One Identity Safeguard for Privileged Sessions 7.0.1 LTS

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/trademark-information.aspx. 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.

SPSREST API Reference Guide
Updated - 11 November 2022, 17:26

For the most recent documents and product information, see Online product documentation.

Contents

Introduction **10**
Message format 10
How to configure SPS using REST 14
How to configure SPS using REST: a sample transaction 15

Using the SPS REST API **19**
Authenticate to the SPS REST API 19
Authenticate to the SPS REST API using X.509 certificate 23
Listing SPS login methods 26
Retrieve user information 29
Checking the transaction status 31
Open a transaction 32
Commit a transaction 35
Delete a transaction 37
Reviewing the changelog of a transaction 39
Application level error codes 41
Navigating the configuration of SPS 44
Modifying the configuration of SPS 47
Delete an object 47
Create a new object 49
Change an object 53

Basic settings **57**
Retrieve basic firmware and host information 57
Listing integrated products with SPS 60
Firmware management 64
Retrieving information about SPS firmware image slots 66
Testing new SPS firmware before upgrade 70
Upgrading SPS to a new firmware 75
Uploading new firmware to SPS 80
Downloading and installing SPS firmware through HTTP 82
Network settings 91
Network configuration options 91

DNS servers 93
Routing between interfaces 96
Naming options 99
Network addresses 101
Routing table 107
Local services of SPS 109
Local services: Web login for administrators 113
Local services: Web login for users 117
Local services: cluster interface 122
System backup policy 124
Encrypting system backup policy 126
Date and time 129
Date & time 129
NTP servers 131
Timezone 134
Logs, monitoring and alerts 142
Management options 142
Web gateway authentication 146
Syslog server settings 148
Disk fill-up prevention 155
Mail settings 157
Health monitoring 165
SNMP settings 167
SNMP traps 169
Local services: access for SNMP agents 176
Alerting 184
System alerts 186
Traffic alerts 193
Trust stores 200
Enabling One Identity Safeguard Remote Access without Starling Join 207
Managing Starling Join 211
Retrieving the status of services related to Starling Join/Unjoin 215
User management and access control 219
User management and access control 219
Login settings 221

Privileges of usergroups ... 224
Audit data access rules ... 229
Active sessions ... 233
Manage users and usergroups locally on SPS ... 235
Manage usergroups locally on SPS ... 237
Manage users locally on SPS ... 241
Configuring LDAP servers ... 244
Configuring SPS login methods ... 257

Managing SPS ... 269
Troubleshooting options ... 269
Internal certificates ... 272
Passwords stored on SPS ... 275
Private keys stored on SPS ... 280
Private keys generated on SPS ... 284
Certificates stored on SPS ... 300
Local services: enabling SSH access to the SPS host ... 305
Manage the SPS license ... 311
Change contact information ... 315
Splunk integration ... 317
Splunk integration ... 321
Manage SPS clusters ... 328
Promote a SPS node to be the Central Management node in a new cluster ... 331
Join node(s) to the cluster ... 332
Query join status ... 334
Assign a role to a node ... 335
Query nodes ... 337
Query one particular node ... 338
Query the status of all nodes in the cluster ... 339
Query the status of one particular node ... 345
Upload and enable a configuration synchronization plugin ... 350
Disable a configuration synchronization plugin ... 355
Configuration tools in SPS ... 355
Resolving hostnames to IP addresses ... 357
Testing LDAP server connection ... 359

General connection settings **366**
Channel policy 366
Policies 372
Archive/Cleanup policy 376
Audit policies 381
Backup policy 390
Real-time content monitoring with Content Policies 398
LDAP servers 405
Signing CA policies 418
Time policy 422
Trusted Certificate Authorities 425
Local user databases 431
User lists 437

HTTP connections **443**
HTTP connections 443
HTTP connection policies 445
HTTP channels 464
HTTP authentication policies 465
Global HTTP options 472
HTTP settings policies 476
Creating custom HTTP error templates 483
Uploading a custom logo to your custom HTTP proxy error pages 487

Citrix ICA connections **491**
ICA connections 491
ICA connection policies 493
ICA channels 494
Global ICA options 496
ICA settings policies 500

MSSQL connections **506**
Limitations in handling MSSQL connections 506
MSSQL connections 506
MSSQL connection policies 509
MSSQL channels 527
MSSQL authentication policies 528

Global MSSQL options 535
MSSQL settings policies 539

RDP connections 545
RDP connections 545
RDP connection policies 547
RDP channels 572
Configuring domain membership 575
Global RDP options 578
RDP settings policies 582

SSH connections 590
SSH connections 590
SSH connection policies 592
SSH channels 612
SSH authentication policies 617
Global SSH options 629
SSH settings policies 634
SSH host keys and certificates 640

Telnet connections 646
Telnet connections 646
Telnet connection policies 648
Telnet channels 668
Telnet authentication policies 669
Global Telnet options 676
Telnet pattern sets 680

VNC connections 685
VNC connections 685
VNC connection policies 687
Global VNC options 688

Search, download, and index sessions 693
Audited sessions 693
Download audit trails 713
Searching in the session database 714
Searching in connection content 730

Generate and retrieve screenshot for content search 733
Session statistics 734
Session histogram 744
Session alerts 747
Session events 751
Indexing sessions 756
Session audit trail downloads 759
Local services: configuring the indexer 761
Indexer policies 769

Reporting 777
Reporting 777
Reports 780
Built-in subchapters 790
Pre-defined reports 793
Content subchapters 798
Connection statistics subchapters 806

Health and maintenance 813
Monitor appliance health status 813

Advanced authentication and authorization 820
Usermapping policy 820
Plugins 825
Upload a plugin 827
Delete a plugin 829
Check the integrity of a plugin 830
Authentication and authorization plugins 832
Configuring Authentication and Authorization plugin instances 836
Credential store plugins 840
Credential stores 843

Completing the Welcome Wizard using REST 851
Completing the Welcome Wizard using REST 851

Enable and configure analytics using REST 858
Enable One Identity Safeguard for Privileged Analytics 858
Configure One Identity Safeguard for Privileged Analytics 860

About us **866**

Contacting us **867**

Technical support resources **868**

1

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/authentication`. 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.

Element		Type	Description	Notes
`meta`			Top level element, contains links to different parts of the REST service	
	`changes`	string	Path to the transaction changelog	This value is always `/api/transaction/changes`. For details, see Reviewing the changelog of a transaction on page 39.
	`remaining_seconds`	integer	Time left until the session times out in seconds	SPS closes idle sessions after a period of inactivity. This value shows the number of seconds left until the timeout.
	`href`	string (relative path)	Path of the resource that returned the response. When creating a new object, this is the URL of the created object.	For example, `/api/authentication`
	`parent`	string (relative path)		
	`next`	string (relative path)	Path of the next sibling of the current resource	For example, `/api/configuration`
	`prev`	string (relative path)	Path of the previous sibling of the current resource	
	`first`	string (relative	Path of the	

Element		Type	Description	Notes
		path)	first sibling of the current resource	
	`last`	string (relative path)	Path of the last sibling of the current resource	
	`transaction`	string (`/api/transaction`)	The endpoint of the trans-action log	For details on how SPS handles transactions, see How to configure SPS using REST on page 14.
`items`		list of JSON objects	List of endpoints (objects) available from the current endpoint	Each object in the list contains a `key` and a `meta` object for the endpoint. For example:

```
{
        "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"
        }
        },
        {
        "key": "ssh-rsa-
```

Element	Type	Description	Notes

			`10.40.0.28:22",`
			`"meta": {`
			`"href": "/api/ssh-host-keys/ssh-rsa-10.40.0.28:22"`
			`}`
			`}`
			`]`
			`}`

For example:

```
{
                        "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.

Element		Type	Description	Notes
`error`			Top level element, contains links to different parts of the REST service	
	`type`	string	The type of the error that occurred	For example, `Unauthenticated`, or `NodeNotFound`. For a complete list, see Application level error codes on page 41.
	`message`	string	A textual message that describes the error	For example, `Unable to locate the requested path.`

	`details`	JSON object	List of additional information about the error (for example, the path where the error occurred)	For example:
				`"details": {`
				`"path": "no/such/path"`
				`}`

The following is a complete error response.

```
{
     "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 15.

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 32.

   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 44 and Modifying the configuration of SPS on page 47

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 35.

   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 37.

   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 ---
{
  "key": "settings",
  "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"
  },
  "settings": {
    "backend": {
      "cracklib_enabled": false,
      "expiration_days": 0,
      "minimum_password_strength": "good",
      "remember_previous_passwords": 10,
      "selection": "local"
    },
    "method": {
      "selection": "passwd"
    },
    "require_commitlog": false
  }
}
--- END 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.

```
# 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.

2

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 23.

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

  For more information, see Listing SPS login methods.
- 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 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

Header name	Description	Required	Values
Authorization	Contains the username and password of the user	Required	The string `Basic` followed by the username:password encoded using the RFC2045-MIME. For example, **`Basic YWRtaW46YQ==`**

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.

```
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.

```
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 more information on the `meta` object, see Message format on page 10.

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

Authenticate with SPS login methods

You can use different login methods (local, LDAP, RADIUS, or X.509) to authenticate to SPS. To determine which login methods are available to you on SPS, use the Listing SPS login methods on page 26 endpoint to list them.

URL

```
https://<IP-address-of-SPS>/api/authentication?login_method=<name-of-the-login-method>&type=<password|x509>
```

Query parameters related to the `/authentication` endpoint:

- ```
  ?type=<password|x509>
  ```

  Where each value indicates the means of authentication to SPS:

  - `password` - You have to send the username and password in a Basic Authorization header.
  - `x509` - You have to identify yourself with a client certificate

- ```
  ?login_method=<name-of-the-login-method>
  ```

  A unique identifier for the login method. For example, `x509_with_ldap_groups_name`.

Example: Authenticating with password

The following command authenticates the user to SPS with local login.

```
curl --user <username>:<password> https://<IP-address-of-SPS>/api/authentication?login_method=example_login_method&type=password
```

Example: Authenticating with a x509 certificate

The following command authenticates the user to SPS with a x509 certificate.

```
curl --key <private-key.pem> --cert <x509-certificate.pem> https://<IP-address-of-SPS>/api/authentication?login_method=example_login_method&type=x509
```

Response

In all four examples, the following sample response is received if the authentication is successful.

```
{
"meta": {
    "href": "/api/authentication",
    "next": "/api",
    "remaining_seconds": 600,
    "transaction": "/api/transaction"
}
```

The HTTP response code is `302 Moved Temporarily`. An HTTP response with this status code provides a redirect URL in the header field `Location`.

For more information on 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 41.

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

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/login_methods` endpoint.

  For more information, see Listing SPS login methods.
- 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

Header name	Description	Required	Values
Authorization	Contains the username and password of the user	Required	The string `Basic` followed by the username:password encoded using the RFC2045-MIME. For example, `Basic YWRtaW46YQ==`

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.

```
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.

```
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 more information on 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 41.

Code	Description	Notes
302	OK	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/`.
400	InvalidAuthenticationRequest	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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
405	MethodNotAllowed	You tried using an unsupported HTTP method. Use the GET method for authentication.

Listing SPS login methods

List login methods (local, LDAP, RADIUS, or X.509) to determine which login methods are available to you to authenticate to SPS. For more information on authentication, see Authenticate to the SPS REST API on page 19.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 login methods in SPS.

```
curl -X GET https://<IP-address-of-SPS>/api/authentication/login_methods
```

Response

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

For more information on the `meta` object, see Message format on page 10.

```
{
                "login_methods": [
                    {
                        "name": "local",
                        "title": "Local login",
                        "authentication": "local",
                        "credential": "password"
                    },
                    {
                        "name": "x509_name",
                        "title": "X509 login",
                        "authentication": "x509",
                        "credential": "x509"
                    },
                    {
```

```
                        "name": "x509_with_ldap_groups_name",
                        "title": "X509 login ldap groups",
                        "authentication": "x509",
                        "credential": "x509"
                    },
                    {
                        "name": "radius_with_local_groups_name",
                        "title": "Radius login with local groups",
                        "authentication": "radius",
                        "credential": "password"
                    },
                    {
                        "name": "radius_with_ldap_groups_name",
                        "title": "Radius login with ldap groups",
                        "authentication": "radius",
                        "credential": "password"
                    },

                ],
                "meta": {
                    "href": "/api/authentication/login_methods",
                    "next": "/api/authentication"
                }
            }
```


Elements of the response message body include:

Element	Type	Description	Notes
`login_methods`	object array	A list of available SPS login methods.	
`login_methods.name`	string	A unique identifier for the login method.	
`login_methods.title`	string	The title that appears above the login method button on the SPS web interface. This can be customized.	In the case of X.509 login method, there is only one button.
`login_methods.authentication`	enum	The login method used for authentication.	
`credential`	enum	The type of credential used for authentication.	Possible values are: • **`password`** Send the username and password in a Basic Authorization

Element	Type	Description	Notes
			header. • **x509** Use a X.509 certificate for authentication.

HTTP response codes

For more information and a list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
    "user": {
       "name": "admin",
    }
    "endpoints": [
       {
          "methods": [
             "DELETE",
             "GET",
             "POST",
             "PUT"
          ],
          "url": "/api"
       },
       {
          "...": "..."
       }
    ],
    "meta": {
       "href": "/api/user_info",
       "...": "..."
    }
 }
```

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.

Element	Type	Description
`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 41.

Code	Description	Notes
200	OK	User information has been retrieved successfully.
401	Unauthenticated	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.
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.

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 39.

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 more information on the `meta` object, see Message format on page 10.

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

Element	Type	Description
`transaction`		Top level element, contains the details of the current transaction
`status`	string	The status of the current transaction. By default, or after a successful commit it is `closed`. After successfully opening a transaction, it is `open`

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

Cookie name	Description	Required	Values
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 more information 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 more information on 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 41.

Code	Description	Notes
200	OK	Transaction opened successfully.
401	Unauthenticated	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.
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.
405	MethodNotAllowed	You tried using an unsupported HTTP method. Use the POST method to open a transaction.
409	WebGuiOrRpcApiConfigInProgress	The configuration of SPS is locked.

Code	Description	Notes
		Committing 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.
411	UnsupportedMethod	You must send a body (which can be empty) in this POST request, otherwise the SPS REST server returns a `411 - Length Required` error.

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

Cookie name	Description	Required	Values
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 more information 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).

PUT body

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

```
{
    "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.

```
{
    "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 more information on 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.

```
{
    "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 41.

Code	Description	Notes
200	OK	Transaction committed successfully.
401	Unauthenticated	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.
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.
405	MethodNotAllowed	You tried using an unsupported HTTP method. Use the PUT method to commit a transaction.

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

Cookie name	Description	Required	Values
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 more information on authentication, see Authenticate to the SPS REST API on page 19.

Cookie name	Description	Required	Values
			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 more information on 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 41.

Code	Description	Notes
200	OK	Transaction deleted successfully.
401	Unauthenticated	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.
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.

Code	Description	Notes
405	MethodNotAllowed	You tried using an unsupported HTTP method. Use the DELETE method to reset a transaction.

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 on page 14.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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.

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

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 41.

Code	Description	Notes
200	OK	Transaction changelog has been retrieved successfully.
401	Unauthenticated	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.
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.
405	MethodNotAllowed	You tried using an unsupported HTTP method. Use the GET method to retrieve the changelog a transaction.

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.

Code	Description	Notes
400	InvalidRequestBody	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.
400	ConfigTreeNotAvailable	An error occurred while preparing the configuration tree for the REST API.

Code	Description	Notes
400	SyntacticError	A value to be set is not accepted syntactically. The `details` section contains the path that was found to be invalid.
400	InvalidPath	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.
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.
401	Unauthenticated	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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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	NodeNotFound	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.
404	NodeNotAvailable	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.
405	MethodNotAllowed	An attempt was made to change a config-

Code	Description	Notes
		uration subtree in an unsupported way. The method <method> is not allowed for this node.
409	MidAirCollisionSemanticError	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.
409	WebGuiOrRpcApiConfigInProgress	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.
409	MidAirCollision	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.
409	NoTransaction	An attempt was made to change the configuration when no transaction was open.
409	DoubleTransaction	This error is returned when the client attempts to open a transaction while another transaction of that client is already started.
417	Expectation Failed	If you receive the "417 - Expectation Failed" error code when using curl, use curl with the `--http1.0` or the `-H "Expect:"` option.

Code	Description	Notes
500	CommitMessageMissing	This error is returned when a commit message is required for committing a transaction, but it was not provided in the commit request.
500	TransactionCommitError	Unexpected internal errors during committing a transaction are interpreted as TransactionCommitError.
500	AuthorizationError	The request could not be authorized due to an unexpected internal error.

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 49.

To modify or delete an object, you need the ID of the object. For details, see Change an object on page 53 and Delete an object on page 47.

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

```
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"
    }
  },
  {
    "key": "network",
    "meta": {
      "href": "/api/configuration/network"
    }
  },
```

```
    {
      "key": "passwords",
      "meta": {
        "href": "/api/configuration/passwords"
      }
    },
    {
      "key": "plugins",
      "meta": {
        "href": "/api/configuration/plugins"
      }
    },
    {
      "key": "policies",
      "meta": {
        "href": "/api/configuration/policies"
      }
    },
    {
      "key": "private_keys",
      "meta": {
        "href": "/api/configuration/private_keys"
      }
    },
    {
      "key": "rdp",
      "meta": {
        "href": "/api/configuration/rdp"
      }
    },
    {
      "key": "reporting",
      "meta": {
        "href": "/api/configuration/reporting"
      }
    },
    {
      "key": "ssh",
      "meta": {
        "href": "/api/configuration/ssh"
      }
    },
    {
      "key": "telnet",
      "meta": {
        "href": "/api/configuration/telnet"
      }
    },
```

```
    {
      "key": "troubleshooting",
      "meta": {
        "href": "/api/configuration/troubleshooting"
      }
    },
    {
      "key": "trust_stores",
      "meta": {
        "href": "/api/configuration/trust_stores"
      }
    },
    {
      "key": "vnc",
      "meta": {
        "href": "/api/configuration/vnc"
      }
    },
    {
      "key": "x509",
      "meta": {
        "href": "/api/configuration/x509"
      }
    }
  ]
}
--- END RESPONSE BODY ---
```


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 35.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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.

```
{
  "meta": {
    "first": "/api/configuration/rdp/channel_policies/-20100",
    "href": "/api/configuration/rdp/channel_policies/<id-of-the-deleted-
object>",
    "last": "/api/configuration/rdp/channel_policies/<id-of-the-deleted-
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 41.

Code	Description	Notes
200	OK	The resource was successfully deleted.
401	Unauthenticated	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.
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.
409	Conflict	No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction.

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:

```
"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:

```
{
      "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

Header name	Description	Required	Values
Content-Type	Specifies the type of the data sent. SPS uses the JSON format	Required	`application/json`
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.

Sample request

The following command creates a new 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=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.

```
{
  "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, showing the properties of Content policy objects.

For more information on 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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	Bad Request	The request body format is invalid. The data is not a properly formatted JSON object.
401	Unauthenticated	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.
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.
409	Conflict	No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction on page 32.

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

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:

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

URL

```
PUT https:<IP-address-of-SPS>/api/configuration/<path-to-the-parent-
resource>/<id-of-the-object-to-modify>
```

Table 2: Headers

Header name	Description	Required	Values
Content-Type	Specifies the type of the data sent. SPS uses the JSON format	Required	`application/json`
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.

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=07640a0bf14cdd361d8f5ae2b0b482a786c7a604 -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.

```
{
    "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 more information on the `meta` object, see Message format on page 10.

```
{
    "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 41.

Code	Description	Notes
201	Created	The new resource was successfully created..
400	Bad Request	The request body format is invalid. The data is not a properly formatted JSON object.
401	Unauthenticated	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.
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.
409	Conflict	No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction on page 32.
417	Expectation Failed	If you receive the "417 - Expectation Failed" error code when using curl, use curl with the `--http1.0` or the `-H "Expect:"` option.

3

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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": "2abde4c81d9b544bf53fae4f4b9657fc",
        "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"
```

```
        ],
         "support_link": "mailto:scb-administrator@example.com",
         "version": "5 F7"
    },
    "key": "about_info",
    "meta": {
         "href": "/api/info",
         "remaining_seconds": 9889
         "parent": "/api"
    }
}
```


Element	Description
analytics_enabled	Indicates whether or not the One Identity Safeguard for Privileged Analytics module has been enabled.
build_date	Build date of the SPS firmware. This element is included in the response only for authenticated users.
config_hash	Contains the hash of the XML database running on the given SPS host.
domainname	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 99.
hostname	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 99.
nickname	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 99.
plugin_sdk_version	The version number of the Plugin SDK. • The value of feature represents the feature release version. • The value of full represents the minor release version.
support_link	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 157.
firmware_version	The version number of the firmware running on SPS, for

Element	Description
	example, `4.3.2a`. This element is included in the response only for authenticated users.
`version`	The name of the major release running on SPS, for example, `4 F3`. This element is included in the response only for authenticated users.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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.

