**Response**

If completing the Welcome Wizard is successful, you should receive the 303 status code. The body of the response is empty.
If you GET the /api/setup endpoint, the status field of the response should be completed, for example:

```json
{
    "meta": {
        "href": "/api/setup",
        "parent": "/api",
        "remaining_seconds": 0
    },
    "status": "completed"
}
```

### Request body

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accept_eula</td>
<td>boolean</td>
<td>Indicates that you have read and accept the terms of the Software Transaction, License and End User License Agreements. Must be true to complete the Welcome Wizard.</td>
</tr>
<tr>
<td>network</td>
<td>JSON object</td>
<td>Contains the initial networking configuration of SPS.</td>
</tr>
</tbody>
</table>
| license      | string     | Your SPS license as a string. You can download your license from support portal. Replace the line-breaks in the license file with \n characters, for example:

```
"license": "Product: Shell Control Box\nEdition: Single\n[
```

Note that you can complete the Welcome Wizard without uploading a license. In this case, SPS will start in demo mode. To skip uploading the license, use the null value:

```
"license": null,
```

To upload a license file, see Upload a new license.

<p>| certificates  | JSON object | Contains the initial certificates used on SPS: the internal Certificate Authority, Timestamping Authority, and the SSL certificate of the web and REST interface. After completing the Welcome Wizard, you can manage these certificates at Internal certificates on page 259. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| administration | JSON object   | Contains the passwords of the root and admin users, for example: \[
|              |               | "administration": {
|              |               |   "root_password": "asdgf-sdffe-aasc-oijernf",
|              |               |   "admin_password": "bd9r3-adedfk2-fsdf-fs"
|              |               | }, \[
| email       | JSON object   | Contains the SMTP server to use, and the e-mail address of the SPS administrator. For example: \[
|              |               | "email": {
|              |               |   "smtp_server": "smtp.example.com",
|              |               |   "admin_email": "psm-administrator@example.com"
|              |               | }, \[
| datetime    | JSON object   | Contains the timezone of SPS and the address of an NTP server to use for date synchronization. For example: \[
|              |               | "datetime": {
|              |               |   "timezone": "Europe/Budapest",
|              |               |   "primary_ntp_server": "time.test-domain"
|              |               | } \[
| network     | JSON object   | The initial networking configuration of SPS. \[
| hostname    | string        | Name of the machine running SPS. For example: \[
|              |               | "hostname": "psm", \[
| domainname  | string        | Name of the domain used on the network. For example: \[
|              |               | "domainname": "example.com", \[

One Identity
by Quest

SPS 6.13.0 REST API Reference Guide
Completing the Welcome Wizard using REST

857
<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| initial_address | IPv4 address/net-mask       | The IP address of interface 1 (or EXT, for older hardware) of SPS (for example, 192.168.1.1). The IP address can be chosen from the range of the corresponding physical subnet. Clients will connect to this interface, therefore it must be accessible to them. The IP prefix of the given range. For example, general class C networks have the /24 prefix. Use an IPv4 address.  

**NOTE:** Do not use IP addresses that fall into the following ranges:  
- 1.2.0.0/16 (reserved for communication between SPS cluster nodes)  
- 127.0.0.0/8 (localhost IP addresses)  

```
"initial_address":  
"192.168.1.10/24",
```

| vlantag | string | The VLAN ID of interface 1 (or EXT). Optional, use null if it is not set. For example:  

```
"vlantag": null,
```

**CAUTION:**  
Do not set the VLAN ID unless your network environment is already configured to use this VLAN. Otherwise, your SPS appliance will be unavailable using this interface.

| gateway | IPv4 address | The IP address of the default gateway.  

```
"gateway":  
"192.168.1.1",
```

Use an IPv4 address.

| primary_dns | IPv4 address | The IP address of the name server used for domain name resolution.  

```
```
Element

Type

Description
"primary_dns": "192.168.1.1",
Use an IPv4 address.