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

Cookie name	Description	Required	Values
session_id	Contains the authentication token of the	Required	The value of the session ID cookie received from the REST server in the authentication response, for example,

Cookie name	Description	Required	Values
	user		**a1f71d030e657634730b9e887cb59a5e56162860**. For more information 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 `/integrated_products` endpoint include:

Operation	HTTP method	URL	Notes

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`

Operation	HTTP method	URL	Notes
			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.

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 more information on the `meta` object, see Message format on page 10.

```
{
     "items": [
     {
       "name": "Defender",
       "link": null,
       "activated": false
     },
     {
       "name": "Connect",
       "link": null,
       "activated": false
     },
     {
       "name": "Governance",
       "link": null,
       "activated": false
     },
     {
       "name": "RemoteAccess",
       "link": null,
```

```
      "activated": false
    }
   ]
  }
```

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:

Element	Type	Description	Notes
`items`	object array	A list of One Identity Starling products	When there are no products integrated with

Element	Type	Description	Notes
		that are integrated SPS.	SPS, the `items` field returns empty: `{ "items": [] }`
`items.name`	string	The name of the integrated One Identity Starling product.	
`items.link`	format (uri)	The URL of the integrated One Identity Starling product.	If the product is not integrated, the value of the link parameter will be null.
`items.activated`	boolean	Indicates whether the product is integrated with SPS or not.	Possible values: • true - the product is integrated with SPS • false - the product is not integrated with SPS

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

Firmware management

A list of endpoints managing SPS firmware images.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "fetch",
      "meta": {
        "href": "/api/firmware/fetch"
      }
    },
       {
      "key": "slots",
      "meta": {
        "href": "/api/firmware/slots"
      }
    },
    {
      "key": "test",
```

```
      "meta": {
        "href": "/api/firmware/test"
      }
    },
    {
      "key": "upgrade",
      "meta": {
        "href": "/api/firmware/upgrade"
      }
    }
  ],
  "meta": {
    "href": "/api/firmware",
    "parent": "/api",
    "fetch": "/api/firmware/fetch",
    "slots": "/api/firmware/slots",
    "test": "/api/firmware/test",
    "upgrade": "/api/firmware/upgrade"
  }
}
```


Endpoints	Description
`fetch`	Install firmware files by providing a URL.
`slots`	Retrieve information about SPS firmware images maintained on the device in locations called slots.
`test`	Trigger an upgrade test without an actual upgrade.
`upgrade`	Upgrade SPS to new firmware.
`upload`	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 41.

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

Cookie name	Description	Required	Values
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 more information 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:

Operation	HTTP method	URL	Notes
Retrieving information from firmware image slots	GET	`/api/firmware/slots`	
Retrieving information from a specific firmware image slot	GET	`/api/firmware/slots/<slot_id>`	Example: `/api/firmware/slots/1`
Deleting a firmware image slot	DELETE	`/api/firmware/slots/<slot_id>`	NOTE: Deleting the current firmware, or the one after the reboot, is not allowed.

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 if the request was successful.

For more information on the `meta` object, see Message format on page 10.


```
{
     "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"
          },
          "key": "4",
          "meta": {
            "href": "/api/firmware/slots/4"
           }
        },
        {
          "body": {},
          "key": "5",
          "meta": {
            "href": "/api/firmware/slots/5"
          }
        }
      ],
      "meta": {
        "current_slot": "/api/firmware/slots/1",
        "href": "/api/firmware/slots",
        "number_of_empty_slots": 2,
        "parent": "/api/firmware",
        "slot_after_reboot": "/api/firmware/slots/2"
      }
    }
```


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:

Element	Type	Description	Notes
`after_reboot`	boolean	This flag shows that the	Possible values:

Element	Type	Description	Notes
		firmware is selected to be the active firmware after upgrade.	• true - the firmware is the active firmware after upgrade • false - the firmware is not the active firmware after upgrade
`current`	boolean	Indicates whether the firmware is active.	Possible values: • **`true`** - the current firmware is the latest version • **`false`** - the current firmware is not the latest version
`upgrade_news`	string	Displays SPS *Upgrade Notes* that is relevant to the current firmware.	If there is no such information available, the value will be **`null`**.
`upgrade_notes`	string	The content of the SPS *Upgrade Notes*.	
`version`	number	The version number of that specific SPS firmware image.	

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

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

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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/test` endpoint include:

Operation	HTTP method	URL	Notes
Testing new SPS firmware before upgrade	POST	`/api/firmware/test`	

Sample request

The following command triggers a firmware upgrade test.

```
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

```
{"error": {"type": "IncompleteRequestBodyError"}}
```

  - the tested firmware slot is empty

```
{"error": {"type": "ResourceNotFound"}}
```

- the upgrade was not successful due to an error:

```
{"error": {"type": "FirmwareTestFailed"}}
```

The following is a sample response received when the upgrade test was successful.

For more information on 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": {
      "href": "/api/firmware/test",
      "parent": "/api/firmware"
    }
  }
```

Elements of the response message body include:

Element	Type	Description	Notes
`error.details`	object		
`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/test`
`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 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:

Element	Type	Description	Notes
error.details	object		
error.details.missing_paths	array	A list of missing URL path parameters.	In this case, there can be only one value here, which is slot_id.
error.details.missing_paths.slot_id		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 the upgrade test was not successful, because an error was found during testing.

```
{
        "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:

Element	Type	Description	Notes
error.details	object		
error.details.exit_code	number		Possible values: • 0 - the firmware test passed successfully • 1 - the firmware test failed
error.details.test_	string	The summary of the	For example:

Element	Type	Description	Notes
`summary`		upgrade test in free text format.	HA check started, HA check failed
`error.message`	string	The content of the error message.	
`error.type`	string	The type of the error message.	

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

Upgrading SPS to a new firmware

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

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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/upgrade` endpoint include:

Operation	HTTP method	URL	Notes
Upgrading SPS to a new firmware	POST	`/api/firmware/upgrade`	

Sample request

The following command upgrades SPS to a new firmware.

```
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:

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

- the upgrade was not successful due to invalid input:
  - the `slot_id` is missing from the payload

    ```
    {"error": {"type": "IncompleteRequestBodyError"}}
    ```

  - the tested firmware slot is empty

    ```
    {"error": {"type": "ResourceNotFound"}}
    ```

- the upgrade was not successful due to an error:

  ```
  {"error": {"type": "FirmwareTestFailed"}}
  ```

The following is a sample response received when a SPS is upgraded to a new firmware.

For more information on the `meta` object, see Message format on page 10.

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

```
      "parent": "/api/firmware",
      "slot": "/api/firmware/slots/1"
    }
  }
```

The following is a sample response received when a firmware upgrade is attempted on an empty slot.

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

Elements of the response message body include:

Element	Type	Description	Notes
`error.details`	object		
`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.

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

Elements of the response message body include:

Element	Type	Description	Notes
`error.details`	object		
`error.details.missing_paths`	array	A list of missing URL path parameters.	In this case, there can be only one value here, which is `message`.
`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 the firmware test fails during upgrade.

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

```
      "href": "/api/firmware/upgrade",
      "parent": "/api/firmware"
    }
  }
```

Elements of the response message body include:

Element	Type	Description	Notes
`error.details`	object		
`error.details.exit_code`	number		Possible values: • 0 -the firmware test passed successfully • 1 - the firmware test failed
`error.details.test_summary`	string	The summary of the upgrade test in free text format.	For example: • HA check started • HA check failed
`error.message`	string	The content of the error message.	
`error.type`	string	The type of the error message.	

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.

HTTP response code	Status/Error	Description
400	UpgradeMessageMissing	The upgrade request did not contain a message. Adding a message is required.

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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 more information 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 `/upload/firmware` endpoint include:

Operation	HTTP method	URL	Notes
Uploading SPS firmware images	POST	`/api/upload/firmware`	The **`enctype`** attribute of the POST request must be `multipart/form-data`.

Sample request

The following command uploads a new firmware to SPS.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
        "body": {
           "after_reboot": true,
           "current": false,
           "upgrade_news": null,
           "upgrade_notes": "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:

Element	Type	Description	Notes
`after_reboot`	boolean	This flag shows that the firmware is selected to be the active firmware after upgrade.	In this particular case, the value of **`after_reboot`** will always be false, as the firmware has not yet been chosen for upgrade.
`current`	boolean	Indicates whether the firmware is active.	Possible values: • true - the current firmware is the latest version • false - the current firmware is not the latest version
`upgrade_news`	string	Displays SPS *Upgrade Notes* that is relevant to the current firmware.	If there is no such information available, the value will be null.

Element	Type	Description	Notes
`upgrade_notes`	string	The content of the SPS *Upgrade Notes* for that firmware image.	
`version`	number	The version number of that specific SPS firmware image.	

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

Downloading and installing SPS firmware through HTTP

With the `/firmware/fetch` endpoint you can avoid having to manually upload large SPS firmware ISO files before upgrading. Instead, you can install SPS firmware files by providing a URL through the SPS REST API. This also makes it easier to download and install a new firmware image on multiple SPS instances simultaneously. Download requests are processed asynchronously, as indicated by the `202 Accepted` HTTP response.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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

Cookie name	Description	Required	Values
			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/fetch` endpoint include:

Operation	HTTP method	URL	Notes
Start downloading a SPS firmware	POST	`/api/firmware/fetch`	
Retrieve the status of SPS firmware downloads	GET	`/api/firmware/fetch`	
Retrieve the status of a single SPS firmware download	GET	`/api/firmware/fetch/<key-of-the-SPS-firmware-download>`	
Cancel a SPS firmware download	DELETE	`/api/firmware/fetch/<key-of-the-SPS-firmware-download>`	You can safely cancel running a SPS firmware file download while the status of the download is `queued` or `in-progress`. If you cancel running a firmware download while its status is in-progress, any partially downloaded files will be removed from SPS. However, the status information remains in the queue, and can

Operation	HTTP method	URL	Notes
			be removed by sending a second DELETE request to the same URL. NOTE: After the firmware download and verification steps are completed, it is not possible to cancel the installation of the firmware.
Remove status information of finished, failed, and cancelled downloads	DELETE	`/api/firmware/fetch/<key-of-the-SPS-firmware-download>`	There is no automated procedure to remove outdated firmware downloads. You must remove them manually.

Sample request

Example: Start downloading SPS firmware files

The following command starts downloading SPS firmware files.

```
curl -X POST -b "${COOKIE_PATH}" --data "@-" https://<IP-address-of-SPS>/api/firmware/fetch
```

```
{
            "url": "https://example.com/sps.iso",
            "checksum": {
                "sha256": "e6837af0b5d-cb5397c6bc8b52c0fb95aa49a3fea4303e274c7501f0857cb9b80"
```

```
        }
    }
```

Elements of the request message body include:

Element	Type	Description	Notes
`url`	string		
`checksum`	object	When SPS downloads from a HTTPS URL, the server certificate is not validated during the SSL handshake. As a result, a checksum is required so that SPS can verify that the data was not tampered with in transit. The SPS download page for the respective version of the product provides the SHA-256 checksum of all official SPS images.	`checksum` is a required parameter and can only accept a `sha256` hash as a value. This parameter must be used when you are running SPS in a production envir-onment.
`checksum.sha256`	string	The SHA-256 hash of the SPS ISO image.	

Response

The following is a sample response received if the request was successful.

For more information on the `meta` object, see Message format on page 10.

```
{
            "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
            "meta": {
                "href": "/api/firmware/fetch/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                "parent": "/api/firmware",
                "slots": "/api/firmware/slots",
                "test": "/api/firmware/test",
                "upgrade": "/api/firmware/upgrade"
            }
        }
```

Elements of the response message body include:

Element	Type	Description	Notes
key	string	The identifier of the SPS firmware file that is being downloaded.	

Example: Retrieving the status of SPS firmware downloads

The following command is used to retrieve the download status of SPS firmwares.

```
curl -X GET -b "${COOKIE_PATH}" https://<IP-address-of-
SPS>/api/firmware/fetch
```

Response

The following is a sample response received if the request was successful.

For more information on the `meta` object, see Message format on page 10.


```
{
                "items" : [
                    {
                        "body" : {
                            "bytes_downloaded" : 1234,
                            "bytes_total" : 1521614848,
                            "error": null,
                            "info" : "Downloading",
                            "slot_id" : null,
                            "start_time": "2022-03-30T12:00:01+00:00",
                            "status" : "in-progress",
                            "url" : "https://example.com/sps.iso"
                        },
                        "key" : "42dee328-9a26-4337-a1fd-3ee6cea3e815",
                        "meta" : {
                            "href" : "/api/firmware/fetch/42dee328-9a26-4337-a1fd-
3ee6cea3e815"
                        }
                    },
                    {
                        "body" : {
                            "bytes_downloaded" : 0,
                            "bytes_total" : 0,
                            "error": "<class 'request-
s.exceptions.ConnectionError'>: HTTPConnectionPool(host='unreachable.com',
port=80): Max retries exceeded with url: /sps.iso (Caused by NewConnectionError
('<urllib3.connection.HTTPConnection object at 0x7fc45e0e1ac0>: Failed to
establish a new connection: [Errno -2] Name or service not known'))",
```

```
                        "info" : "Firmware fetching failed",
                        "slot_id" : null,
                        "start_time": "2022-03-30T12:00:01+00:00",
                        "status" : "failed",
                        "url" : "https://unreachable.com/sps.iso"
                    },
                    "key" : "023bc2b4-0efb-4eaa-a301-b29f8c395f8c",
                    "meta" : {
                        "href" : "/api/firmware/fetch/023bc2b4-0efb-4eaa-a301-
b29f8c395f8c"
                    }
                },
                {
                    "body" : {
                        "bytes_downloaded" : 1594798080,
                        "bytes_total" : 1594798080,
                        "error": null,
                        "info" : null,
                        "slot_id" : 2,
                        "status" : "finished",
                        "start_time": "2022-03-30T12:00:01+00:00",
                        "url" : "https://example.com/sps_old.iso"
                    },
                    "key" : "8f6f7b0f-bba1-4d86-babb-54868106d2e9",
                    "meta" : {
                        "href" : "/api/firmware/fetch/8f6f7b0f-bba1-4d86-babb-
54868106d2e9",
                        "slot": "/api/firmware/slots/2"
                    }
                },
                {
                    "body" : {
                        "bytes_downloaded" : 0,
                        "bytes_total" : 0,
                        "error": null,
                        "info" : null,
                        "slot_id" : null,
                        "status" : "queued",
                        "start_time": null,
                        "url" : "https://example.com/sps.iso"
                    },
                    "key" : "6bf3113c-ad66-4cb3-9827-77fcbe1b6853",
                    "meta" : {
                        "href" : "/api/firmware/fetch/6bf3113c-ad66-4cb3-9827-
77fcbe1b6853"
                    }
                }
            ],
            "meta": {
                "href": "/api/firmware/fetch",
                "parent": "/api/firmware",
                "slots": "/api/firmware/slots",
                "test": "/api/firmware/test",
                "upgrade": "/api/firmware/upgrade"
```

```
            }
        }
```

Example: Retrieving the status of a single SPS firmware download

The following command is used to retrieve the download status of a single SPS firmware.

```
curl -X GET -b "${COOKIE_PATH}" https://<IP-address-of-
SPS>/api/firmware/fetch/<key-of-the-SPS-firmware-download>
```

Response

The following is a sample response received if the request was successful.

For more information on the `meta` object, see Message format on page 10.

```
{
            "body" : {
                "bytes_downloaded" : 1594798080,
                "bytes_total" : 1594798080,
                "error": null,
                "info" : null,
                "slot_id" : 2,
                "start_time": "2022-03-30T12:00:01+00:00",
                "status" : "finished",
                "url" : "https://example.com/sps.iso"
            },
            "key" : "8f6f7b0f-bba1-4d86-babb-54868106d2e9",
            "meta": {
                "href": "/api/firmware/fetch/8f6f7b0f-bba1-4d86-babb-
54868106d2e9",
                "parent": "/api/firmware/fetch",
                "slot": "/api/firmware/slots/2"
            }
        }
```

Elements of the response message body include:

Element	Type	Description	Notes
key	string	The identifier	

Element	Type	Description	Notes
		of the SPS firmware download job that is running in the background.	
`bytes_downloaded`	number	The number of bytes already downloaded.	
`bytes_total`	number	The total number of bytes to be downloaded from the `Content-Length` header received from the server.	
`error`	string	The error message, containing details of the error.	If no error has happened, then the value is `null`.
`info`	string	Indicates download-specific information, for example, the SPS firmware file is currently downloading, or that SPS is unable to download the firmware file.	Possible values: • `null` • `Downloading` • `Verifying checksum` • `Firmware fetching failed` • `Installing` The value of the `info` field can be `null` in two scenarios: • The firmware download has not started yet. • The firmware download and installation has been successful.
`slot_id`	number	The identifier of theSPS firmware image slot.	Before the firmware download is finished, the value of the `slot_id` is `null`.

Element	Type	Description	Notes
			When the status of the firmware download is finished, the `slot_id` indicates the location where the downloaded firmware was installed. Note that if an administrator later runs other operations on that particular firmware slot, like deleting it or uploading another firmware, then these will not be reflected in the status of the firmware download.
`start_time`	string (ISO 8601)	The time at which the SPS firmware download job has started.	
`status`	string	The status of the SPS firmware fetching job.	Possible values are: • `queued` - the SPS firmware download is queued. • `in-progress` - the SPS firmware download is being processed. • `finished` - the SPS firmware download has been completed. • `failed` - the SPS firmware could not be downloaded. • `canceled` - the SPS firmware download was canceled.
`url`	string (URI)	The resource URL of the SPS firmware.	Example: `https://example.com/sps.iso`

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

Network settings

Network configuration options

Contains the endpoints for configuring networking on SPS.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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",
        "next": "/api/configuration/passwords",
        "parent": "/api/configuration",
        "previous": "/api/configuration/management",
        "transaction": "/api/transaction"
    }
}
```


Element	Description
dns	The address of the primary and secondary DNS server.

Element	Description
`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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

DNS servers

Contains the address of the primary and secondary DNS server.

URL

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

Cookies

Cookie name	Description	Required	Values
session_	Contains the	Required	The value of the session ID cookie received

Cookie name	Description	Required	Values
id	authentication token of the user		from the REST server in the authentication response, for example, **a1f71d030e657634730b9e887cb59a5e56162860**. For more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
    "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"
    }
 }
```

Element	Type	Description
key	string	Top level element, contains the ID of the

Element	Type	Description
		endpoints.
`body`	Top level element (string)	Contains the addresses of the DNS servers.
`primary`	string	The IP address of the primary DNS server.
`secondary`	string	The address of the secondary DNS server.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
    "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"
    }
 }
```


Element	Type	Description
key	string	Top level element, contains the ID of the endpoint.
body	Top level element (list)	Contains the rules for routing between the network interfaces.
interface_a	string	References the identifier of the network interface. You can configure network interfaces at the `/api/configuration/network/nics/` endpoint. 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).
interface_b	string	References the identifier of the network interface. You can configure network interfaces at the

Element	Type	Description
		`/api/configuration/network/nics/` endpoint. To modify or add a network interface, use the value of the returned `key` as the value of the `interface_b` element, and remove any child elements (including the key).

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 32.

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 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
    }
 }
```


Element		Type	Description
key		string	Top level element, contains the ID of the endpoint.
body		Top level element (string)	Contains the naming settings.
	domainname	string	The domain name of the network.
	hostname	string	The hostname of SPS.
	nickname	string	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.

Modify a name

To modify a name, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Network addresses

Contains the network addresses configured for each physical NIC.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "nic1",
      "meta": {
```

```
        "href": "/api/configuration/network/nics/nic1"
      }
    },
    {
      "key": "nic2",
      "meta": {
        "href": "/api/configuration/network/nics/nic2"
      }
    },
    {
      "key": "nic3",
      "meta": {
        "href": "/api/configuration/network/nics/nic3"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/network/dns",
    "href": "/api/configuration/network/nics",
    "last": "/api/configuration/network/routing",
    "next": "/api/configuration/network/routing",
    "parent": "/api/configuration/network",
    "previous": "/api/configuration/network/naming",
    "transaction": "/api/transaction"
  }
}
```


When retrieving the endpoint of a specific physical network interface, the response is the following.


```
{
  "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"
    }
}
```


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

Element	Type	Description
		• `10-full` 10BaseT/Full. • `100-full` 100BaseT/Full. • `1000-full` 1000BaseT/Full.

Elements of `interfaces`	Type			Description
`@order`			list	Lists the keys of the interfaces in the order they are be displayed on the SPS web UI.
`<key-of-an-interface>`			string	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.
	`addresses`		string	Contains the addresses of the interface, and their display order.
		`<key-of-address>`	string	Contains the IP address range.
		`@order`	list	Lists the keys of the addresses in the order they are displayed on the SPS web UI.
	`mtu`		integer	Maximum Transmission Unit (MTU) to set per network interface (VLAN or network interface card). Default value: `1500`
	`name`		string	The name of the interface, as displayed on the SPS web UI.
	`source_`		list	Contains details of the network

Elements of `interfaces`	Type	Description
	`based_routes`	routing rule specific to packets coming out of this particular interface.
	`vlantag` string	The ID of the interface. For the physical interface, the value is `0`. For virtual interfaces, the value is between `1` and `4094`. **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.**

Elements of `source_based_routes`	Type	Description
`gateway`	string	The IPv4 or IPv6 address of the gateway used to access the network set in this routing rule.
`target_network`	string	The IPv4 or IPv6 address of the host or network accessible via this routing rule.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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

Code	Description	Notes
		could not be retrieved.
404	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
    }
 }
```

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

Add a subnet

To add a subnet, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 35.

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 32.

2. **Modify the JSON object of the routing table.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/netowrk/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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
            }
        },
        {
            "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"
    }
}
```


Element	Description
admin_web	Web login for administrators and users: On this address, users can,

Element	Description
	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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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.ff7574025754b3df1647001.addresses.1",
                                        "meta": {
                                                "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/address
es/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	Top level element (string)	Contains the configuration options of the SPS web interface.
access_	JSON	Enables and configures limitations on the clients that can

Element		Type	Description
`restriction`		object	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.
	`address`	JSON object	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:

```
{
    "body": {
        "interfaces": {
            "@order": [
                "ff7574025754b3df1647001"
            ],
            "ff7574025754b3df1647001": {
                "addresses": {
                    "1": "10.40.255.171/24",
                    "@order": [
                        "1"
                    ]
                },
                "name": "default",
                "vlantag": 0
            }
        },
        "name": "eth0",
        "speed": "auto"
    },
```

Element	Type	Description

```
    "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.

```
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.in-
terfaces.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.in-
terfaces.ff7574025754b3df1647001.addresses.1",
    "meta": {
        "href": "/api/-
config-
uration/net-
work/n-
ics/n-
ic1#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`.
`https_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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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.ff7574025754b3df1647001.addresses.1",
                    "meta": {
                        "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/address
es/1"
```

```
                    }
                },
                "http_port": 80,
                "https_port": 443
            }
        ]
    },
    "key": "user_web",
    "meta": {
        "first": "/api/configuration/local_services/user_web",
        "href": "/api/configuration/local_services/user_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			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.

Element	Type	Description
`address`	JSON object	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:

```
{
    "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.

Element	Type	Description

		`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" } },`

`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`.
`https_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 41.

Code	Description	Notes
401	Unauthenticated	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.

Code	Description	Notes
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
       "body": {
              "enabled": true,
              "listen_address": {
                     "key":
"nic1.interfaces.ff7574025754b3df1647001.addresses.2553887595ce3ca7f1eae4",
                     "meta": {
                            "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/address
es/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"
       }
 }
```


Element	Type	Description
`enabled`	boolean	By default, this option is set to `false`. Set it to `true` to enable the cluster interface.
`listen_address`	Top level element (string)	Contains the key of the network interface that is used as the cluster interface.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 encypt the backup, see Encrypting system backup policy.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 more information on the `meta` object, see Message format on page 10.


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


Element	Description
`backup_policy`	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.

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 41.

Code	Description	Notes
401	Unauthenticated	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`

Code	Description	Notes
		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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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/exported_configuration_encryption
```

Response

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

For more information on 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-----
\nmQGNBF3rnZ0BDADHdz5/kCkrl7T8w861AGGXdGK/lwxunTCx6tfhSsFREWmKjhfr\nYTLNxsodALXt
AphHeNAeUWwXjYDJelAlMVcDrVtLp7Ht8tqnmNt2NWUSmfFIF3ga\nD1OsH2UjT5Xt6XAjKvFfWeHSxk
S0QHIcLfUT5WDoUcTEsR8jEdj80A7Z6.......

CT1WwbMg5VoXQ3Rpp8evcUTzy3+ra/GosCSaFSrE31pyXkULB9+EAU7W\n23YDiM21csIaqX+XDGMex5
Hq4PMhO7cqSMyB\n=j20J\n-----END PGP PUBLIC KEY BLOCK-----\n",
                            "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"
       }
}
```

Elements of encryption	Type	Description
encryption	top-level element	Defines encryption settings for system backups.
enabled	boolean	When set to True, enables encryption of the system backups. Enabling encryption requires setting the `gpg_public_key` option.
gpg_public_key	JSON object	Contains the `fingerprint`, `public_key`, and the list of `uids` of the GPG public key used to encrypt system backups. For example:

```
"gpg_public_key": {
    "fingerprint": "2F2E3967EDAD2F288E54EE8693B99C4F545B7670",
    "public_key": "-----BEGIN PGP PUBLIC KEY BLOCK-----
\nmQGNBF3rnZ0BDADHdz5/kCkrl7T8w861AGGXdGK/l-
wxunTCx6tf-
hSsFREWmKjh-
fr\nYTLNxs-
odALXtAphHeNAeUWwXjYDJelAlMVcDrVtLp7Ht8tqn-
mNt2NWUSm-
fFIF3ga\nD1OsH2UjT5Xt6XAjKvFfWeHSxkS0QHIcLfUT5WDoUcTEsR8jEd-
j80A7Z6hKyF29g\...

R40Niv4Ge6aYneDp-
k3yTBco6bBYDR7NKA7OREXCfqcyCeYB121UQ\n-
bb5aTZAaW8D8IRmy-
bxpRxRAaHZX0apBgDLKwWf48kLOnOC9O7hgcyY1spZgTGz7i\nTryxlBl/CT1Ww-
```

Elements of encryption	Type	Description

```
bMg5VoXQ3Rp-
p8evcUTzy3+ra/GosCSaFSrE31pyXkULB9+EAU7W\n23YDiM21c-
sIaqX+XDGMex5Hq4PMhO7cqSMyB\n=j20J\n-----END PGP PUBLIC KEY BLOCK---
--\n",
    "uids": [
        "Demo User <example@oneidentity.com>"
    ]
}
```

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 endpoints for configuring date and time settings on SPS.

```
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 more information on 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",
```

```
    "href": "/api/configuration/datetime",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/http",
    "parent": "/api/configuration",
    "previous": "/api/configuration/alerting",
    "transaction": "/api/transaction"
  }
}
```

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

NTP servers

This endpoint contains NTP server addresses.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "body": [
    {
      "selection": "fqdn",
      "value": "pool.ntp.org"
    }
  ],
  "key": "ntp_servers",
  "meta": {
    "first": "/api/configuration/datetime/ntp_servers",
    "href": "/api/configuration/datetime/ntp_servers",
    "last": "/api/configuration/datetime/timezone",
    "next": "/api/configuration/datetime/timezone",
```

```
    "parent": "/api/configuration/datetime",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

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

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 32.

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 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Timezone

Configures the time zone.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 displays the configured time zone.

```
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 more information on 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"
    }
 }
```

Element	Type	Description
key	string	Top level element, contains the ID of the endpoint.
body	string	Contains the configured time zone. Possible values are:

```
Africa/Abidjan
Africa/Accra
Africa/Addis_Ababa
Africa/Algiers
Africa/Asmara
Africa/Asmera
Africa/Bamako
Africa/Bangui
Africa/Banjul
Africa/Bissau
Africa/Blantyre
Africa/Brazzaville
Africa/Bujumbura
Africa/Cairo
```

Element	Type	Description

		`Africa/Casablanca`
		`Africa/Ceuta`
		`Africa/Conakry`
		`Africa/Dakar`
		`Africa/Dar_es_Salaam`
		`Africa/Djibouti`
		`Africa/Douala`
		`Africa/El_Aaiun`
		`Africa/Freetown`
		`Africa/Gaborone`
		`Africa/Harare`
		`Africa/Johannesburg`
		`Africa/Kampala`
		`Africa/Khartoum`
		`Africa/Kigali`
		`Africa/Kinshasa`
		`Africa/Lagos`
		`Africa/Libreville`

Element	Type	Description

		`t`
		`Africa/Ouagadougou`
		`Africa/Porto-Novo`
		`Africa/Sao_Tome`
		`Africa/Timbuktu`
		`Africa/Tripoli`
		`Africa/Tunis`
		`Africa/Windhoek`
		`America/Adak`
		`America/Anchorage`
		`America/Anguilla`
		`America/Antigua`
		`America/Araguaina`
		`America/Argentina/Buenos_Aires`
		`America/Argentina/Catamarca`
		`America/Argentina/ComodRivadavia`
		`America/Argentina/Cordoba`
		`America/Argentina/Jujuy`
		`America/Argentina/La_Rioja`
		`America/Argentina/Mendoza`

Element	Type	Description

		`America/El_Salvador`
		`America/Ensenada`
		`America/Fort_Wayne`
		`America/Fortaleza`
		`America/Glace_Bay`
		`America/Godthab`
		`America/Goose_Bay`
		`America/Grand_Turk`
		`America/Grenada`
		`America/Guadeloupe`
		`America/Guatemala`
		`America/Guayaquil`
		`America/Guyana`
		`America/Halifax`
		`America/Havana`
		`America/Hermosillo`
		`America/Indiana/Indianapolis`
		`America/Indiana/Knox`
		`America/Indiana/Marengo`
		`America/Indiana/Petersburg`
		`America/Indiana/Tell_City`
		`America/Indiana/Vevay`
		`America/Indiana/Vincennes`

Element	Type	Description
		a/Baghdad Asia/Bahrain Asia/Baku Asia/Bangkok Asia/Beirut Asia/Bishkek Asia/Brunei Asia/Calcutta Asia/Choibalsan Asia/Chongqing Asia/Chungking Asia/Colombo Asia/Dacca Asia/Damascus Asia/Dhaka Asia/Dili Asia/Dubai Asia/Dushanbe Asia/Gaza Asia/Harbin Asia/Hong_Kong Asia/Hovd Asia/Irkutsk Asia/Istanbul Asia/Jakarta Asia/Jayapura Asia/Jerusalem Asia/Kabul Asia/Kamchatka Asia/Karachi Asia/Kashgar

Element	Type	Description

Modify the time zone

To modify time zone, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Logs, monitoring and alerts

Management options

Contains the configuration endpoints for managing SPS.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 management configuration endpoints.

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

Response

The following is a sample response received when listing management endpoints.

For more information on 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"
      }
    },
    {
      "key": "splunk_forwarder",
      "meta": {
        "href": "/api/configuration/management/splunk_forwarder"
      }
    },
    {
      "key": "support_info",
      "meta": {
        "href": "/api/configuration/management/support_info"
      }
    },
    {
      "key": "syslog",
      "meta": {
        "href": "/api/configuration/management/syslog"
      }
    },
    {
      "key": "system_backup",
      "meta": {
        "href": "/api/configuration/management/system_backup"
      }
    },
    {
      "key": "universal_siem_forwarder",
      "meta": {
        "href": "/api/configuration/management/universal_siem_forwarder"
      }
    },
    {
      "key": "web_gateway_authentication",
      "meta": {
        "href": /api/configuration/management/web_gateway_authentication"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa",
    "href": "/api/configuration/management",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/network",
    "parent": "/api/configuration",
    "previous": "/api/configuration/local_services",
    "transaction": "/api/transaction"
```

```
  }
}
```

Endpoints	Description
certificates	References the certificates of SPS's internal Certificate Authority, Timestamping Authority, and the SSL certificate of the web interface.
disk_fillup_prevention	Disk fill-up prevention.
email	SMTP server address and authentication, administrator e-mail, and e-mail addresses for alerts and reports.
exported_configuration_encryption	SMTP server address and authentication, administrator e-mail, and e-mail addresses for alerts and reports.
health_monitoring	Configuration settings for monitoring the utilization of SPS.
snmp	SNMP settings.
syslog	Syslog server address and authentication.
web_gateway_authentication	Configuration of the banner on the login screen before web gateway authentication.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Web gateway authentication

Use the `/web_gateway_authentication` endpoint to configure tha 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

```
PUT https://<IP-address-of-SPS>/api/configuration/management/web_gateway_
authentication
```

Cookies

Cookie name	Description	Required	Values
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 more information 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 with the `/web_gateway_authentication` endpoint include:

Operation	HTTP method	URL
Configuring the web gateway authentication banner	PUT	/api/configure/management/web_gateway_authentication
Querying the web gateway authentication banner	GET	/api/configure/management/web_gateway_authentication

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 more information on the `meta` object, see Message format on page 10.

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

Elements of the response message body include:

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

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 41.

Code	Description	Notes
400	InvalidRequestBody	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.
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

Code	Description	Notes
		applied. The details section contains the errors that were found in the configuration.
401	Unauthenticated	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.
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.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
  },
  "key": "syslog",
  "meta": {
    "first": "/api/configuration/management/certificates",
    "href": "/api/configuration/management/syslog",
    "last": "/api/configuration/management/webinterface",
    "next": "/api/configuration/management/webinterface",
```

```
    "parent": "/api/configuration/management",
    "previous": "/api/configuration/management/snmp",
    "transaction": "/api/transaction"
  }
}
```

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 filed 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.
	receivers		Top level list	Contains the addresses of the syslog servers.

Element	Type	Description
`server_key_check`	string	Configures validating the syslog server's certificate with the CA. The following values are possible: • `optional-trusted` 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. • `optional-untrusted` SPS accepts any certificate shown by the server. • `required-trusted` SPS verifies the certificate shown by the server. • `required-untrusted` 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.

Elements of `receivers`	Type	Description
`address`	Top level item	Contains the address of the syslog server.
`selection`	string	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 IP address.

Elements of `receivers`		Type	Description
	`value`	string	The address of the syslog server, corresponding to the format set in the `selection` field.
`port`		int	The port of the syslog server.
`protocol`		Top level item	Contains the syslog protocol settings.
	`ip_protocol`	string	Configures the IP protocol. The following options are available: • `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.

```
{
    "certificates": {
       "ca": null,
       "client": null
    },
    "include_node_id": true,
    "receivers": [],
    "server_key_check": "optional-untrusted"
 }
```

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-----
MIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC
Q0ExEDAOBgNVBAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT
B0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi
aXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4XDTE2MDQyMjE2
MDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP
bnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD
VQQLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN
AQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS
59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw
0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc
3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs
X5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX
hJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
R5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP
AB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy
0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7XDB
uNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc
N5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt
qYyyD1ld89OUEYZ+aJQd/A==
-----END CERTIFICATE-----
```


The same certificate, as accepted by the SPS API:


```
"certificate": "-----BEGIN CERTIFICATE-----
\nMIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC\nQ0ExEDAOBgNV
BAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDU
RvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxh
Yml0Lmh1MB4XDTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDg
YDVQQIEwdP\nbnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQ
QLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxA
YmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7a
hTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY
6vDU9/4v9dDiQfEPlauw\n0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2
oSobPc\n3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv
0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4di
KU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh
6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57
```

```
IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7X
DB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc\nN5+4ImYnFN
xSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyyD1ld89OUEYZ+aJQd/A==
\n-----END CERTIFICATE-----\n"
```


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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
  }
}
```

Element	Type	Description
key	string	Top level element, contains the ID of the endpoint.
body	Top level element (string)	Contains the configuration settings for disk fill-up prevention.
archiving_enabled	boolean	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.
enabled	boolean	Set to `true` to enable disk fill-up prevention.
used_space_ratio_limit	int	Disk utilization limit, in percent. When used disk space reaches this limit, SPS disconnects all clients. Set to 0 to turn the feature off.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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

Cookie name	Description	Required	Values
			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 more information on 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"
    }
 }
```

Element			Type	Description
key			string	Top level element, contains the ID of the endpoint.
body			Top level element (string)	Contains the configuration options for e-mail.
	admin_address		string	The e-mail address of the administrator of SPS.
	alerting_address		string	The e-mail address where monitoring alerts are sent.
	reporting_address		string	The e-mail address where traffic reports are sent.
	sender_address		string	The address of the sender (SPS).
	smtp_auth		Top level item	Configures authentication to the SMTP server.
		enabled	boolean	Set to true to enable authenticating to the SMTP server.
		password	string	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).
		username	string	The username for authenticating to the SMTP server.
	smtp_encryption		Top level item	Configuration settings for encrypting the communication between SPS and the SMTP server.
	smtp_server		Top level item	Contains the address of the SMTP server.
		selection	string	Defines the address type (IP or domain name). Possible values are: • fqdn The SMTP server address is

Element	Type	Description
		provided as a fully qualified domain name. • `ip` The SMTP server address is provided as an IP address.
`value`	string	The address of the SMTP server.

Elements of `smtp_encryption`	Type	Description
`client_authentication`	Top level item	Configures authenticating as a client with an X.509 certificate. Can only be enabled if the value of the `selection` element is set to `starttls`.
`enabled`	boolean	Set to `true` to enable authenticating as a client with an X.509 certificate. Can only be enabled if the value of the `selection` element of `snmp_encryption` is set to `starttls`.
`x509_identity`		References the identifier of the authenticating client'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). For details, see Certificates stored on SPS on page 300.
`selection`	string	Configures encrypted communication with the SMTP server. The following values are possible: • `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`.

Elements of `smtp_encryption`	Type	Description
`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 `snmp_encryption` is set to `starttls`.
`server_certificate_ca`	string	The CA certificate of the Certificate Authority.

Examples:

Enable authentication to the SMTP server.

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

Configure STARTTLS encryption without certificate checks.

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


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


```
{
  "admin_address": "<admin-email>",
  "alerting_address": "<alerts-target-email>",
  "reporting_address": "<reports-target-email>",
  "sender_address": null,
  "smtp_auth": {
    "enabled": true,
    "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-----
MIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC
Q0ExEDAOBgNVBAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT
B0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi
aXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4XDTE2MDQyMjE2
MDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP
bnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD
VQQLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN
AQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS
59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw
0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc
3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs
X5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX
hJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
R5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP
AB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy
0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7XDB
uNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc
N5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt
qYyyD1ld89OUEYZ+aJQd/A==
-----END CERTIFICATE-----
```

The same certificate, as accepted by the SPS API:


```
"certificate": "-----BEGIN CERTIFICATE-----
\nMIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC\nQ0ExEDAOBgNV
BAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDU
RvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxh
Yml0Lmh1MB4XDTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDg
YDVQQIEwdP\nbnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQ
QLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxA
YmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7a
hTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY
6vDU9/4v9dDiQfEPlauw\n0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2
oSobPc\n3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv
0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4di
KU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh
6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57
IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7X
DB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc\nN5+4ImYnFN
xSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyyD1ld89OUEYZ+aJQd/A==
\n-----END CERTIFICATE-----\n"
```


Modify mail settings

To modify mail settings, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 description of the available parameters listed in Element .

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Health monitoring

Configuration settings for monitoring the utilization of SPS.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 health monitoring settings.

```
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 more information on the `meta` object, see Message format on page 10.


```
{
    "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"
    }
 }
```


Element	Type	Description
key	string	Top level element, contains the ID of the endpoint.
body	Top level element (string)	Contains health monitoring settings.
`maximum_disk_utilization_ratio`	int	The highest allowed value for disk utilization (in %).
`maximum_load1`	int	Average maximum for load for 1 minute.
`maximum_load15`	int	Average maximum load for 15 minutes.
`maximum_load5`	int	Average maximum load for 5 minutes.
`maximum_swap_utilization_ratio`	int	The highest allowed value for swap utilization (in %).

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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
`trap`	Configuration settings for 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	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

Cookie name	Description	Required	Values
session_	Contains the	Required	The value of the session ID cookie received

Cookie name	Description	Required	Values
id	authentication token of the user		from the REST server in the authentication response, for example, **a1f71d030e657634730b9e887cb59a5e56162860**. For more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
  "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"
  }
}
```

Element	Type	Description
`key`	string	Top level element, contains the ID of the endpoint.
`body`	Top level element (string)	Contains the address and protocol settings of the SNMP server.
`enabled`	boolean	Set to `true` to send alerts to an SNMP server.
`version`	Top level item	Contains the configuration settings for the server address, and the SNMP protocol.

Elements of `version`	Type	Description
`selection`	string	Defines the SNMP protocol to use. Possible values are: • `2c` Configures version 2c of the SNMP protocol. • `3` Configures version 3 of the SNMP protocol.
`value`	Top level item	Contains the SNMP server address, and the protocol-specific settings.
`auth_method`	string	Required parameter when using SNMP version 3. Configures encrypted communication with the SNMP server. Possible values are: • `md5`: Use MD5 encryption. The `auth_password` element must reference a valid password. • `sha1`: Use SHA1 encryption. The `auth_password` element must reference a valid password.

Elements of `version`	Type	Description
`auth_password`	string	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 `password` 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 `(!"#$%&'()*+,;<=&@[\]^`{|}_./:?-)` and the space character.
`community`	string	Must be used if version `2c` of the SNMP protocol is configured. The name of the SNMP community.
`encryption_method`	string	Must be used if version `3` of the SNMP protocol is configured. Configures encrypted communication with the SNMP server. Possible values are: • `none`: No encryption. The value of the `encryption_password` element must also be set to `null`. • `aes`: AES encryption. The `encryption_password` element must reference a valid password. • `des`: DES encryption. The `encryption_password` element must reference a valid password.
`encryption_password`	string	Must be used if version `3` of the SNMP protocol is configured.

Elements of `version`			Type	Description
				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 `(!"#$%&'()*+,;<=&@[\]^`{|}_./:?-)` and the space character.
	`engine_id`		string	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).
	`server`		top level item	Contains the IP address of FQDN of the SNMP server.
		`selection`	string	Defines the address type (IP or domain name). Possible values are: • `fqdn` The SNMP server address is provided as a fully qualified domain name. • `ip` The SNMP server address is provided as an IP address.
		`value`	string	The address of the SNMP server.
	`username`		string	Must be used if version `3` of the SNMP

Elements of `version`	Type	Description
		protocol is configured. The username for sending SNMP traps.

Examples:

Configure a server with the SNMP v2c protcol.