Element

Description
Type

certifica
tes

JSO- The internal certificates of SPS.
N
The key must be in PKCS-1 PEM format.
objeYou need the certificate and the private key as well.
ct
Encrypted private keys are not supported.
The attributes of the POST message that contain the
certificate and the private key must be a single line, enclosed
in double-quotes.
Replace line-breaks in the PEM certificate with \n
The certificate and the certificate chain must be valid, SPS
will reject invalid certificates and invalid certificate chains.
TIP: One Identity recommends using 2048-bit RSA keys
(or stronger).
For example:
"certificates": {
"ca": {
"certificate": "-----BEGIN
CERTIFICATE----\nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMAkGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n"
},
"webserver": {
"certificate": "-----BEGIN
CERTIFICATE----\nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMAkGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n",
"private_key": "-----BEGIN RSA
PRIVATE KEY-----\nMIIEogIBAAKCAQEA/JERC+o1UksvUfbzS5Yp77CNlS6RkkdZLPjl2i9+ACzv/lOy\n...\n-----END
RSA PRIVATE KEY-----\n"
},
"tsa": {
"certificate": "-----BEGIN

SPS 6.13.0 REST API Reference Guide
Completing the Welcome Wizard using REST

859


<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| ca      | JSO- N object | The certificate of SPS's internal Certificate Authority.
| webserv | JSO- N object | The SSL certificate of SPS's web and REST interface.
| tsa     | JSO- N object | The certificate of SPS's internal Timestamping Authority.

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>WebGuiOrRpcApiConfigInProgress</td>
<td>Web based or RPC API configuration is in progress — another client is connected to SPS. You can see the IP address of the client in the details key of the response, for example:</td>
</tr>
</tbody>
</table>

```json
{
    "error": {
        "details": {
            "user":
        }
    }
}
```
The Welcome Wizard was already successfully completed on this SPS.

```
{
  "error": {
    "message": "The configuration of the system is already initialized."
  },
  "type": "ConfigurationAlreadyInitialized"
}
```

---

401  ConfigurationAlreadyInitialized  The Welcome Wizard was already successfully completed on this SPS.
Enable and configure analytics using REST

Enable One Identity Safeguard for Privileged Analytics

This endpoint allows you to enable One Identity Safeguard for Privileged Analytics.

To enable One Identity Safeguard for Privileged Analytics and analyze the behavior of your users, One Identity Safeguard for Privileged Sessions (SPS) requires a special license. Also, depending on the number of your users and sessions, the performance and sizing of SPS must be considered. If you are interested in One Identity Safeguard for Privileged Analytics, contact our Sales Team, or your One Identity representative. For details on One Identity Safeguard for Privileged Analytics, see the One Identity One Identity Safeguard for Privileged Analytics website. For details on enabling One Identity Safeguard for Privileged Analytics, see Safeguard for Privileged Analytics Configuration Guide.

URL

GET https://<IP-address-of-SPS>/api/configuration/local_services/analytics/

Querying this endpoint returns the true if One Identity Safeguard for Privileged Analytics is enabled, false otherwise. For example:

```json
{
  "body": {
    "enabled": false
  },
  "key": "analytics",
  "meta": {
    "first": "/api/configuration/local_services/admin_web",
    "href": "/api/configuration/local_services/analytics",
    "last": "/api/configuration/local_services/user_web",
    "next": "/api/configuration/local_services/indexer"
  }
}
```
Enable One Identity Safeguard for Privileged Analytics

To modify enable One Identity Safeguard for Privileged Analytics, you have to complete the following.

Prerequisites

To enable One Identity Safeguard for Privileged Analytics and analyze the behavior of your users, One Identity Safeguard for Privileged Sessions (SPS) requires a special license. Also, depending on the number of your users and sessions, the performance and sizing of SPS must be considered. If you are interested in One Identity Safeguard for Privileged Analytics, contact our Sales Team, or your One Identity representative. For details on One Identity Safeguard for Privileged Analytics, see the One Identity One Identity Safeguard for Privileged Analytics website. For details on enabling One Identity Safeguard for Privileged Analytics, see Safeguard for Privileged Analytics Configuration Guide.