```
{
    "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.

```
{
  "enabled": true,
  "version": {
    "selection": "3",
    "value": {
      "auth_method": "md5",
      "auth_password": {
        "key": "d21f3675-8dff-43c5-a982-17839390a6b3",
        "meta": {
          "href": "/api/configuration/passwords/d21f3675-8dff-43c5-a982-
17839390a6b3"
        }
      },
      "encryption_method": "none",
      "encryption_password": null,
      "engine_id": "<0x0123456789ABCDEF>",
      "server": {
        "selection": "ip",
        "value": "<server-ip>"
```

```
      },
      "username": "<username>"
    }
  }
}
```

Configure a server with the SNMP v3 protocol, SHA1 authentication, and AES-encrypted communication.

```
{
    "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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
    "body": {
        "access_restriction": {
            "enabled": false
        },
        "enabled": true,
        "listen": [
            {
                "address": {
                    "key":
"nic1.interfaces.ff7574025754b3df1647001.addresses.1",
                    "meta": {
                        "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/address
es/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"
    }
}
```

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 SNMP agent.
	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.
	enabled		boolean	Enables the SNMP server. If this option is set to `False`, SPS ignores every other option of this endpoint.
	listen		list	Selects the network interface, IP address, and port where the clients can access the web interface.
		address	JSON object	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:

```
{
    "body": {
        "interfaces": {
            "@order": [
                "ff7574025754b3df1647001"
            ],
            "ff7574025754b3df1647001": {
                "addresses": {
                    "1": "10.40.255.171/24",
                    "@order": [
                        "1"
                    ]
```

Element	Type	Description

```
                },
                "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.

```
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.in-
terfaces.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.in-
terfaces.ff7574025754b3df1647001.addresses.1",
    "meta": {
```

Element		Type	Description
			"href": "/api/-config-uration/net-work/n-ics/n-ic1#interfaces/ff7574025754b3df1647001/addresses/1" } },
	port	integer	The port number where this local service accepts connections.
system_contact		string	Optional. For example, it can contain the contact information of the SPS administrator.
system_description		string	Optional. For example, it can contain information of the SPS host.
system_description		string	Optional. For example, it can contain the location of the SPS appliance.
version_2c		JSON object	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: "version_2c": { "community": "mycommunity", "enabled": true },
	community	string	Optional. Specifies the community to use.
	enabled	boolean	Optional. Enables SNMP queries using the SNMP v2c protocol.
version_3		JSON object	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: "version_3": {

Element	Type	Description

```
    "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"
        }
    ]
}
```

Elements of `version_3`	Type	Description
`enabled`	boolean	Optional. Enables SNMP queries using the SNMP v2c protocol.
`users`	JSON object	Contains the configuration parameters for the SNMP v3 protocol.
`auth_method`	string	Required parameter when using SNMP version 3. Configures encrypted communication with the SNMP server. Possible values are: • `md5`: Use MD5 encryption. The `auth_password` element must reference a valid password. • `sha1`: Use SHA1 encryption. The `auth_password` element must reference a valid password.
`auth_`	string	Required parameter when using SNMP version 3.

Elements of `version_3`	Type	Description
`password`		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 `(!"#$%&'()*+,;<=&@[\]^`{|}_./:?-)` and the space character.
`encryption_method`	string	Configures encrypted communication with the SNMP server. Possible values are: • `none`: No encryption. The value of the `encryption_password` element must also be set to `null`. • `aes`: AES encryption. The `encryption_password` element must reference a valid password. • `des`: DES encryption. The `encryption_password` element must reference a valid password.
`encryption_password`	string	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 `(!"#$%&'()*+,;<=&@[\]^`{|}_./:?-)` 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Alerting

Contains the endpoints for configuring alerting on SPS.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
    }
 }
```

Element	Description
`system_alerts`	Configuration options for system-related alerts.
`traffic_alerts`	Configuration options for traffic-related alerts.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 endoint.

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

Cookie name	Description	Required	Values
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 more information 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 configuration options for system-related alerts.

```
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 more information on 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"
  }
}
```


Element		Type	Description
key		string	Top level element, contains the ID of the endpoint.
body		Top level element (string)	Contains the configuration options for system-related alerts.
xcbAlert		Top level item	General alert.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbArchiveFailed		Top level item	Data archiving failure.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbBackupFailed		Top level item	Data and configuration backup failure.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbBruteForceAttempt		Top level	Too many successive failed login attempts.

Element		Type	Description
		item	
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbConfigChange		Top level item	Configuration change.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbDBError		Top level item	Database error occured.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbDiskFull		Top level item	Disk utilization reached the percentage configured in the maximum_disk_utilization_ratio element of the api/configuration/management/monitoring endpoint.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbError		Top level item	General error.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbFirmwareTainted		Top level item	The firmware is tainted.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbHWError		Top level item	Hardware error.
	email	boolean	Set to true to enable e-mail alerts.

Element		Type	Description
	snmp	boolean	Set to true to enable SNMP alerts.
xcbHaNodeChanged		Top level item	HA node state changed.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbLicenseAlmostExpired		Top level item	License expires soon.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbLimitReached		Top level item	License limit reached.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbLoadAvgHigh		Top level item	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 endpoint.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbLogin		Top level item	Successful login.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
xcbLoginFailure		Top level item	Failed login.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.

Element		Type	Description
`xcbLogout`		Top level item	Logout from the web configuration interface.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`xcbRaidStatus`		Top level item	RAID status changed.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`xcbSwapFull`		Top level item	The utilization of the swap exceeded the value configured in the `maximum_swap_utilization_ratio` element of the api/configuration/management/monitoring endpoint.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`xcbTimeSyncLost`		Top level item	Time sync lost.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`xcbTimestampError`		Top level item	Time stamping error.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 endoint.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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
    },
    "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"
  }
}
```


Element		Type	Description
key		string	Top level element, contains the ID of the endpoint.
body		Top level element (string)	Contains the configuration options for traffic-related alerts.
scbAuthFailure		Top level item	User authentication failed.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbAuthSuccess		Top level item	Successful user authentication.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbChannelDenied		Top level item	Channel opening denied.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbConnectionDenied		Top level	Connection denied.

Element		Type	Description
		item	
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbConnectionFailed		Top level item	Connection to the server failed.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbConnectionTimedout		Top level item	Connection timed out.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbCredStoreClosed		Top level item	The requested credential store is closed.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbCredStoreDecryptError		Top level item	Failure to decrypt a credential.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.
scbCredStoreUnlockFailure		Top level item	Failure to unlock the credential store.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.

Element		Type	Description
`scbGWAuthFailure`		Top level item	The user failed to authenticate on the gateway.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`scbGWAuthSuccess`		Top level item	Successful authentication on the gateway.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`scbProtocolViolation`		Top level item	Protocol violation.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`scbRealTimeAlert`		Top level item	Real-time audit event detected.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`scbSshHostKeyLearned`		Top level item	New SSH host key learned.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.
`scbSshHostKeyMismatch`		Top level item	SSH host key mismatch.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.

Element		Type	Description
`scbUserMappingFailure`		Top level item	User mapping failed on the gateway.
	`email`	boolean	Set to `true` to enable e-mail alerts.
	`snmp`	boolean	Set to `true` to enable SNMP alerts.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 with the `trust_stores` endpoint include:

Operation	HTTP method	URL	Notes
Create a	POST	`/api/configuration/trust_`	The name of the trust store must

Operation	HTTP method	URL	Notes
trust store		stores	be unique.
List trust stores	GET	/api/configuration/trust_stores	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): • /pages/starlingjoin • /config/xcb/aaa/settings • /config/scb/pol_ldaps
Query a trust store	GET	/api/configuration/trust_stores/<id of the trust store>	
Query the built-in trust store	GET	/api/configuration/trust_stores/-7001	
Update a trust store	PUT	/api/configuration/trust_stores/<id of the trust store>	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 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.
Delete a trust store	DELETE	/api/configuration/trust_stores/<id of the trust store>	

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 more information on 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-XXXXXXXXXXXX",
            "meta": {
              "href": "/api/configuration/trust_stores/XXXXXXXX-XXXX-XXXX-XXXX-
XXXXXXXXXXXX"
            },
            "body": {
              "name": "My_Custom_Trust_Store",
              "authorities": [
                {
                  "fingerprint": {
                    "digest":
"01:25:1f:a2:df:2a:31:1a:29:7a:ba:43:c4:03:42:a5:d7:30:ec:2d:e0:d7:7a:72:a7:1b:
c3:99:c5:6c:10:ea",
                    "hash_algorithm": "sha256"
                  },
                  "issuer": "C=HU/ST=Budapest/L=None/O=Internet Widgits Pty
Ltd/OU=None/CN=None/emailAddress=None",
                  "pem": "-----BEGIN CERTIFICATE-----
\nMIIDZzCCAk+gAwIBAgIUMlI5+EgTDAh2zqRDGYrzFRyozI8wDQYJKoZIhvcNAQEL\nBQ
AwQzELMAkGA1UEBhMCSFUxETAPBgNVBAgMCEJ1ZGFwZXN0MSEwHwYDVQQKDBhJ\n
```

```
bnRlcm5ldCBXaWRnaXRzIFB0eSBMdGQwHhcNMTQwODEyMTIzNjQ4WhcNMzQwNjE4\n
MTIzNjQ4WjBDMQswCQYDVQQGEwJIVTERMA8GA1UECAwIQnVkYXBlc3QxITAfBgNV\nB
AoMGEludGVybmV0IFdpZGdpdHMgUHR5IEx0ZDCCASIwDQYJKoZIhvcNAQEBBQAD\ngg
EPADCCAQoCggEBALffJBDD6A/ZGBTgFbyLXHulU+hGnMW3DoPo2q4HY1/FfbkS\nrzmK
+Fiz+3EwJCWi+EwK9mqve/nh6YRRw/VaAVQ7CkA7f7to+I7gP647Bq1wk0lh\nBVEJNlN0
jfYYSumGxzPotw/fon1MkXuMbLc0Pr/vFX3NQC7/STAV5dZFcdboXDA7\nZZ3rzBIr93ThO
bsGj01MRO6wrS3rfE7Px9D7C2u9YSkP3OQ1Sfm/jqyLNaT6xt4i\nhrLnfYEc8mClnrlvILi+q
/D6mIUSjb4IGvergAyl4jgPjO02UcvBzOIA9tDlBJBi\nQxZx+T620ubmEwOl9Q0G8RAWKz
7szrBcXEjXhYUCAwEAAaNTMFEwHQYDVR0OBBYE\nFCDfEeq5Hsm8jMrG110iNpt5cikTM
B8GA1UdIwQYMBaAFCDfEeq5Hsm8jMrG110i\nNpt5cikTMA8GA1UdEwEB/wQFMAMBAf8
wDQYJKoZIhvcNAQELBQADggEBAK3iizM4\nCx69YD+4CWOUswULrCJA38C+nDYONLbN
kact8JKXqCn/MaZTII+dZoV9RjjX4AzA\nPTQkZT+RoVeCZyt+qWHMdjq6koabXwQmXNo
zUtaxEZTrnoUDEWtNIbjV/gNtRcSG\nsU7i9L2YnwDzTw0cR/pu1Hykq8fwqNqjQGYnmXtJ
spMkKAtVe1CrtnPLiC6JBr0g\n5GZF58sHx5+gO0RkqdzJgRAGnImdfAahqfHmKRFmxoxW
LyylRyqDgQ+KqcaDvZI+\ni36M+NQHVrDX4jo4CFoXhFlSOepvtDOpmzoWhugwDNMPuU
1IEY7//CJBXQnjp+uf\nLO6PsNmMKDGi9Dk=\n-----END CERTIFICATE-----\n",
                    "subject": "C=HU/ST=Budapest/L=None/O=Internet Widgits Pty
Ltd/OU=None/CN=None/emailAddress=None"
                  }
                ],
                "crl_urls": [
                  "http://crl.it/sec"
                ],
                "revocation_check": "full",
                "trust_store_type": "custom"
              }
            }
          ]
        }
```


Elements of the response message body include:

Elements of items		Type	Description	Notes
`items`		object array	List of JSON objects available from the current endpoint.	
	`key`	string	The ID of the trust store.	Each trust store has a unique key. The built-in trust store's ID is "-7001".
	`meta`	string (uri)	The href field contains the URL of the trust store.	
	body			

Elements of body		Type	Description	Notes
body		object	Top level element.	
	name	string	The name of the trust store.	The name field is set by the user and it must be unique. For example: "name": "My_Custom_Trust_Store". The built-in trust store's name is "Built-in".
	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 CAs that are part of the chain of a given certificate which is being verified. "leaf" - The crl_urls field must contain at least the CRL URL of the CA which signed the certificate which is being validated. "none" - The crl_urls field must be empty.
	trust_store_type	enum	The type of the trust store.	Possible values: "built-in", "custom". The built-in trust store comes with the operation system. This

Elements of body	Type	Description	Notes
			type of trust store is read-only. There is no CRL check involved, and it cannot be removed.

Elements of authorities		Type	Description	Notes
`authorities`		array	List of Certificate Authorities.	
	`fingerprint`			
	`issuer`	string	The name of the entity that signed the certificate.	
	`pem`	string	The certificate in PEM format.	
	`subject`	string	The subject of the certificate.	

Elements of fingerprint		Type	Description	Notes
`fingerprint`			A two-piece byte sequence consisting of a hash algorithm and a message digest.	
	`digest`	string	The string of digits produced by the hash algorithm.	
	`hash_algorithm`	string	The name of the hash algorithm.	

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 41.

Code	Description	Notes
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:

Code	Description	Notes
		• 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.
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. Possible semantic error messages related to trust store: • The name of the trust stores must be unique. • The authorities of a trust store must be unique. • The CRL URLs of a trust store must

Code	Description	Notes
		be unique.
401	Unauthenticated	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.
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.

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

Cookie name	Description	Required	Values
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 more information 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 /starling endpoint include:

Operation	HTTP method	URL	Notes
Query One Identity Starling Join without a join to One Identity Starling	GET	`/api/configuration/starling`	The value of the join_info field is `null`.
Query One Identity Starling Join after join	GET	`/api/configuration/starling`	The values of the `join_info` field are `environment`, `product_instance`, and `product_tims`.
Enable SRA	PUT	`/api/configuration/starling`	SRA can be enabled only if the node is joined to One Identity Starling. Use the `starling/join` endpoint to join to One Identity Starling.
Disable SRA	PUT	`/api/configuration/starling`	To disable SRA to One Identity Starling, the enabled field must be set to `false`.

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 32.

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

Sample request

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

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

```
{
     "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:

Elements	Type	Description	Notes
`join_info`	object		
join_info.environment	string	The environment of the product.	Possible values: `prod`.
`join_info.product_instance`	string	The instance of the product.	
`join_info.product_tims`	string	The TIMS license of the product.	
`remote_access`			To disable SRA to One Identity Starling, the **`enabled`** field

Elements	Type	Description	Notes
			must be set to `false`.
`remote_access.enabled`	boolean	Enables or disables SRA to One Identity Starling.	Possible values: `true`, `false`
`key`			Possible values: `starling`

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

Response

The response is a regular meta object.

For more information on 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 41.

Endpoint-specific HTTP response codes for this request:

HTTP response code	Status / Error	Description
403	RemoteAccessDependsOnStarlingJoinError	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.
403	StarlingJoinInfoIsReadOnlyError	You cannot join or unjoin from 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

Managing Starling Join

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

Retrieving One Identity Starling Join information

Check whether your SPS appliance is joined to the One Identity Starling platform.

If you are interested which One Identity Starling services are available to you, you can list them at the Retrieving the status of services related to Starling Join/Unjoin on page 215 endpoint.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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

HTTP GET operations with the /starling/join endpoint include:

Operation	HTTP method	URL	Notes
Querying Starling Join endpoint from a browser without TIMS	GET	`/api/starling/join`	Headers to be used: `Accept: text/html`.
Querying Starling Join endpoint when joined from a browser without TIMS			
Querying Starling Join info without TIMS			
Querying Starling Join info when joined without TIMS			
Querying Starling Join endpoint from a browser with TIMS	GET	`/api/starling/join?product_tims=<TIMS value>`	Example of TIMS value: `product_tims=222-333-444` .
Querying Starling Join endpoint when joined from a browser with TIMS			
Querying Starling Join info with			

Operation	HTTP method	URL	Notes
TIMS			
Querying Starling Join info when joined with TIMS			

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 more information on 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:

Elements	Type	Description	Notes
`joined`	boolean	Displays whether or not the user is joined to One Identity Starling or unjoined from it.	
`product_instance`	string	The instance of the product.	
`product_name`	string	The name of the product.	
`product_tims`	string	The TIMS license of the product.	

NOTE: While it is possible to send a POST HTTP request to the `/starling/join` endpoint, if you want to join your SPS appliance to One Identity Starling, visit the SPS web interface and initiate the join process under **Basic Settings** > **Starling Integration** > **Start join**.

Unjoining SPS from One Identity Starling

Use the `/starling/join` endpoint to unjoin your SPS appliance from One Identity Starling.

URL

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

Operations

HTTP DELETE operations with the `/starling/join` endpoint include:

Operation	HTTP method	URL	Notes
Unjoining One Identity Starling	DELETE	`/api/starling/join`	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.
Force unjoining One Identity Starling	DELETE	`/api/starling/join?force=true`	The Force Unjoin functionality works even when SRA is enabled.

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.

HTTP response code	Status / Error	Description
`400`	MissingCredentialStringError	The mandatory `credential_string` parameter is not specified for One Identity Starling Join.

HTTP response code	Status / Error	Description
400	MissingProductInstanceError	The mandatory `product_instance` parameter is not specified for One Identity Starling Join.
403	OpenTransactionError	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.
403	ForbiddenActionError	Forbidden action. To unjoin from One Identity Starling, use the `/starling/join` endpoint.
403	StarlingJoinIsInUseByRemoteAccessError	Unjoining One Identity Starling is not allowed while One Identity Safeguard Remote Access (SRA) is in use. Disable SRA in the configuration before unjoining from One Identity Starling.

For more information and a list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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 more information 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 returns information about the availability of the services needed for Starling Join, or Starling Unjoin .

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

Response

The following is a sample response.

For more information on the `meta` object, see Message format on page 10.

```
{
   "internet_connection": {
     "error": "N/A",
     "status": true
   },
   "meta": {
     "href": "/api/starling/status",
     "parent": "/api/starling",
     "remaining_seconds": 593
   },
   "proxy": true,
   "starling_status": {
     "Account Email Notifications": "operational",
     "Account Services Portal": "operational",
     "Active Roles On Demand": "operational",
     "CertAccess": "operational",
     "Connect": "operational",
```

```
    "Database Service": "operational",
    "Hybrid Subscription": "operational",
    "Identity Manager On Demand": "operational",
    "Job Service": "operational",
    "One Identity Starling": "operational",
    "Password Manager On Demand": "operational",
    "Safeguard On Demand": "operational",
    "Safeguard Remote Access": "operational",
    "Safeguard for Privileged Passwords On Demand": "operational",
    "Safeguard for Privileged Sessions On Demand": "operational",
    "Subscription and Billing": "operational",
    "Two-Factor Authentication": "operational",
    "Web UI": "operational"
  },
  "verdict": true
}
```

Elements of the response message body include:

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

Elements	Type	Description	Notes
			the status of the One Identity Starling service is relevant.
`verdict`	boolean	The value is `true` if SPS can connect to the status page, a proxy is enabled, and the relevant service(s) are operational.	

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 list of standard HTTP response codes, see Application level error codes on page 41.

4

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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",
```

```
    "parent": "/api/configuration",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 `/aaa/settings` endpoint include:

Operation	HTTP method	URL	Notes
Creating login settings	GET	`/api/configuration/aaa/settings`	
Retrieving login settings	POST		

Sample request

The following command lists the login settings.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
        "key": "settings",
        "body": {
           "authentication_banner": "",
           "bruteforce_protection": {
              "attempt_limit": 20,
              "lockout_minutes": 10
           },
           "webinterface_timeout": 10
        }
     }
```

Elements of the request message body include:

Element	Type	Description	Notes
`authentication_banner`	string	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.	If you specify an empty string, then no authentication banner will show.
`bruteforce_protection`	JSON object	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.	
`bruteforce_protection.attempt_limit`	number	The number of unnsuccessful 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

Element	Type	Description	Notes
`bruteforce_protection.lockout_minutes`	number	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.	Value range: 1-720 minutes Default value: 10 minutes
`webinterface_timeout`	number	The period of inactivity after which SPS terminates the web session of a user. It is measured in minutes.	Value range: 5-720 minutes Default value: 10 minutes

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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 more information on authentication, see Authenticate to the SPS REST API on page 19. NOTE: This session ID refers to the connec-

Cookie name	Description	Required	Values
			tion 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 more information on 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"
```

```
        },
        {
            "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"
        },
        {
            "group": "ica-write",
            "objects": [
                "/special/ica"
            ],
            "permission": "write"
        },
        {
            "group": "http-view",
            "objects": [
                "/special/http"
            ],
            "permission": "read"
        },
        {
            "group": "http-write",
            "objects": [
                "/special/http"
            ],
            "permission": "write"
        },
        {
            "group": "indexer-view",
            "objects": [
                "/special/indexer"
            ],
            "permission": "read"
        },
        {
            "group": "indexer-write",
            "objects": [
                "/special/indexer"
            ],
            "permission": "write"
        },
    ],
    "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"
    }
}
```

Element	Type	Description	
body		Top level element (JSON object)	Contains the properties of the user.
	group	string	The name of the usergroup.
	objects	list	The list of privileges that the group has access to.
	permission	read | write	The type of the permission. The group needs write access to configure an object, or to perform certain actions.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
       }
}
```

Element	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 endpoint.
`meta`	Top-level item (JSON object)	Contains the path to the endpoint.
`href`	string (relative path)	The path of the resource that returned the response.

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 more information on the `meta` object, see Message format on page 10.

```
{
    "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",
```

```
        "parent": "/api/acl/audit_data",
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
    }
}
```

Elements	Type	Description
body	Top-level element (JSON object)	Contains the JSON object of the rule.
name	string	The human-readable name of the audit data access rule that you specified when you created the rule.
query	string	The query that members of the usergroup(s) are allowed to perform.
groups	list	The usergroup(s) whose access to audit data you want to restrict.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
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.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 Access Control Lists (ACLs):

```
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.

```
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 41.

Code	Description	Notes
400	SessionIdMissing	No session ID is given in the "id" query parameter.
404	SessionCouldNotBeFound	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 "id" query parameter.
500	SessionTerminationFailed	The session could not be terminated due to internal errors.
500	RemoteNodeInfoMissing	The cluster node where the session is being recorded is missing from your primary node's configuration. For assistance, contact our Support Team.
503	SessionTerminationServiceUnavailable	Session termination service is

Code	Description	Notes
		unavailable on the specific host for closing sessions. To make sure session termination service is running, login to the host CLI and issue the 'systemctl restart sessionterminationservice.service' command.
504	MinionUnavailable	The minion node that is recording the session is unavailable. To get more information about the missing node, navigate to /api/cluster-/status.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
	}
}
```

Element	Description
`groups`	Endpoint that contains local usergroups.
`users`	Endpoint that contains local usernames.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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

Cookie name	Description	Required	Values
			that SPS records (and which also have a session ID, but in a different format).

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 more information on 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"
       }
 }
```

Element	Type	Description
`body`	Top level element (JSON object)	Contains the properties of the usergroup.
`members`	list	Lists the names of the users belonging to

Element	Type	Description
		the group.
`name`	string	The name of the group.
`key`	string	Top level element, contains the ID of the 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.
409	NoTransaction	No open Transaction is available. You must open a transaction first (for details, see Open a transaction on page 32).

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 49.

1. **Open a transaction.**

   For details, see Open a transaction on page 32.

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.

```
{
        "name":  "new-userggroup",
        "members":  ["46785097158061f46c63d0",  "1362061674580df4e00620d"]
 }
```

For example:

```
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 35.

Delete usergroup

To delete a usergroup, you have to:

1. Open a transaction (for details, see Open a transaction on page 32).
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 47. If the DELETE request is successful, the response includes only the meta object, for example:

```
{
        "meta":  {
                "href":  "/api/configuration/aaa/local_
database/groups/b080b1ba546232548bb1a9",
                "parent":  "/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 35).

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 32).
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 47.
4. Commit your changes to actually delete the object from SPS (for details, see Commit a transaction on page 35).

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 221.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
       "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"
        }
```


Element		Type	Description
body		Top level element (JSON object)	Contains the properties of the user.
	name	string	The username of the user account.
	password	reference	A reference to a password object. To create or update passwords, see Passwords stored on SPS on page 275.
	password_created	integer	The date when the password of the account was changed in UNIX timestamp format (for example, 1476796261).
key		string	Top level element, contains the ID of the user.

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 41.

Code	Description	Notes
400	SemanticError	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.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.
409	NoTransaction	No open Transaction is available. You must open a transaction first (for details, see Open a transaction on page 32).

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

Cookie name	Description	Required	Values
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 more information 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

Cookie name	Description	Required	Values
			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/ldap_servers` endpoint include:

Operation	HTTP method	URL	Notes
Creating a new LDAP server	POST	`/api/configuration/aaa/ldap_servers`	
Retrieving a LDAP server	GET		

Sample request

The following command creates a new LDAP server.

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

The following is a sample request.

```
{
     "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": "uniqueMember"
        }
      ]
    },
    "servers": [
      {
        "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"
    }
  }
```

Response

The following is a sample response received when you retrieve LDAP server information. For more information on the `meta` object, see Message format on page 10.

```
{
    "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"
            },
```

```
            "selection": "posix",
            "user_dn_in_groups": [
              {
                "attribute": "member",
                "object_class": "groupOfNames"
              },
              {
                "attribute": "uniqueMember",
                "object_class": "groupOfUniqueNames"
              }
            ],
            "username_attribute": "uid"
          },
          "servers": [
            {
              "host": {
                "selection": "ip",
                "value": "10.110.0.1"
              },
              "port": 389
            }
          ],
          "user_base_dn": "ou=People,dc=example"
        },
        "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
        "meta": {
          "href": "/api/configuration/aaa/ldap_servers/XXXXXXXX-XXXX-XXXX-XXXX-
XXXXXXXXXXXX"
        }
      }
    ]
  }
```


Elements of the response message body include:

Element	Type	Description	Notes
`bind_dn`	string	The Distinguished Name that SPS should use to bind to the LDAP directory. Must be used if the value of the `selection` element is set to `ldap`.	NOTE: SPS accepts both pre Windows 2000-style and Windows 2003-style account names, or User Principal Names (UPNs). For example, `administrator@example.com` is also accepted.
`bind_password`	null | string	References the password	NOTE: SPS accepts

Element	Type	Description	Notes
		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).	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: `!"#$%&'()*+,-./:;<>=?@[]\^-`{}_|`
`encryption`	union	Configuration settings for encrypting the communication between SPS and the LDAP server.	The displayed value type depends on the `encryption.selection` parameter.

`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 must appear in the
			• Example if the value is `disabled`
			`{ "selection": "disabled" }`
			• Example if the value is `ssl` or `starttls`:
			`{ "selection": "ssl | starttls", "trust_store": null|<trust-store-ref>, "client_authentication": null|<x509-ref> }`

Element	Type	Description	Notes
		Common Name of the certificate. NOTE: TLS-encrypted connection to Microsoft Active Directory is supported only on Windows 2003 Server and newer platforms. Windows 2000 Server is not supported. • `starttls` Opportunistic TLS.	
`group_base_dn`	string	Name of the DN to be used as the base of queries regarding groups. 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.	Example: `"ou=users,ou=example"`
`name`	string	Top level element, the name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.	
`schema`	union	AD and POSIX specific LDAP configuration.	• Example with LDAP AD server, where the schema selection is `ad`: `{ "selection": "ad",`

Element	Type	Description	Notes

			`"membership_check": ...,`
			`"memberof_check": ...,`
			`"user_dn_in_groups": …`
			`}`
			
			• Example with LDAP POSIX server, where the schema selection is `posix`:
			
			`{`
			`"selection": "posix",`
			`"membership_check": ...,`
			`"memberof_check": ...,`
			`"user_dn_in_groups": ...,`
			`"username_attribute": …`
			`}`

`schema.memberof_check`	object	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`.	• Example with LDAP AD server, if `memberof_check` and `memberof_user_attribute` are enabled:
			
			`"schema": {`
			`"memberof_check": {`
			`"enabled": true,`
			`"memberof_user_attribute": "memberOf"`
			`}`

Element	Type	Description	Notes

			• Example with LDAP POSIX server, if `memberof_check` is enabled:
			`"schema": { "memberof_check": { "enabled": true, "memberof_group_objectclass": "groupOfNames", "memberof_user_attribute": "memberOf" }`
			• To disable `memberof_check` for both LDAP AD and LDAP POSIX servers, set the `enabled` parameter to `false`.
			`"schema": { "memberof_check": { "enabled": false }`

`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

Element	Type	Description	Notes
`check.memberof_user_attribute`		`memberof_check` is set it to `true`. The name of the user attribute (for example, `memberOf`) that contains the group DNs.	for both LDAP AD and LDAP POSIX schema.

`schema.membership_check`	object	• POSIX: POSIX primary and supplementary group membership checking.
		• AD: Active Directory specific non-primary group membership checking.
			• Example with LDAP AD server, if `membership_check` and `nested_groups` are enabled:
			`"membership_check": { "enabled": true, "nested_groups": true | false" }`
			• Example with LDAP POSIX server, if `membership_check` and `member_uid_attribute` are enabled:
			`"membership_check": { "enabled": true, "member_uid_attribute": null | "memberUid" }`
			• To disable `membership_check` for both LDAP AD and LDAP POSIX servers, set the `enabled` parameter to `false`.

Element	Type	Description	Notes
			`"membership_check": { "enabled": false }`
`schema.membership_check.enabled`	boolean	POSIX: Enables POSIX primary and supplementary group membership checking. AD: Enables Active Directory specific non-primary group membership checking.	
`schema.membership_check.member_uid_attribute`	string	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`.	Default value: `memberUid`
`schema.selection`	string	Configures which LDAP schema to use: AD or POSIX. Possible values are: • `ad`: Microsoft Active Directory server. For details and examples, see Configuring LDAP servers. • `posix`: The server uses the POSIX LDAP scheme. Must be used with the	Default value: `ad`, `posix`

Element	Type	Description	Notes
		`member_uid_attribute` and `username_attribute` elements. For details and examples, see Configuring LDAP servers.	

`schema.user_dn_in_groups`	array	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.	Example:
			`"user_dn_in_groups": [ { "attribute": "member", "object_class": "groupOfNames" }, { "attribute": "uniqueMember", "object_class": "groupOfUniqueNames" } ]`
			The array can return empty:
			`"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.

`schema.user_dn_in_groups.attribute`	string	Name of the group attribute which contains the user DN.	Default value: `member`

Element	Type	Description	Notes
`schema.user_dn_in_groups.object_class`	string	Consider groups of this objectClass.	Possible values: `groupOfNames`, `groupOfUniqueNames`
`schema.username_attribute`	string	The login attribute that uniquely identifies a single user record.	Default value: `uid`

`servers`	array	Contains the addresses and ports of the LDAP servers.	The displayed value type depends on the `servers.host.selection` parameter.
			Possible values are: `ip`, `fqdn`
			• Example if the host selection is `ip`:
			`"servers": [ { "host":{ "selection": "ip", "value": "1.2.3.4" }, "port": 123 } ],`
			• Example, if the host selection is `fqdn`:
			`"servers": [ { "host": {`

Element	Type	Description	Notes
			"selection": "fqdn", "value": "my.example" }, "port": 123 }],
`servers.host`	object	Contains the address of the LDAP server.	
`servers.host.selection`	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.
`servers.host.value`	string	The address of the LDAP server.	
`servers.port`	int	The port of the LDAP server.	
`user_base_dn`	string	Name of the DN to be used as the base of queries regarding users. Must be used if the value of the `selection` element is set to `ldap`. NOTE: You must fill in this field. It is OK to use the same value for `user_base_dn` and `group_base_dn`.	For example: `"ou=groups,o-ou=example"`

Element	Type	Description	Notes
		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 32.

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 35.

HTTP response codes

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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 more information 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 `/aaa/login_methods` endpoint include:

Operation	HTTP method	URL	Notes
Configuring the Local login method	POST	`/api/configuration/aaa/login_methods`	NOTE: Changing your password is possible only with Local login.
Configuring the X509 login method			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.
Configuring the password-based LDAP login method (AD or POSIX)			Before you can configure the LDAP login method, you must create a new LDAP server at the `/api/configuration/aaa/ldap_servers` endpoint.
Configuring the RADIUS login method			

Operation	HTTP method	URL	Notes
Reordering login methods	PUT	`/api/configuration/aaa/login_methods/@order`	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.

Sample request

The following command creates a new login method.

```
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**.

Element	Type	Description	Notes
`type`	enum	The login method type used for authentication.	Possible values are: • `local` Use a local user database for authentication. • `x509` Use a X509 certificate for authentication. • `ldap` Use an LDAP server (AD or POSIX) for authentication. • `radius` Use a RADIUS server for authentication.
`name`	string	A unique identifier of the login method.	
`title`	string	The title that appears above the login method button on the SPS web	In the case of X509 login method, there is

Element	Type	Description	Notes
		interface. This can be customized.	only one button.
`active`	boolean	Indicates whether the login method is enabled or not.	

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:

Element	Type	Description	Notes
`cracklib_enabled`	boolean	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.	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**.
`expiration_days`	integer	Password setting. Configures the number of days the user passwords	Set to `0` if a RADIUS server or X.509 certificate is used for authentication.

Element	Type	Description	Notes
		are considered valid. Expired passwords must be changed upon login.	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`.
`remember_previous_passwords`	integer	Password setting. Configures the number of previous passwords to retain to prevent password reuse.	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`.
`minimum_password_strength`	string	Password setting. Configures the required password strength for new passwords.	Set to `disabled` if a RADIUS server or X.509 certificate is used for authentication. Possible values are: • `disabled` Any password is accepted. • `good` Weak passwords are not accepted. • `strong` Only strong passwords are accepted. Must be used if the value of the `selection` element is set to `local`.
`minimum_password_length`	number	The minimum number of characters the password must consist of.	

Example: Sample request for configuring LDAP login

```
{
        "type": "ldap",
        "name": "ldap",
        "title": "LDAP login",
        "active": true,
        "ldap_server": "5ed1b422-395a-4d5e-abf0-b43cd6c752f8"
    }
```

Elements of the request message body include:

Element	Type	Description	Notes
`ldap_server`	string	The identifier of the previously created LDAP server.	

Example: Sample request for configuring X509 login

```
{
        "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:

Element	Type	Description	Notes
`trust_store`	string	The identifier of the X509 certificate that was previously uploaded into	Trust stores serve as local certificate storages where users can store the

Element	Type	Description	Notes
		Trust stores.	certificate chains of trusted Certificate Authorities (CAs).
`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.	Possibe 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 32.

2. Create a new LDAP server at the `/api/configuration/aaa/ldap_servers` endpoint.

```
{
    "name": "ldap_server",
    "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": "ip",
                "value": "1.2.3.4"
            },
            "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"
    }
}
```

3. Upload a X509 certificate at the `/api/configuration/trust_stores` endpoint.

```
{
    "name": "X509_Trust_Store",
    "authorities": [
        "-----BEGIN CERTIFICATE-----
```

```
\nMIIDZzCCAk+gAwIBAgIUMlI5+EgTDAh2zqRDGYrzFRyozI8wDQYJKoZIhvcNAQEL\nB
QAwQzELMAkGA1UEBhMCSFUxETAPBgNVBAgMCEJ1ZGFwZXN0MSEwHwYDVQQKDBhJ\nbnRl
cm5ldCBXaWRnaXRzIFB0eSBMdGQwHhcNMTQwODEyMTIzNjQ4WhcNMzQwNjE4\nMTIzNjQ
4WjBDMQswCQYDVQQGEwJIVTERMA8GA1UECAwIQnVkYXBlc3QxITAfBgNV\nBAoMGEludG
VybmV0IFdpZGdpdHMgUHR5IEx0ZDCCASIwDQYJKoZIhvcNAQEBBQAD\nggEPADCCAQoCg
gEBALffJBDD6A/ZGBTgFbyLXHulU+hGnMW3DoPo2q4HY1/FfbkS\nrzmK+Fiz+3EwJCWi
+EwK9mqve/nh6YRRw/VaAVQ7CkA7f7to+I7gP647Bq1wk0lh\nBVEJNlN0jfYYSumGxzP
otw/fon1MkXuMbLc0Pr/vFX3NQC7/STAV5dZFcdboXDA7\nZZ3rzBIr93ThObsGj01MRO
6wrS3rfE7Px9D7C2u9YSkP3OQ1Sfm/jqyLNaT6xt4i\nhrLnfYEc8mClnrlvILi+q/D6m
IUSjb4IGvergAyl4jgPjO02UcvBzOIA9tDlBJBi\nQxZx+T620ubmEwOl9Q0G8RAWKz7s
zrBcXEjXhYUCAwEAAaNTMFEwHQYDVR0OBBYE\nFCDfEeq5Hsm8jMrG110iNpt5cikTMB8
GA1UdIwQYMBaAFCDfEeq5Hsm8jMrG110i\nNpt5cikTMA8GA1UdEwEB/wQFMAMBAf8wDQ
YJKoZIhvcNAQELBQADggEBAK3iizM4\nCx69YD+4CWOUswULrCJA38C+nDYONLbNkact8
JKXqCn/MaZTII+dZoV9RjjX4AzA\nPTQkZT+RoVeCZyt+qWHMdjq6koabXwQmXNozUtax
EZTrnoUDEWtNIbjV/gNtRcSG\nsU7i9L2YnwDzTw0cR/pu1Hykq8fwqNqjQGYnmXtJspM
kKAtVe1CrtnPLiC6JBr0g\n5GZF58sHx5+gO0RkqdzJgRAGnImdfAahqfHmKRFmxoxWLy
ylRyqDgQ+KqcaDvZI+\ni36M+NQHVrDX4jo4CFoXhFlSOepvtDOpmzoWhugwDNMPuU1IE
Y7//CJBXQnjp+uf\nLO6PsNmMKDGi9Dk=\n-----END CERTIFICATE-----\n"
     ],
     "crl_urls": [
         "http://crl.it/sec"
     ],
     "revocation_check": "full"
}
```


4. Create a new LDAP login method at the `/api/configuration/aaa/login_methods` endpoint.

```
{
        "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 35.

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 32.

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 35.

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 32.
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.

```
{
     "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": {
```

```
        "type": "ldap",
        "ldap_server": "<'key' from the response of the penultimate
creation>"
    }
}
```

5. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

Response

For more information on 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 41.

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

Cookie name	Description	Required	Values
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 more information 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 queries the troubleshooting settings.

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

Response

The following is a sample response received.

For more information on the `meta` object, see Message format on page 10.

```
{
    "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"
    }
 }
```

Element	Type	Description
`key`	string	Top level element, contains the ID of the endpoint.
`body`	Top level element (string)	Contains the troubleshooting settings.
`core_files`	Top level item	Contains the settings for core dump file retention.
`retention_days`	int	Retention time for core files, in days.
`debug_`	Top level	Settings for debug logging.

Element	Type	Description
`logging`	item	
`enabled`	boolean	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.

Modify troubleshooting settings

To modify troubleshooting settings, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
  }
}
```


Element		Type	Description
key		string	The ID of the endpoint.
body		Top level element (string)	Contains the internal certificates of SPS.
ca		Top level item	Contains the certificate of SPS's internal Certificate Authority.
	selection	string	Must be set to `identity`.
	x509_identity	string	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

Element	Type	Description
		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 300.
`server`	string	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 300.
`tsa`	string	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 300.

Modify a certificate

To modify a certificate, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 49. 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

Header name	Description	Required	Values
Content-Type	Specifies the type of the data sent. SPS uses the JSON format	Required	`application/json`
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.

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$If20/EFyQ4dW3dg/$xrECLfXgZlC2Xr1s257E2aZen42fM7R.sOGG9pkPy1x5ORTx6j03oPWexVlB3f5wnaZOQCBF.NjlDgyg2WEe./"}'
```

Table 4: Password parameters

Element	Type	Description
`hash`	string	Must contain the SHA-256 hash of the password to be created, for example, `"hash": "ddec437eeb1da25a146a24c432d1165bc646daa7fecc6aa14c636265c83caa14"`.
`nthash`	string	Optional. Contains the NT-HASH of the password to be created, for example, `"nthash": "2c01a73ad9e597f6eab0d072ed74616c"`
`plain`	string	Contains the password in plain-text format, for example, `"plain": "mypassword"`.

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 more information on 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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.
405	MethodNotAllowed	The method <method> is not allowed for this node.

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 32.

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 35.

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 32.

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:

   ```
   curl -X PUT -H "Content-Type: application/json" --cookie cookies
   https://<IP-address-of-SPS>/api/configuration/management/root_password --
   data '"<key-of-the-new-password>"'
   ```

   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 35.

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 32.

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. **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:

   ```
   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:

   ```
   {
         "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 35.

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 49. 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

Header name	Description	Required	Values
Content-Type	Specifies the type of the data sent. SPS uses the JSON format	Required	`application/json`
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.

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-----
\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2SchDibk/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .........
Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END RSA PRIVATE KEY-----\n"
```

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;>
```

Element		Type	Description
`public-key-fingerprint`		string	The fingerprint of the public key that matches the private key.
	`digest`	string	The fingerprint of the key, for example `2048 SHA256:JPKdfkT6wU9c11bbqX53hovDo7KbIB8OREfumUWDh9f no comment (RSA)`
	`hash_algorithm`	string	The hash algorithm used to create the fingerprint, for example, `sha256`.
`type`	string	The type of the private key. Must be `rsa`	

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 more information on the `meta` object, see Message format on page 10.

```
{
       "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:

```
{
       "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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	SyntacticError	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.
401	Unauthenticated	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.
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

Code	Description	Notes
		could not be retrieved.
404	NotFound	The requested object does not exist.
405	MethodNotAllowed	The method <method> is not allowed for this node.