For details on uploading a license, see Upload a new license.

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Change the enabled option to true.**
   
   PUT the enabled option with the true value as a JSON object to the https://<IP-address-of-SPS>/api/configuration/local_services/analytics/ endpoint. For example:
   
   ```
   curl -H "Content-Type: application/json" -d '{ "enabled": true}' -X POST https://<IP-address-of-SPS>/api/configuration/local_services/analytics/
   ```

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 37.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>Updating the resource was successful</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the client is not authenticated and the resource requires authorization to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>

## Configure One Identity Safeguard for Privileged Analytics

The `/api/configuration/policies/analytics` endpoint allows you to configure One Identity Safeguard for Privileged Analytics by adding and removing analytics policies.

**URL**

GET `https://<IP-address-of-SPS>/api/configuration/policies/analytics/`

**Cookies**

<table>
<thead>
<tr>
<th>Cookie name</th>
<th>Description</th>
<th>Required</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>session_id</td>
<td>Contains the authentication token of the user</td>
<td>Required</td>
<td>The value of the session ID cookie received from the REST server in the authentication response, for example, a1f71d030e657634730b9e887cb59a5e56162860. For details on authentication, see <a href="#">Authenticate to the SPS REST API on page 19</a>.</td>
</tr>
</tbody>
</table>

**NOTE:** This session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).
Sample request
The following command lists the analytics policies configured.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/analytics/

The following command retrieves the properties of a specific policy.

curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/analytics/<policy-key>

Response
The following is a sample response received when listing analytics policies.

For details of the meta object, see Message format on page 10.

```json
{
  "items": [
    {
      "key": "9316362595a747b24d295e",
      "meta": {"href": "/api/configuration/policies/analytics/9316362595a747b24d295e"}
    },
    {
      "key": "9316362595a747b24d295f",
      "meta": {"href": "/api/configuration/policies/analytics/9316362595a747b24d295f"}
    }
  ]
}

```

When retrieving the endpoint of a specific analytics policy, the response is the following.

```json
{
  "body": {
    "name": "my_analytics_policy",
    "scoring": {
      "command": "trust",
```
"fis": "disable",
"hostlogin": "use",
"keystroke": "trust",
"logintime": "use",
"mouse": "disable",
"windowtitle": "disable"
}
},
"key": "9316362595a747b24d295e",
"meta": {
 "first": "/api/configuration/policies/analytics/9316362595a747b24d295e",
"href": "/api/configuration/policies/analytics/9316362595a747b24d295e",
"last": "/api/configuration/policies/analytics/9316362595a747b24d295e",
"next": null,
"parent": "/api/configuration/policies/analytics",
"previous": null,
"remaining_seconds": 600,
"transaction": "/api/transaction"

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body, or items when a list is returned</td>
<td>Top-level element (string)</td>
<td>Contains the properties of the analytics policy.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The unique name of the policy. This name is also displayed on the SPS web interface. It cannot contain whitespaces.</td>
</tr>
<tr>
<td>scoring</td>
<td>Top-level element</td>
<td>Scoring settings for analytics.</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>Top-level element, contains the ID of the policy.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements of scoring</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>command</td>
<td>string</td>
<td>Contains one of the following values:</td>
</tr>
<tr>
<td>fis</td>
<td>string</td>
<td>• disable: The algorithm is not used and is therefore not scoring session data.</td>
</tr>
<tr>
<td>hostlogin</td>
<td>string</td>
<td>• use: The algorithm is used and is therefore scoring session data. The highest and lowest scores given by this algorithm are ignored when aggregating scores.</td>
</tr>
<tr>
<td>keystroke</td>
<td>string</td>
<td>• trust: The algorithm is used and is therefore scoring session data. The highest and lowest scores given by this algorithm are taken into account when aggregating scores.</td>
</tr>
<tr>
<td>logintime</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>mouse</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>windowtitle</td>
<td>string</td>
<td></td>
</tr>
</tbody>
</table>
Add an analytics policy

To add an analytics policy, complete the following steps.