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 272)
- SMTP client authentication (`/api/configuration/management/email`, see Mail settings on page 157)
- Syslog client authentication (`/api/configuration/management/syslog`, see Syslog server settings on page 148)
- LDAP client authentication (for policies: `/api/configuration/policies/ldap_servers`, see LDAP servers on page 405, and for authentication: `/api/configuration/aaa/ldap_servers`, see Configuring LDAP servers on page 244)

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 200.

URL

```
https://<IP-address-of-SPS>/api/pki/certificate
```

Table 6: Headers

Header name	Description	Required	Values
Content-Type	Specifies the type of the data sent. SPS uses the JSON format	Required	`application/json`
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.

Operations

Operations with the /api/pki/certificate endpoint include:

Operation	HTTP method	URL	Notes
Generating a new CSR	POST	`/api/pki/certificate/requests`	
Adding an X.509 certificate chain to a CSR to create a X.509 identifier	PUT	`/api/pki/certificate/requests/<ID-of-the-CSR>`	

Operation	HTTP method	URL	Notes
Setting or replacing a certificate chain for a CSR without knowing the CSR identifier	POST	`/api/pki/certificate`	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.
Querying existing CSRs	GET	`/api/pki/certificate/requests`	
Querying a single CSR	GET	`/api/pki/certificate/requests/<ID-of-the-CSR>`	
Deleting a CSR	DELETE	`/api/pki/certificate/requests/<ID-of-the-CSR>`	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.

Example: Generating a new CSR

The following command creates a new CSR.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/pki/certificate/requests
```

```
{
        "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:

Element	Type	Description	Notes
`subject`			
`subject.name`	string	The subject name must be an object identifier (OID), or a name that can be translated to an OID.	Example values are: • `countryName` • `stateOrProvinceName` • `streetAddress` • `organizationName` • `commonName` • `emailAddress`
`subject.value`	string		
`extensions`	object	The list of extensions.	If you do not want to specify

Element	Type	Description	Notes
			any extensions in the request, use an empty list.
`extensions.name`	enum	The name of the extension.	Possible values are: • `basicConstraints` • `keyUsage` • `extendedKeyUsage` • `subjectAltName` • `subjectInfoAccess`
`extensions.value`	string	The value of the extension.	
`extensions.critical`	boolean	Indicates whether the extension should be marked as critical in the request.	

Response

The following is a sample response received when a new CSR is created.

For more information on the `meta` object, see Message format on page 10.

```
{
        "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 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 32.

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


```
{
     "certificate_chain": "-----BEGIN CERTIFICATE-----
\nMIID+zCCAeOgAwIBAgIBTDANBgkqhkiG9w0BAQsFADAXMRUwEwYDVQQDDAxFe
GFt\ncGxlIENBIDIwHhcNMjAwODMxMTIyMDU2WhcNMzAxMDE4MTIyMDU2WjCBsj
ELMAkG\nA1UEBhMCVVMxCzAJBgNVBAgMAkNBMRcwFQYDVQQHDA5FeGFtcGxlIEN
pdHkgMjEf\nMB0GA1UECgwWRXhhbXBsZSBPcmdhbml6YXRpb24gMjEVMBMGA1UE
CwwMRXhhbXBs\nZSBVbml0MRowGAYDVQQDDBF0aGlyZC5leGFtcGxlLmNvbTEOM
AwGA1UEEQwFMTIz\nNDUxGTAXBgNVBAkMEEV4YW1wbGUgU3RyZWV0IDIwWTATBg
cqhkjOPQIBBggqhkjO\nPQMBBwNCAASYBjc7KadwuOFlI4YAPxtBUxc1fUj9DIg
uud5Bl+06jTdPnTqVo00w\n23LO0ILzuJ+JXMc8gvv+BtRhzrNM1IYao4GAMH4w
CQYDVR0TBAIwADALBgNVHQ8E\nBAMCA4gwEwYDVR0lBAwwCgYIKwYBBQUHAwIwE
QYJYIZIAYb4QgEBBAQDAgeAMB0G\nA1UdDgQWBBSt0NXz4/3yMPCmfoz8hurej0
mByzAdBgNVHREEFjAUghJmb3VydGgu\nZXhhbXBsZS5jb20wDQYJKoZIhvcNAQE
LBQADggIBAIHEw56a3Wmhyx9qOlVEDYsz\nQYYfmyxapPBxSrBCfhPq7hDSyUf5
ZizeQ14C48zgd0pWEjONI3jyJp0pQzu++Qsy\nFyErYqhXsbGObhBTyAjGfvPiB
```

```
uNjIbrfzMAdavYUv4dtFCi49gByjHshJbGYDqPP\nbR1Zzky8/B20IvarmlEigp
8bnJXWqk0juQOQ6lM06bjycrFRXyNo3EdF8JS4TGy4\n/H9ZCPKvQXB5fGVjGyx
tfbr3Hij3/B/Lv0mrKb/qCxEvl8ACtrT1lVRDAbgVIzn4\nYMporoTJhqkU7Oau
Bqu9eDDHUzc1VfXMUSV3UD+IuCEpoB1f7a1YRp/kSLp+XpO+\nZn+9SA4IFI7cb
PWDM45po51GkmpCG9xQhjq7UKnvCj4fov34fp/GWjPrqZZ5TykQ\npYNJfUd/dn
8N4zNM/lkw2HLbg2bgO6ARaTl0s9kR0gv3RKFrNZb9nXYvkedNeXFA\n4siyfG9
kNF9CoSYZB1pz5aZNBZn9re5+PKoIiccBUKS209jD6ZJZTfu3oq3FibaU\nYVJK
ZraUajXFEDr0qS5/XtJUMcmQCXITLlpsOdnyGhN23I7w/vImqN06cTeoKFli\nY
t+zCq8nNfcJp6n3YsfUT1ZRW2ros+8ARY0Wzdd8SCv0sx9xu+CFotWR4aOqCd93
\nnoq6yMj8UwretI+1kHim\n-----END CERTIFICATE-----\n-----BEGIN
CERTIFICATE-----
\nMIIFOzCCAyOgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADAXMRUwEwYDVQQDDAxFe
GFt\ncGxlIENBIDIwHhcNMjAwNzEzMTczMTE4WhcNMzAwNzExMTczMTE4WjAXMR
UwEwYD\nVQQDDAxFeGFtcGxlIENBIDIwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwA
wggIKAoIC\nAQCaQ937PQAp9CcNXk5b6VhqIBXRax1TYcwGR2elf0SRY2KP41mS
0jYoZbbJRcJ+\nwPtFKO2AD4RNU0OnSkfTX8aEANbZTBWdMQy9Nod+lOrHtmOoS
We4dbkDLYZPD0qn\n8VYMrr/aHwImli7MHsITNzdioVZ7p3andLWrEh8a04yDAq
kdQwi9M8X6GPzBmLKk\nVtYR/wMaZg9W24eT9mMN06sCFxtUeIT2v+jrCSV7FLW
AgEFJhoyZpT2uigbFhnIp\nB3gnJfUv6MRh6BSeLNF8SOGbqoyJFYFtWlKv/HL9
rGtCOjfdxX8K3zhmNKpMOAjw\njg2XUiVWxySZ1OTPi8Fu7KKj8g47hiGkERWHp
BmswjAq+fBoaircIHmqqUEHPLaD\ny6IIPuCDljAvtC/M6TlAMX7aGOG0R49LEO
0UtVvWJyHAkLSntACx7sVLXXWJr0ku\nrrVdm4UUx5aLLbS+s0Xum5sNKZLqBYu
5B2KPxBfhqxKGL0AJoIHAM5cgG7LPTrdX\nRDin0/82RErqvGK+DrhgLP+/kTK/
UvWIm8SGN5HfP4Cod/di/11GBjhMYBcHePW7\nCbGHap4m4vNHSGoPYdKbD/daL
Me1pjTx+lw1HfVIXSysWkC7PTG+LZNn1zLzjs2J\nVE0OcG+gjDouddb43j/T4j
1pw5R24iaQ+oq9gpj0MY5qewIDAQABo4GRMIGOMAwG\nA1UdEwQFMAMBAf8wCwY
DVR0PBAQDAgEGMBEGCWCGSAGG+EIBAQQEAwIBBjAdBgNV\nHQ4EFgQUs5wDSfrQ
a+fJkM6Ek07dbkG3l30wPwYDVR0jBDgwNoAUs5wDSfrQa+fJ\nkM6Ek07dbkG3l
32hG6QZMBcxFTATBgNVBAMMDEV4YW1wbGUgQ0EgMoIBAjANBgkq\nhkiG9w0BAQ
0FAAOCAgEAkf2J2M4eHDnRTGQsZTcs91SdV/2fH0W+NLTRdGO9V0NL\nKMRXrlJ
8Gy3A/4U/Hx5Lo4dQQckePTdXzFQARX5m/7VIf2+YlUDlNre/fMt5aeWG\n67vl
UnUYLqgHqV2G5QmqZ26DUwcMTXK3oy8rqel1xtQAk1Cpfdvfi7gn2cEFRD7U\ne
xg3AemJMBV26spcGnaf/smfcHeVWo9lwqzyWzwvGYTEeb1MajCgINnh39DZBHlO
\nPE77yRyuANtDzWMo0ZNnlU+FpHTexhooQnYRKtEagHDTTF2ZuSkcJnczlTFAP
4cM\ntvEro4ePMdxLuMf4lVrBt+OudsVnoij+U9OHJst7Czk5MXyZzGHgkmXOg+
JUW49D\nOfjDq0HiW64QeWYu+lsEFS+2sHES82R/JBmXWbqHy68JhDUubzAi7nR
XEHkKP9N3\nXF6jD3Fyn47kN4uYeHas9eCvMFmxfv6TGzCPgfS9PH0ZaZLM/Cp6
u9l4DhyvFV9n\nuazsa9pkwlQftE8L1Xo/hGmKBdLGJDRDQo1r4eYYZlzS94tj5
p0thztTmUWZFOl2\ndOTyux8EvhpsefDYwtlJ1wxqvFdDm5WGCxeZd6YAsV/Kuk
VQLakX1LS/i7FEFsuL\nZ89WjWdHC2A3liKOa/y0/vy9gOY7GScvRnyUSthQ2Y0
DYO89KDzY6Q6ILbpsVr8=\n-----END CERTIFICATE-----",
     "trust_store": "2222"
   }
```

Elements of the request message body include:

Element	Type	Description	Notes
`certificate_chain`	string or list	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.	
`trust_store`	string	The identifier of the trust store that is used to validate the certificate chain, or `null`, if you want to disable validation.	

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 272.

7. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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


```
{
     "certificate_chain": "-----BEGIN CERTIFICATE-----
\nMIID+zCCAeOgAwIBAgIBTDANBgkqhkiG9w0BAQsFADAXMRUwEwYDVQQDDAxFe
GFt\ncGxlIENBIDIwHhcNMjAwODMxMTIyMDU2WhcNMzAxMDE4MTIyMDU2WjCBsj
ELMAkG\nA1UEBhMCVVMxCzAJBgNVBAgMAkNBMRcwFQYDVQQHDA5FeGFtcGxlIEN
pdHkgMjEf\nMB0GA1UECgwWRXhhbXBsZSBPcmdhbml6YXRpb24gMjEVMBMGA1UE
CwwMRXhhbXBs\nZSBVbml0MRowGAYDVQQDDBF0aGlyZC5leGFtcGxlLmNvbTEOM
AwGA1UEEQwFMTIz\nNDUxGTAXBgNVBAkMEEV4YW1wbGUgU3RyZWV0IDIwWTATBg
cqhkjOPQIBBggqhkjO\nPQMBBwNCAASYBjc7KadwuOFlI4YAPxtBUxc1fUj9DIg
uud5Bl+06jTdPnTqVo00w\n23LO0ILzuJ+JXMc8gvv+BtRhzrNM1IYao4GAMH4w
CQYDVR0TBAIwADALBgNVHQ8E\nBAMCA4gwEwYDVR0lBAwwCgYIKwYBBQUHAwIwE
QYJYIZIAYb4QgEBBAQDAgeAMB0G\nA1UdDgQWBBSt0NXz4/3yMPCmfoz8hurej0
mByzAdBgNVHREEFjAUghJmb3VydGgu\nZXhhbXBsZS5jb20wDQYJKoZIhvcNAQE
LBQADggIBAIHEw56a3Wmhyx9qOlVEDYsz\nQYYfmyxapPBxSrBCfhPq7hDSyUf5
ZizeQ14C48zgd0pWEjONI3jyJp0pQzu++Qsy\nFyErYqhXsbGObhBTyAjGfvPiB
uNjIbrfzMAdavYUv4dtFCi49gByjHshJbGYDqPP\nbR1Zzky8/B20IvarmlEigp
8bnJXWqk0juQOQ6lM06bjycrFRXyNo3EdF8JS4TGy4\n/H9ZCPKvQXB5fGVjGyx
tfbr3Hij3/B/Lv0mrKb/qCxEvl8ACtrT1lVRDAbgVIzn4\nYMporoTJhqkU7Oau
Bqu9eDDHUzc1VfXMUSV3UD+IuCEpoB1f7a1YRp/kSLp+XpO+\nZn+9SA4IFI7cb
PWDM45po51GkmpCG9xQhjq7UKnvCj4fov34fp/GWjPrqZZ5TykQ\npYNJfUd/dn
8N4zNM/lkw2HLbg2bgO6ARaTl0s9kR0gv3RKFrNZb9nXYvkedNeXFA\n4siyfG9
kNF9CoSYZB1pz5aZNBZn9re5+PKoIiccBUKS209jD6ZJZTfu3oq3FibaU\nYVJK
ZraUajXFEDr0qS5/XtJUMcmQCXITLlpsOdnyGhN23I7w/vImqN06cTeoKFli\nY
t+zCq8nNfcJp6n3YsfUT1ZRW2ros+8ARY0Wzdd8SCv0sx9xu+CFotWR4aOqCd93
\nnoq6yMj8UwretI+1kHim\n-----END CERTIFICATE-----\n-----BEGIN
CERTIFICATE-----
\nMIIFOzCCAyOgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADAXMRUwEwYDVQQDDAxFe
GFt\ncGxlIENBIDIwHhcNMjAwNzEzMTczMTE4WhcNMzAwNzExMTczMTE4WjAXMR
UwEwYD\nVQQDDAxFeGFtcGxlIENBIDIwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwA
wggIKAoIC\nAQCaQ937PQAp9CcNXk5b6VhqIBXRax1TYcwGR2elf0SRY2KP41mS
0jYoZbbJRcJ+\nwPtFKO2AD4RNU0OnSkfTX8aEANbZTBWdMQy9Nod+lOrHtmOoS
```

```
We4dbkDLYZPD0qn\n8VYMrr/aHwImli7MHsITNzdioVZ7p3andLWrEh8a04yDAq
kdQwi9M8X6GPzBmLKk\nVtYR/wMaZg9W24eT9mMN06sCFxtUeIT2v+jrCSV7FLW
AgEFJhoyZpT2uigbFhnIp\nB3gnJfUv6MRh6BSeLNF8SOGbqoyJFYFtWlKv/HL9
rGtCOjfdxX8K3zhmNKpMOAjw\njg2XUiVWxySZ1OTPi8Fu7KKj8g47hiGkERWHp
BmswjAq+fBoaircIHmqqUEHPLaD\ny6IIPuCDljAvtC/M6TlAMX7aGOG0R49LEO
0UtVvWJyHAkLSntACx7sVLXXWJr0ku\nrrVdm4UUx5aLLbS+s0Xum5sNKZLqBYu
5B2KPxBfhqxKGL0AJoIHAM5cgG7LPTrdX\nRDin0/82RErqvGK+DrhgLP+/kTK/
UvWIm8SGN5HfP4Cod/di/11GBjhMYBcHePW7\nCbGHap4m4vNHSGoPYdKbD/daL
Me1pjTx+lw1HfVIXSysWkC7PTG+LZNn1zLzjs2J\nVE0OcG+gjDouddb43j/T4j
1pw5R24iaQ+oq9gpj0MY5qewIDAQABo4GRMIGOMAwG\nA1UdEwQFMAMBAf8wCwY
DVR0PBAQDAgEGMBEGCWCGSAGG+EIBAQQEAwIBBjAdBgNV\nHQ4EFgQUs5wDSfrQ
a+fJkM6Ek07dbkG3l30wPwYDVR0jBDgwNoAUs5wDSfrQa+fJ\nkM6Ek07dbkG3l
32hG6QZMBcxFTATBgNVBAMMDEV4YW1wbGUgQ0EgMoIBAjANBgkq\nhkiG9w0BAQ
0FAAOCAgEAkf2J2M4eHDnRTGQsZTcs91SdV/2fH0W+NLTRdGO9V0NL\nKMRXrlJ
8Gy3A/4U/Hx5Lo4dQQckePTdXzFQARX5m/7VIf2+YlUDlNre/fMt5aeWG\n67vl
UnUYLqgHqV2G5QmqZ26DUwcMTXK3oy8rqel1xtQAk1Cpfdvfi7gn2cEFRD7U\ne
xg3AemJMBV26spcGnaf/smfcHeVWo9lwqzyWzwvGYTEeb1MajCgINnh39DZBHlO
\nPE77yRyuANtDzWMo0ZNnlU+FpHTexhooQnYRKtEagHDTTF2ZuSkcJnczlTFAP
4cM\ntvEro4ePMdxLuMf4lVrBt+OudsVnoij+U9OHJst7Czk5MXyZzGHgkmXOg+
JUW49D\nOfjDq0HiW64QeWYu+lsEFS+2sHES82R/JBmXWbqHy68JhDUubzAi7nR
XEHkKP9N3\nXF6jD3Fyn47kN4uYeHas9eCvMFmxfv6TGzCPgfS9PH0ZaZLM/Cp6
u9l4DhyvFV9n\nuazsa9pkwlQftE8L1Xo/hGmKBdLGJDRDQo1r4eYYZlzS94tj5
p0thztTmUWZFOl2\ndOTyux8EvhpsefDYwtlJ1wxqvFdDm5WGCxeZd6YAsV/Kuk
VQLakX1LS/i7FEFsuL\nZ89WjWdHC2A3liKOa/y0/vy9gOY7GScvRnyUSthQ2Y0
DYO89KDzY6Q6ILbpsVr8=\n-----END CERTIFICATE-----",
     "trust_store": "2222"
   }
```


Elements of the request message body include:

Element	Type	Description	Notes
`certificate_chain`	string or list	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.	
`trust_store`	string	The identifier of the trust store that is used to validate the certificate chain, or null, if you want to disable validation.	

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 272.

7. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

Example: Querying existing CSRs

The following is a sample response received when existing CSRs are queried.

For more information on the `meta` object, see Message format on page 10.


```
{
     "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": {
             "digest":
"eb:46:b6:bf:dc:4e:c6:cb:81:9b:ee:fd:a1:8d:7d:72:86:3d:48:87:ba:94:e0:0c:7
9:8e:73:77:fd:5b:97:3b",
             "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-----
\nMIICPzCCAeQCAQAwgegxCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG\nA1UECA
wCQ0ExFTATBgNVBAcMDEV4YW1wbGUgQ2l0eTEXMBUGA1UECQwORXhhbXBs\nZSBTdHJlZXQxHT
AbBgNVBAoMFEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQL\nDA5FeGFtcGxlIFVuaXQgMT
EXMBUGA1UECwwORXhhbXBsZSBVbml0IDIxGjAYBgNV\nBAMMEWZpcnN0LmV4YW1wbGUuY29tMR
8wHQYJKoZIhvcNAQkBFhBpbmZvQGV4YW1w\nbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQ
cDQgAEC/MAO3IIhG6zInpQxOJ9\nfFnOQlW11IoeMXHfhrhRC9OI9W77MjxRNX7gXS1WVcEQPx
gXtE9sHdc6Z8jgupIi\ncKCBmDCBlQYJKoZIhvcNAQkOMYGHMIGEMAkGA1UdEwQCMAAwCwYDVR
0PBAQDAgOI\nMBMGA1UdJQQMMAoGCCsGAQUFBwMCMBEGCWCGSAGG+EIBAQQEAwIGQDAdBgNVHQ
4E\nFgQUDr0ZP/F5s++a46mW+yIgs1CNWwYwIwYDVR0RBBwwGocEe3t7e4ISc2Vjb25k\nLmV4
YW1wbGUuY29tMAoGCCqGSM49BAMEA0kAMEYCIQCrRLitgHeDJ34VSksqwbZy\nUA0Klz6l2Ezr
RHGR0UOPbAIhAKA7u8xplNauUutkQPd4KHT5eyBMs0GUYJm1gr3r\ntZFr\n-----END
CERTIFICATE REQUEST-----\n",
            "public_key": "-----BEGIN PUBLIC KEY-----
\nMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEC/MAO3IIhG6zInpQxOJ9fFnOQlW1\n1IoeMX
HfhrhRC9OI9W77MjxRNX7gXS1WVcEQPxgXtE9sHdc6Z8jgupIicA==\n-----END PUBLIC
KEY-----\n"
            }
          },
          {
            "meta": {"href": "/api/pki/certificate/requests/XXXXXXXX-XXXX-
XXXX-XXXX-XXXXXXXXXXXY"},
            "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXY",
            "body": {
              "certificate_chain": [
"-----BEGIN CERTIFICATE-----
\nMIID+zCCAeOgAwIBAgIBSzANBgkqhkiG9w0BAQsFADAXMRUwEwYDVQQDDAxFeGFt\ncGxlIE
NBIDIwHhcNMjAwNzE0MTUxMzAwWhcNMzAwODMxMTUxMzAwWjCBsjELMAkG\nA1UEBhMCVVMxCz
AJBgNVBAgMAkNBMRcwFQYDVQQHDA5FeGFtcGxlIENpdHkgMjEf\nMB0GA1UECgwWRXhhbXBsZS
BPcmdhbml6YXRpb24gMjEVMBMGA1UECwwMRXhhbXBs\nZSBVbml0MRowGAYDVQQDDBF0aGlyZC
5leGFtcGxlLmNvbTEOMAwGA1UEEQwFMTIz\nNDUxGTAXBgNVBAkMEEV4YW1wbGUgU3RyZWV0ID
IwWTATBgcqhkjOPQIBBggqhkjO\nPQMBBwNCAASYBjc7KadwuOFlI4YAPxtBUxc1fUj9DIguud
5Bl+06jTdPnTqVo00w\n23LO0ILzuJ+JXMc8gvv+BtRhzrNM1IYao4GAMH4wCQYDVR0TBAIwAD
```

```
ALBgNVHQ8E\nBAMCA4gwEwYDVR0lBAwwCgYIKwYBBQUHAwIwEQYJYIZIAYb4QgEBBAQDAgeAMB
0G\nA1UdDgQWBBSt0NXz4/3yMPCmfoz8hurej0mByzAdBgNVHREEFjAUghJmb3VydGgu\nZXhh
bXBsZS5jb20wDQYJKoZIhvcNAQELBQADggIBAIvZuARB37ZLux/aCaRDwqOW\nW/+TctyeLRku
g31BGH75cLdEwO63VT4xmB9cbd1fipNl4KwBxUQePBin59f4y3C5\nL6PveBi1xzMl9RtTY3k0
lcjPH3qF7uZusmLi4WnpoVT3cTVxKZb1LgSJnwbfjY0x\nO7w8NcBNNuixgYXnbN74nfof2uC1
mh0c7vcVhWxPcH3KQdXfcOMhyaKGB2s5U+K1\ncWqVLTKhdEuSUi2ZrW5jXIAZdj53C1sVRnsD
kZ5lKwrPsrxPeCH7T4PG9f67cv3U\nqbuIiu6lMGK4tN8dEvbAgMOEhx8dWqynW4zj0bSFJMSd
shlS/oqMXpkph2/vQGeE\nDBmcZqaH4B6zuOj3cWC6IKfyQbxt+70kEG1YywwvtXs2vEZLKtQr
qaChusLaROx4\naE3cVeOa2sWNNjKTE9twyMobPUdvCQU59sAV9W0kEMYxa9sJdEsI+/+LowRk
E3sD\nQ1B5PE++mJYmPkBcNH2Mv9sutYMQy1/8ukNm+BTw+xpIDdZ86fuqEU7Rq3687A5Y\nZK
R5Rvn0kDEg9sydeN3FGvIAKssx9DXHJK7VXqZEIb/Xf4xekh37MgGyw94uPI08\nJnmaoo0My0
Izk2L3rmJP0MYoeoWT1KY7KMgNeaS3peWwkXa9FrHUNlKh07vM9v0S\nHH7vuBXml+G2Ujd+aV
Fg\n-----END CERTIFICATE-----\n",
                              "-----BEGIN CERTIFICATE-----
\nMIIFOzCCAyOgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADAXMRUwEwYDVQQDDAxFeGFt\ncGxlIE
NBIDIwHhcNMjAwNzEzMTczMTE4WhcNMzAwNzExMTczMTE4WjAXMRUwEwYD\nVQQDDAxFeGFtcG
xlIENBIDIwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoIC\nAQCaQ937PQAp9CcNXk5b6V
hqIBXRax1TYcwGR2elf0SRY2KP41mS0jYoZbbJRcJ+\nwPtFKO2AD4RNU0OnSkfTX8aEANbZTB
WdMQy9Nod+lOrHtmOoSWe4dbkDLYZPD0qn\n8VYMrr/aHwImli7MHsITNzdioVZ7p3andLWrEh
8a04yDAqkdQwi9M8X6GPzBmLKk\nVtYR/wMaZg9W24eT9mMN06sCFxtUeIT2v+jrCSV7FLWAgE
FJhoyZpT2uigbFhnIp\nB3gnJfUv6MRh6BSeLNF8SOGbqoyJFYFtWlKv/HL9rGtCOjfdxX8K3z
hmNKpMOAjw\njg2XUiVWxySZ1OTPi8Fu7KKj8g47hiGkERWHpBmswjAq+fBoaircIHmqqUEHPL
aD\ny6IIPuCDljAvtC/M6TlAMX7aGOG0R49LEO0UtVvWJyHAkLSntACx7sVLXXWJr0ku\nrrVd
m4UUx5aLLbS+s0Xum5sNKZLqBYu5B2KPxBfhqxKGL0AJoIHAM5cgG7LPTrdX\nRDin0/82RErq
vGK+DrhgLP+/kTK/UvWIm8SGN5HfP4Cod/di/11GBjhMYBcHePW7\nCbGHap4m4vNHSGoPYdKb
D/daLMe1pjTx+lw1HfVIXSysWkC7PTG+LZNn1zLzjs2J\nVE0OcG+gjDouddb43j/T4j1pw5R2
4iaQ+oq9gpj0MY5qewIDAQABo4GRMIGOMAwG\nA1UdEwQFMAMBAf8wCwYDVR0PBAQDAgEGMBEG
CWCGSAGG+EIBAQQEAwIBBjAdBgNV\nHQ4EFgQUs5wDSfrQa+fJkM6Ek07dbkG3l30wPwYDVR0j
BDgwNoAUs5wDSfrQa+fJ\nkM6Ek07dbkG3l32hG6QZMBcxFTATBgNVBAMMDEV4YW1wbGUgQ0Eg
MoIBAjANBgkq\nhkiG9w0BAQ0FAAOCAgEAkf2J2M4eHDnRTGQsZTcs91SdV/2fH0W+NLTRdGO9
V0NL\nKMRXrlJ8Gy3A/4U/Hx5Lo4dQQckePTdXzFQARX5m/7VIf2+YlUDlNre/fMt5aeWG\n67
vlUnUYLqgHqV2G5QmqZ26DUwcMTXK3oy8rqel1xtQAk1Cpfdvfi7gn2cEFRD7U\nexg3AemJMB
V26spcGnaf/smfcHeVWo9lwqzyWzwvGYTEeb1MajCgINnh39DZBHlO\nPE77yRyuANtDzWMo0Z
NnlU+FpHTexhooQnYRKtEagHDTTF2ZuSkcJnczlTFAP4cM\ntvEro4ePMdxLuMf4lVrBt+Ouds
Vnoij+U9OHJst7Czk5MXyZzGHgkmXOg+JUW49D\nOfjDq0HiW64QeWYu+lsEFS+2sHES82R/JB
mXWbqHy68JhDUubzAi7nRXEHkKP9N3\nXF6jD3Fyn47kN4uYeHas9eCvMFmxfv6TGzCPgfS9PH
0ZaZLM/Cp6u9l4DhyvFV9n\nuazsa9pkwlQftE8L1Xo/hGmKBdLGJDRDQo1r4eYYZlzS94tj5p
0thztTmUWZFOl2\ndOTyux8EvhpsefDYwtlJ1wxqvFdDm5WGCxeZd6YAsV/KukVQLakX1LS/i7
FEFsuL\nZ89WjWdHC2A3liKOa/y0/vy9gOY7GScvRnyUSthQ2Y0DYO89KDzY6Q6ILbpsVr8=\
n-----END CERTIFICATE-----\n"
             ],
             "fingerprint": {
               "digest":
"a3:34:5a:77:3d:14:da:a6:d7:de:7a:43:4d:e3:2c:36:35:53:74:5c:61:cf:c1:39:b
b:75:50:40:29:30:dc:2e",
```

```
                "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-----
\nMIICIzCCAckCAQAwgdQxCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG\nA1UECA
wCQ0ExFzAVBgNVBAcMDkV4YW1wbGUgQ2l0eSAyMRkwFwYDVQQJDBBFeGFt\ncGxlIFN0cmVldC
AyMR8wHQYDVQQKDBZFeGFtcGxlIE9yZ2FuaXphdGlvbiAyMRUw\nEwYDVQQLDAxFeGFtcGxlIF
VuaXQxGjAYBgNVBAMMEXRoaXJkLmV4YW1wbGUuY29t\nMSAwHgYJKoZIhvcNAQkBFhFpbmZvMk
BleGFtcGxlLmNvbTBZMBMGByqGSM49AgEG\nCCqGSM49AwEHA0IABJgGNzspp3C44WUjhgA/G0
FTFzV9SP0MiC653kGX7TqNN0+d\nOpWjTTDbcs7QgvO4n4lcxzyC+/4G1GHOs0zUhhqggZEwgY
4GCSqGSIb3DQEJDjGB\ngDB+MAkGA1UdEwQCMAAwCwYDVR0PBAQDAgOIMBMGA1UdJQQMMAoGCC
sGAQUFBwMC\nMBEGCWCGSAGG+EIBAQQEAwIHgDAdBgNVHQ4EFgQUrdDV8+P98jDwpn6M/Ibq3o
9J\ngcswHQYDVR0RBBYwFIISZm91cnRoLmV4YW1wbGUuY29tMAoGCCqGSM49BAMEA0gA\nMEUC
IQDn5/JLVu3TGZqBXodETmj6ndamg9wFi7bxow4krngQtQIgTaDXwBvl0L36\ncHEQP5At2ss8
kKB4QIxEFeesGgMkwx8=\n-----END CERTIFICATE REQUEST-----\n",
                              "public_key": "-----BEGIN PUBLIC KEY-----
\nMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEmAY3OymncLjhZSOGAD8bQVMXNX1I\n/QyILr
neQZftOo03T506laNNMNtyztCC87ifiVzHPIL7/gbUYc6zTNSGGg==\n-----END PUBLIC
KEY-----\n"
              }
            }
          ]
        }
```

Example: Querying a single CSR

The following is a sample response received when a single CSR is queried.

For more information on the `meta` object, see Message format on page 10.


```
{
     "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": {
         "digest":
"eb:46:b6:bf:dc:4e:c6:cb:81:9b:ee:fd:a1:8d:7d:72:86:3d:48:87:ba:94:e0:0c:7
9:8e:73:77:fd:5b:97:3b",
         "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-----
\nMIICPzCCAeQCAQAwgegxCzAJBgNVBAYTAlVTMQ4wDAYDVQQRDAUxMjM0NTELMAkG\nA1UECA
wCQ0ExFTATBgNVBAcMDEV4YW1wbGUgQ2l0eTEXMBUGA1UECQwORXhhbXBs\nZSBTdHJlZXQxHT
AbBgNVBAoMFEV4YW1wbGUgT3JnYW5pemF0aW9uMRcwFQYDVQQL\nDA5FeGFtcGxlIFVuaXQgMT
EXMBUGA1UECwwORXhhbXBsZSBVbml0IDIxGjAYBgNV\nBAMMEWZpcnN0LmV4YW1wbGUuY29tMR
8wHQYJKoZIhvcNAQkBFhBpbmZvQGV4YW1w\nbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQ
```

```
cDQgAEC/MAO3IIhG6zInpQxOJ9\nfFnOQlW11IoeMXHfhrhRC9OI9W77MjxRNX7gXS1WVcEQPx
gXtE9sHdc6Z8jgupIi\ncKCBmDCBlQYJKoZIhvcNAQkOMYGHMIGEMAkGA1UdEwQCMAAwCwYDVR
0PBAQDAgOI\nMBMGA1UdJQQMMAoGCCsGAQUFBwMCMBEGCWCGSAGG+EIBAQQEAwIGQDAdBgNVHQ
4E\nFgQUDr0ZP/F5s++a46mW+yIgs1CNWwYwIwYDVR0RBBwwGocEe3t7e4ISc2Vjb25k\nLmV4
YW1wbGUuY29tMAoGCCqGSM49BAMEA0kAMEYCIQCrRLitgHeDJ34VSksqwbZy\nUA0Klz6l2Ezr
RHGR0UOPbAIhAKA7u8xplNauUutkQPd4KHT5eyBMs0GUYJm1gr3r\ntZFr\n-----END
CERTIFICATE REQUEST-----\n",
                                "public_key": "-----BEGIN PUBLIC KEY-----
\nMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEC/MAO3IIhG6zInpQxOJ9fFnOQlW1\n1IoeMX
HfhrhRC9OI9W77MjxRNX7gXS1WVcEQPxgXtE9sHdc6Z8jgupIicA==\n-----END PUBLIC
KEY-----\n"
       }
     }
```


Elements of the response message body include:

Element	Type	Description	Notes
`certificate_chain`	string	The certificate chain received from the trusted CA.	
`fingerprint`	object		
`fingerprint.digest`	string	The fingerprint of the certificate, for example `ef:d3:8e:d0:81:4f:a2:8f:3b:8b:0c:dd:c7:8f:8c:7e`.	
`fingerprint.hash_algorithm`	string	The hash algorithm used to create the fingerprint, for example, `sha256`.	
`subject`	object	The subject string of the certificate.	
`extensions`	object	The list of extensions.	
`pem`	string	The certificate signing request in PEM format.	
`public_key`	string	The public key in PEM format.	

Example: Deleting a CSR

The following is a sample response received when a CSR is deleted.

For more information on the `meta` object, see Message format on page 10.

```
{
        "meta": {
           "href": "/api/pki/certificate/requests",
           "parent": "/api/pki/certificate"
        },
        "items": []
     }
```

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.

HTTP response code	Status/Error	Description
400	SyntacticError	Syntax error: Could not load PEM certificate: Unable to load certificate; `error=\"[('PEM routines', 'get_name', 'no start line')]\"`
400	CertChainValidationError	You have attempted to store a certificate chain, which could not be validated with the specified Trust Store.
400	OnlyOnCentralNode	Certificate signing requests can only be created or updated on the Central management node of the cluster.
404	NoMatchingCsrFound	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.

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

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 49. 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

Header name	Description	Required	Values
Content-Type	Specifies the type of the data sent. SPS uses the JSON format	Required	`application/json`
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.

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-----
\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2Sch
Dibk/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .........
Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END RSA
PRIVATE KEY-----\n"}'
```

The body should be:

```
{
       "certificate": "-----BEGIN CERTIFICATE-----
\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2Sch
Dibk/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .........
Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END
CERTIFICATE-----",
       "private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2Sch
Dibk/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .........
Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END RSA
PRIVATE KEY-----",
       "issuer_chain": []
 }
```

Element	Type	Description
certificate	string	The certificate in PKCS-1 PEM format (replace line-breaks with `\n`). For example: `-----BEGIN CERTIFICATE-----\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2SchDibk/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .....Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END CERTIFICATE-----`
private_key	string	The private key of the certificate, without encryption or password protection (replace line-breaks with `\n`). For example: `-----BEGIN RSA PRIVATE KEY-----\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2SchDibk/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .....`

Element	Type	Description
		`Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END RSA PRIVATE KEY-----`
`issuer_chain`	list	A comma-separated list of the Certificate Authority (CA) certificates that can be used to validate the uploaded certificate.

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;>
```

Element		Type	Description
`fingerprint`		string	The fingerprint of the certificate.
	`digest`	string	The fingerprint of the certificate, for example `ef:d3:8e:d0:81:4f:a2:8f:3b:8b:0c:dd:c7:8f:8c:7e`
	`hash_algorithm`	string	The hash algorithm used to create the fingerprint, for example, `sha256`.
`subject`	string	The subject string of the certificate.	

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 more information on the `meta` object, see Message format on page 10.

```
{
    "key": "faa96916-c85e-46ff-8697-f4cc5e596e7f",
    "meta": {
```

```
        "href": "/api/configuration/x509/faa96916-c85e-46ff-8697-f4cc5e596e7f",
        "parent": "/api/configuration/x509",
        "transaction": "/api/transaction"
    }
}
```

The response to querying a specific certificate is a JSON object that includes the parameters of the certificate, for example:

```
{
        "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/em
ailAddress=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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	SyntacticError	Syntax error: Could not load PEM key: Unsupported private key format, only PKCS-1 is supported. Encrypted private keys are not supported.
401	Unauthenticated	The requested resource cannot be retrieved because the

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	NotFound	The requested object does not exist.
405	MethodNotAllowed	The method <method> is not allowed for this node.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
	"body": {
		"access_restriction": {
			"allowed_from": [
				"10.40.0.48/24"
			],
			"enabled": true
		},
		"allow_password_auth": true,
		"bruteforce_protection": true,
		"enabled": true,
		"listen": [
			{
				"address": {
					"key":
"nic1.interfaces.ff7574025754b3df1647001.addresses.1",
					"meta": {
						"href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/1"
					}
				},
				"port": 23
			}
		],
		"public_keys": [
			{
				"comment": "key-comment anothercomment",
				"selection": "rsa",
				"value":
"AAAAB3NzaC1yc2EAAAADAQABAAABAQDTnisLCjZ3vONMXqFBIdvpZ0BY73+GdHpgoaL8YsydxJBsYg9dYTDzVVtYFVvdCVzBdcwCjyOuPwtZoYU3pLEFQ7OVoDUDPmVnl6idS/6tB2m89I5zdc02xUeCWTBpTGoOhNtc+YDmxPGZ1FQIpXCw0MT91jviWm3JydDd5YKINwvdTh8zsRT/702ZD9uZslwkQA/b2B9/hidCAkQkvs5H1B3o4laTd0JE9k90N+qbaQjVvoInr+jdXaWvrScwFVxZhb7Q1LvUL6oxW889bOWFMSa+/mnENarw6rpwfk9Ayi5uQQ2imY/tSnfgbS2RvIa1sKwUsJasDqN2lo/DuhON"
			}
		]
	},
	"key": "ssh",
	"meta": {
		"first": "/api/configuration/local_services/admin_web",
		"href": "/api/configuration/local_services/ssh",
		"last": "/api/configuration/local_services/user_web",
		"next": "/api/configuration/local_services/user_web",
		"parent": "/api/configuration/local_services",
		"previous": "/api/configuration/local_services/snmp_agent",
		"transaction": "/api/transaction"
	}
```

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 SSH server.
	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.
	allow_password_auth		boolean	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.
	bruteforce_protection		boolean	Enables protection against brute-force attacks by denying access after failed login attempts for increasingly longer period. Enabled by default.
	enabled		boolean	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.
	listen		list	Selects the network interface, IP address, and port where the clients can access the web interface.
		address	JSON object	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:

```
{
    "body": {
```

Element	Type	Description

```
        "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.

```
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.in-
terfaces.ff7574025754b3df1647001.addresses.1"
```

Element	Type	Description

		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.in-`
		`terfaces.ff7574025754b3df1647001.addresses.1",`
		`  "meta": {`
		`  "href": "/api/-`
		`config-`
		`uration/net-`
		`work/n-`
		`ics/n-`
		`ic1#interfaces/ff7574025754b3df1647001/addresses/1"`
		`  }`
		`  },`

port	integer	The port number where this local service accepts connections.

public_keys	list	Lists the public keys that can be used to authenticate on SPS. For example:
		
		`"public_keys": [`
		`  {`
		`  "comment": "user@example.com anoth-`
		`ercomment",`
		`  "key": {`
		`  "selection": "rsa",`
		`  "value": "AADDB3Nz-`
		`aC1yc2EABBADAQA...../DuhON"`
		`  }`
		`  },`
		`  {`
		`  "comment": "username@example.com",`
		`  "key": {`
		`  "selection": "rsa",`
		`  "value": "ASFDFAB3Nz-`
		`aC1yc2EAAAABIwAAASdfASF/EuQh9zc2umxX...dU="`
		`  }`
		`  }`
		`  ]`
		
		TIP: One Identity recommends using 2048-bit RSA keys (or stronger).