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Create the JSON object for the new analytics policy.**
   
   POST the JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/analytics endpoint. You can find a detailed description of the available parameters listed in Element.

   If the POST request is successful, when querying /api/configuration/policies/analytics, the response includes the key of the new analytics policy. For example:
   
   ```json
   {     "key": "1e089e2a-76b4-4079-94e3-c83ebc74dc2e",     "meta": {       "href": "https://<IP-address-of-SPS>/api/configuration/policies/analytics/1e089e2a-76b4-4079-94e3-c83ebc74dc2e",       "parent": "https://<IP-address-of-SPS>/api/configuration/policies/analytics",       "transaction": "https://<IP-address-of-SPS>/api/transaction"     }   }
   ```

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.

Modify an analytics policy

To modify an analytics policy, complete the following steps.

1. **Open a transaction.**
   
   For more information, see Open a transaction on page 29.

2. **Modify the JSON object of the analytics policy.**
   
   PUT the modified JSON object to the https://<IP-address-of-SPS>/api/configuration/policies/analytics/<policy-key> endpoint. You can find a detailed description of the available parameters listed in Element.

3. **Commit your changes.**
   
   For more information, see Commit a transaction on page 31.
Delete an analytics policy

To delete an analytics policy, complete the following steps.

1. **Open a transaction.**
   
   For more information, see [Open a transaction on page 29](#).

2. **DELETE the JSON object of the analytics policy.**
   
   DELETE the JSON object using the ID of the object as the key: https://<IP-address-of-SPS>/api/configuration/policies/analytics/<policy-key>. For details on how to delete an object, see [Delete an object on page 43](#).

   If the DELETE request is successful, when querying /api/configuration/policies/analytics, the response includes the key of the deleted analytics policy. For example:

   ```json
   {
     "meta": {
       "first": "/api/configuration/policies/analytics/9316362595a747b24d295e",
       "href": "/api/configuration/policies/analytics/9316362595a747b24d295e",
       "last": "/api/configuration/policies/analytics/9316362595a747b24d295e",
       "next": "/api/configuration/policies/analytics/9316362595a747b24d295e",
       "parent": "/api/configuration/policies/analytics",
       "previous": null,
       "transaction": "/api/transaction"
     }
   }
   ```

3. **Commit your changes to actually delete the object from SPS.** For details, see [Commit a transaction on page 31](#).

**Status and error codes**

The following table lists the typical status and error codes for this request. For a complete list of error codes, see [Application level error codes on page 37](#).

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>The new resource was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>SemanticError</td>
<td>The request to create an object has failed due to semantic errors in the configuration.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthenticated</td>
<td>The requested resource cannot be retrieved because the</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized</td>
<td>The requested resource cannot be retrieved because the client is not authorized to access it. The details section contains the path that was attempted to be accessed, but could not be retrieved.</td>
</tr>
<tr>
<td>404</td>
<td>NotFound</td>
<td>The requested object does not exist.</td>
</tr>
</tbody>
</table>
One Identity solutions eliminate the complexities and time-consuming processes often required to govern identities, manage privileged accounts and control access. Our solutions enhance business agility while addressing your IAM challenges with on-premises, cloud and hybrid environments.

Contacting us

For sales and other inquiries, such as licensing, support, and renewals, visit https://www.oneidentity.com/company/contact-us.aspx.

Technical support resources

Technical support is available to One Identity customers with a valid maintenance contract and customers who have trial versions. You can access the Support Portal at https://support.oneidentity.com/.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to videos at www.YouTube.com/OneIdentity
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product