Elements of public_keys	Type	Description
comment	string	Comments of the public key.
key	JSON object	Contains the type of the key and the key itself. For example: `"key": { "selection": "rsa", "value": "ASFDFAB3Nz-aC1yc2EAAAABIwAAASdfASF/EuQh9zc2umxX...dU=" }`
selection	rsa	The type of the public key. Must be `rsa`.
value	string	The public key itself.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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": {
        "product": "Safeguard for Privileged Sessions",
        "product_version": "7",
        "serial": "123-456-789"
        "license_type": "term",
        "valid_not_after": "2022-12-14",
        "enterprise": true
        "limit_type": "session",
        "licensed_options": {
          "basic_proxies": 0,
          "sudo_iolog": 0,
          "analytics": true,
        }
    },
    "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"
    }
}
```

Element	Type	Description
key	string	Top level element, contains the ID of the endpoint.
body	Top level element (string)	Contains the parameters of the license.
`enterprise`	boolean	Set to true to enable evaluating the limit values of `basic_proxies` and `sudo_iologs` of `licensed_options`. When set to true, all limit values other `0` than are considered unlimited.
`license_type`	string	Contains information about the type of your license. This information is mostly relevant for your company's Sales Team.
`licensed_options`	JSON object	Contains the limit values of `basic_proxies` and `sudo_iologs`, and enabling or disabling `analytics`.
`limit_type`	host | session	• `host`: Limits the number of servers (individual IP addresses) that can be connected through SPS. • `session`: Limits the number of concurrent sessions (parallel connections) that can pass through SPS at a time (for example, `25`).
`product`	string	The official name of the product the license is applied for.
`product_version`	string	The product version number currently in use with the license.
`serial`	string	The unique serial number of the license.
`valid_not_`	date	The date when the license expires. The dates

Element	Type	Description
`after`		are displayed in YYYY-MM-DD format. NOTE: The `valid_not_after` key can have a `null` value. In this case, your license does not expire.

Elements of `licensed_options`	Type	Description
`analytics`	boolean	Set to true to enable analytics on this licensed version of SPS.
`basic_proxies`	integer	The value of the limit for sessions or hosts when using basic proxies with this licensed version of SPS.
`sudo_iolog`	integer	The value of the limit for using Sudo I/O logs with this licensed version of 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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.div' https://<IP-address-of-SPS>/api/upload/license
   ```

4. **Commit your uploaded license file.**

   Commit your uploaded license file to the `https://<IP-address-of-SPS>/api/upload/license` endpoint. For example:

   ```
   curl --cookie cookies https://<IP-address-of-SPS>/api/transaction --request PUT --data '{"message": "your commit message"}'
   ```

5. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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

Cookie name	Description	Required	Values
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 more information 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

Cookie name	Description	Required	Values
			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 more information on 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 32.

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:

```
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:

```
{
    "uri":  {
       "selection":  "url",
       "value":  "http://example.com"
     }
 }
```

To use a email address as contact info, use the following JSON object:

```
{
    "uri":  {
       "selection":  "mailto",
       "value":  "mailto:support@example.com"
     }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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`		Type	Description
`body`		JSON object	Top-level element
	`enabled`	boolean	Set to `true` and configure the other options as needed for your environment to forward session data from SPS to Splunk.
	`flush_interval`	integer [seconds]	If the Splunk server becomes unaccessible, SPS will try to resend the data when this period expires.

	`host`	JSON object	Contains the hostname or the IPv4 address of the Splunk server.
			`"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.
	`value`	string	The address of the server.
	`port`	integer	The port number where your Splunk HTTP Event Collector is accepting connections. By default, Splunk uses port 8088.
	`ssl`	JSON object	Determines if encryption is used between SPS and Splunk.
	`selection`	string	Determines if encryption is used between SPS and Splunk. Possible values: • `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

Elements of `remote_desktop_gateway`	Type	Description

		Identity recommends to use HTTPS encryption between SPS and Splunk.
		`"ssl": { "selection": "disabled" },`
		• `insecure`: Use HTTPS encryption between SPS and Splunk.
		`"ssl": { "selection": "insecure" },`
		• `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.
		`"ssl": { "certificate": "-----BEGIN CERTIFICATE----- \nMIIFPzCCAyegA\n....\n-r8lDCPoq\n0wgJ\n-----END CERTIFICATE-----\n", "selection": "secure" },`

`token`	string	The HTTP Event Collector authentication token you have generated for SPS.

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 fowards 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,

```
{
        "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 35.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
                                                }
                                        }
                                }
                        }
                ]
        }
```

Elements	Type	Description
`body`	JSON object	Top-level element
`enabled`	boolean	Set to `true` and configure the other options as needed for your environment to forward session data from SPS to an external SIEM device.
`prefix`	string	A prefix to make the data more readable. 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.

Elements		Type	Description
`targets`		JSON object	Specifies the details of the target SIEM device.
	`format`	cef | json | json_cim	The format of the message sent to the SIEM. Use the following: • `json_cim`: if using Splunk. • `cef`: if using CEF-compatible SIEMs, for example, Microsoft Azure Sentinel. • `json`: for general use.
	`name`	string	The name of the SIEM forwarder policy.
	`protocol`	JSON object	Specifies connection details to the target SIEM device. For example: `"protocol": { "selection": "syslog", "value": { "address": { "selection": "ip", "value": "192.168.1.1" }, "port": 5555, "tls": { "selection": "secure", "trusted_ca_list_ref": "1241814345d074efd1ded7" } } }`

Elements of `protocol`	Type	Description
`selection`	string	Must be `syslog`
`value`	JSON object	Contains the address of the SIEM and the TLS settings of the connection.
`address`	JSON object	Contains the type and the value of the address. For example: `"address": { "selection": "ip", "value":`

Elements of `protocol`			Type	Description

				`"192.168.1.1"`
				`},`
				
				`"address": {`
				`"selection": "fqdn",`
				`"value": "my-siem.example.com"`
				`},`

	`selection`	string	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 IP address.	
	`value`	string	The address of the server, corresponding to the format set in the `selection` field.	
	`port`		integer	The port number of the server.

	`tls`		JSON object	The security settings of the connection. For example:
				
				`tls": {`
				`"selection": "secure",`
				`"trusted_ca_list_`

Elements of `protocol`		Type	Description

			`ref": "1241814345d074ef-d1ded7"`
			`}`
			
			`"tls": {`
			`"selection": "disabled"`
			`}`

`selection`	disabled | insecure | secure	• disabled: Use an unencrypted connection. Since the data forwarded contains sensitive information, One Identity recommends to use TLS encryption between SPS and your SIEM. • insecure: Use TLS encryption, but do not validate the certificate of the SIEM. • secure: Use TLS encryption and validate the certificate of the SIEM. If you use this option, you must also set the `trusted_ca_list_ref`	

Elements of `protocol`	Type	Description
		field.
`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 32.

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,

```
{
    "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"
                    }
                }
            }
        }
    ]
}
```

4. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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

Cookie name	Description	Required	Values
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 more information 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 endpoints available under the `cluster` endpoint.

```
curl --cookie cookies https://<IP-address-of-any-node-in-cluster>/api/cluster
```

Response

The following is a sample response received.

For more information on the `meta` object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "configuration_sync",
            "meta": {
                "href": "/api/cluster/configuration_sync"
            }
        },
        {
            "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"
            }
        }
    ],
    "meta": {
        "href": "/api/cluster",
        "join_request": "/api/cluster/join_request",
        "nodes": "/api/cluster/nodes",
        "parent": "/api",
        "promote": "/api/cluster/promote",
        "status": "/api/cluster/status",
        "configuration_sync": "/api/cluster/configuration_sync"
    }
}
```


Element	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 endpoint.
`meta`	Top-level item (JSON object)	Contains the path to the endpoint.
`href`	string (relative path)	The path of the resource that returned the response.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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 more information on the `meta` object, see Message format on page 10.

```
{
    "body": {
        "address": "<IP-address-of-Central-Management-node>",
        "roles": [
            "central-management"
        ]
    },
    "meta": {
```

```
        "href": "/api/cluster/nodes/b35c54da-b556-4f91-ade5-d26283d68277",
        "parent": "/api/cluster/nodes",
        "remaining_seconds": 28800
    }
}
```

Elements		Type	Description
body		Top-level element (JSON object)	Contains the JSON object of the node.
	address	string	The IP address of the node.
	roles	string	The role of the node.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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:

```
{
       "central_management_address":  "<IP-address-of-Central-Management-
node>"
 }
```

   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 more information on 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":
"Ler7HZDFmZCxnLLgHNRfZYfORhlZS99l9vEVr5UKtJEb1d4WeaHcBmQJLs4VDWIn",
                "roles": []
        },
        "meta": {
                "href": "/api/cluster/join_request",
                "parent": "/api/cluster",
                "remaining_seconds": 600
        }
 }
```

Elements	Type	Description
`body`	Top-level element (JSON object)	Contains the JSON object of the node.
`address`	string	The IP address of the node.
`node_id`	string	A reference ID for the node.
`psk`	string	The pre-shared key of the node used for authentication.
`roles`	string	The role of the node.

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:

```
{
        "address":  "<IP-address-of-node-joined-to-cluster>",
        "node_id":  "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
        "psk":
"Ler7HZDFmZCxnLLgHNRfZYfORhlZS99l9vEVr5UKtJEb1d4WeaHcBmQJLs4VDWIn",
        "roles":  []
 },
```

For example:

```
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":
"Ler7HZDFmZCxnLLgHNRfZYfORhlZS99l9vEVr5UKtJEb1d4WeaHcBmQJLs4VDWIn","role
s":  []}'
```

If the POST request is successful, the response includes:

```
{
        "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 35.

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 more information on 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"
 }
```

Elements	Type	Description
`details`	Top-level element	Contains the IP address of the Central Management node of the cluster.
`central_management_address`	string	The IP address of the Central Management node. Not provided when no cluster has been set up yet.
`status`	string	Possible values are: • `not configured`: Displayed when no cluster has been set up yet. • `in progress`: Displayed when the join action is in progress. • `in cluster`: Displayed when the node is already in the cluster.

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 32.

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:

   NOTE: The `central-management` role can only be assigned using the `/api/cluster/promote` endpoint.

   NOTE: Ensure that each node has a search role and only one search role.

Role	Description
`managed-host`	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.
`search-master`	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.
`search-minion`	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.
`search-local`	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.

   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 35.

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 more information on the `meta` object, see Message format on page 10.

```
{
    "items": [
        {
            "key": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
            "meta": {
                "href": "/api/cluster/nodes/46f97a58-4028-467d-9a22-
9cfe78ae3e1c",
                "status": "/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"
    }
}
```

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.
`meta`	Top-level item (JSON object)	Contains links to different parts of the REST service.
`href`	string (relative path)	The path of the node that returned the response.
`status`	string (relative path)	The path to the status of the node that returned the response.

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 more information on the `meta` object, see Message format on page 10.

```
{
        "body": {
                "address": "<IP-address-of-node-to-be-queried>",
                "roles": [
                        "central-management"
                ]
        },
        "key": "b35c54da-b556-4f91-ade5-d26283d68277",
        "meta": {
                "href": "/api/cluster/nodes/b35c54da-b556-4f91-ade5-
d26283d68277",
                "parent": "/api/cluster/nodes",
                "remaining_seconds": 28800,
                "status": "/api/cluster/status/b35c54da-b556-4f91-ade5-
d26283d68277"
        }
 }
```

Elements		Type	Description
body		Top-level element (JSON object)	Contains the JSON object of the node.
	address	string	The IP address of the node.
	roles	string	The role assigned to the node.
key		string	The ID of the node.

Query the status of all nodes in the cluster

To query the status of all nodes in your cluster, complete the following steps.

1. **Query the /api/cluster/status endpoint on the Central Management node.**

   ```
   curl --cookie cookies https://<IP-address-of-Central-Management-
   node>/api/cluster/status
   ```

   The following is a sample response received.

   For more information on the `meta` object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "b35c54da-b556-4f91-ade5-d26283d68277",
      "fqdn": "sps.example.com",
      "health_status": null,
      "sync_status": "n/a",
      "meta": {
        "configuration": "/api/cluster/nodes/b35c54da-b556-4f91-ade5-
d26283d68277",
        "href": "/api/cluster/status/b35c54da-b556-4f91-ade5-d26283d68277"
      }
    },
    {
      "key": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
      "last_seen": "2018-02-08T10:00:30Z",
      "fqdn": "managed-host.cluster",
      "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": "pending",
      "configuration_sync": {
        "issues": {
          "warnings": [
           {
              "message": "Connection 'simple_ssh_connection' and local
service 'SSH' conflict on 10.30.42.42:22",
              "paths": [
                "/api/configuration/ssh/connections/12345",
                "/api/configuration/local_services/ssh"
              ]
            }
          ]
        }
      },
      "meta": {
        "configuration": "/api/cluster/nodes/46f97a58-4028-467d-9a22-
```

```
9cfe78ae3e1c",
        "href": "/api/cluster/status/46f97a58-4028-467d-9a22-9cfe78ae3e1c"
      }
    }
  ],
  "meta": {
    "href": "/api/cluster/status",
    "parent":  "/api/cluster",
    "self": "/api/cluster/status/b35c54da-b556-4f91-ade5-d26283d68277"
  }
}
```


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
`load1`	floating point	The average system load during the last one

Elements		Type	Description
		integer	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`
			`},`

Elements	Type	Description
`total_sessions`	integer (number of)	The total number of ongoing sessions.
`sync_status`	string	Indicates the status of configuration synchronization. It has the following values: • `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. • `pending`: There has been a configuration change on the Central Management node, and the change has not been synchronized yet to the node. • `outdated`: There has been some error on the node and therefore it is running an old configuration. • `not-fetched`: The node has not fetched any configuration yet. • `n/a`: The node is the Central Management node, so it is not fetching its config-

Elements	Type	Description
		uration from any other node.
meta	Top-level item (JSON object)	Contains links to different parts of the REST service.
configuration	string (relative path)	The path to the configuration of the node that returned the response.
href	string (relative path)	The path to the node that returned the response.
last_seen	string	The last time the node sent status information to the Central Management node, in ISO 8601 format.
configuration_sync	Top-level item (JSON object)	
issues	Top-level item (JSON object)	The issues that occurred during configuration synchronization.

Elements of issues	Type	Description
warning	Top-level item (JSON object)	
message	string	Human-readable text explaining why the warning occurred.
details	array	List of additional information about the warning (for example, the path where the warning occurred).
error	Top-level item (JSON	

Elements of `issues`	Type	Description
	object)	
`type`	string	The type of the error.
`message`	string	Human-readable text explaining why the error occurred.
`details`	JSON object	List of additional information about the error (for example, the path where the error occurred).

Query the status of one particular 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 more information on 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": {
        "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"
    }
}
```


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
`load1`	floating point integer	The average system load during the last one minute. The values

Elements		Type	Description
			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": {`
	total_sessions	integer (number	The total number of ongoing sessions.

Elements	Type	Description
	of)	
`sync_status`	string	Indicates the status of configuration synchronization. It has the following values: • `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. • `pending`: There has been a configuration change on the Central Management node, and the change has not been synchronized yet to the node. • `outdated`: There has been some error on the node and therefore it is running an old configuration. • `not-fetched`: The node has not fetched any configuration yet. • `n/a`: The node is the Central Management node, so it is not fetching its configuration from any other node.

Elements		Type	Description
meta		Top-level item (JSON object)	Contains links to different parts of the REST service.
	configuration	string (relative path)	The path to the configuration of the node that returned the response.
	href	string (relative path)	The path to the node that returned the response.
last_seen		string	The last time the node sent status information to the Central Management node, in ISO 8601 format.
configuration_sync		Top-level item (JSON object)	
	issues	Top-level item (JSON object)	The issues that occurred during configuration synchronization.

Elements of issues		Type	Description
warning		Top-level item (JSON object)	
	message	string	Human-readable text explaining why the warning occurred.
	details	array	List of additional information about the warning (for example, the path where the warning occurred).
error		Top-level item (JSON object)	
	type	string	The type of the error.

Elements of `issues`	Type	Description
`message`	string	Human-readable text explaining why the error occurred.
`details`	JSON object	List of additional information about the error (for example, the path where the error occurred).

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, and management 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:

```
$ curl ... https://<url-of-Central-Management-node>/api/configuration/rdp/connections/<id-of-the-connection-policy>
```

```
{
       "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:

```
$ curl ... https://<url-of-Managed-Node>/api/configuration/rdp/connections/<id-of-the-connection-policy>
```

```
{
    "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:

```
$ 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`.

If you need assistance with writing customized transformations, contact our Professional Services Team, and a One Identity Service Delivery Engineer will 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 32.

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:

```
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 more information on the `meta` object, see Message format on page 10.

```
{
       "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
       }
 }
```

Elements	Type	Description
body	Top-level element (JSON object)	
api	string	Always "1.0".
default_configuration	string	Contains the default configuration of the plugin if there is one.
description	string	The description of what the plugin does.
name	string	The name of the plugin.
path	string	The path to the plugin.
scb_max_version	string	The plugin is compatible with SPS versions not later than this one.

Elements	Type	Description
scb_min_version	string	The plugin is compatible with SPS versions not earlier than this one.
version	string	The version number of the plugin.
key	string	The ID of the plugin.

3. **To enable the plugin**

   Replace `/api/cluster/configuration_sync_plugin` with:

```
{
      "enabled": true,
      "plugin": "<'key' from-response-of-last-creation>",
      "configuration": ""
}
```

   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": true, "plugin": "794a5e17-b8be-4426-8596-
0dfc129c06ef", "configuration": ""}'
```

   The following is a sample response received:

```
{
      "plugin": {
            "key": "794a5e17-b8be-4426-8596-0dfc129c06ef",
            "meta": {
                  "href": "/api/configuration/plugins/configuration_
sync/794a5e17-b8be-4426-8596-0dfc129c06ef"
            }
      }
}
```

4. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

2. **To disable the plugin, replace /api/cluster/configuration_sync_plugin with:**

   ```
   {
         "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:

   ```
   {
         "plugin": {
               "key": null,
               "meta": {}
         }
   }
   ```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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

Cookie name	Description	Required	Values
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 more information 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 more information on 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 357	Resolve the hostname of a computer or server to IP addresses.
Testing LDAP server connection on page 359	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 41.

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

Cookie name	Description	Required	Values
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 more information 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 resolves the hostname to IP addresses.

```
curl -X POST -b "${COOKIE_PATH}" https://<IP-address-of-SPS>/api/tools/hosts-by-name
```

```
{
        "hostname": "example.org",
        "protocol": "TCP"
      }
```

Elements of the request message body include:

Element	Type	Required	Description	Notes
`hostname`	string	Required	The unique identifier that serves as the name of the computer or server whose IP address you want to resolve.	
`protocol`	string	Required	The type of Internet Protocol used to address and route packets of data.	Possible values are: • TCP • UDP

Response

When resolving a hostname to IP addresses, the response is the following.

For more information on the `meta` object, see Message format on page 10.

```
{
        "ipv4": [
           "93.184.216.34",
           "93.184.216.35"
        ],
        "ipv6": [
           "2606:2800:220:1:248:1893:25c8:1946",
           "2606:2800:220:1:248:1893:25c8:1947"
        ]
     }
```

Elements of the response message body include:

Element	Type	Description	Notes
`ipv4`	string array	The Internet Protocol type of the IP addresses is version 4.	
`ipv6`	string array	The Internet Protocol type of the IP addresses is version 6.	

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.

HTTP response code	Status/Error	Description
`400`	Syntactic Error	The protocol you provided is not valid. Use `TCP` or `UDP` as value instead.
`400`	HostnameCannotBeResolved	The hostname you provided cannot be resolved. Check the following: • The hostname you provided is valid. • The hostname is available on the Internet.
`401`	Unauthenticated	Unauthenticated users cannot query the IP addresses of a host.

For more information and a list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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",
      "parent": "/api/tools",
      "policies": "/api/tools/ldaptest/policies",
      "remaining_seconds": 600
    }
  }
```

Operations

Operations with the `/ldaptest` endpoint include:

Operation	HTTP method	URL	Notes
List endpoints for testing LDAP server availability	GET	`/api/tools/ldaptest`	
Testing a LDAP server	POST	`/api/tools/ldaptest/policies`	
Testing an AAA LDAP server	POST	`/api/tools/ldaptest/aaa`	

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 more information on the `meta` object, see Message format on page 10.

```
{
        "meta": {
           "href": "/api/tools/ldaptest/policies",
           "ldap_server": "/api/configuration/policies/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.

   ```
   curl -k -X GET -b "${COOKIE_PATH}" https://<IP-address-of-
   SPS>/api/configuration/aaa/ldap_servers
   ```

3. Select the key of the AAA LDAP server that you want to test.
4. Use the key to test whether you can connect to the AAA LDAP server.

   ```
   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 more information on the `meta` object, see Message format on page 10.

```
{
        "meta": {
           "href": "/api/tools/ldaptest/policies",
           "ldap_server": "/api/configuration/aaa/ldap_
servers/200796612861e564724534a",
           "parent": "/api/tools/ldaptest",
           "remaining_seconds": 600
        },
        "success": true
     }
```

Elements of the response message body include:

Element	Type	Description	Notes
`success`	boolean	Indicates that the connection to the provided LDAP server could be established.	Possible values are: • `true` - the test was successful • `false` - the test was not successful
`meta.href`	string (relative path)	Path of the resource that returned the response. When creating a new object, this is the URL of the created object.	
`meta.ldap_server`	string (relative path)	Identifier of the LDAP server that was tested.	
`meta.parent`	string (relative path)		
`meta.remaining_seconds`	integer	Time left until the session times out in seconds.	SPS closes idle sessions after a period of inactivity. This value shows the number of seconds left until the timeout. For more information on setting the session timeout, Login settings.

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.

HTTP response code	Status/Error	Description
400	TransactionMustBeClosed	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.
400	NoSuchLDAPServer	There are no LDAP servers configured with the identifier specified in the request: `--data '{"key": "<invalid value>"}'` Make sure that you use a valid key from the list of configured LDAP servers. This can also occur if you attempt to test an AAA LDAP server using the `/ldaptest/policies` endpoint or vice versa.
400	LDAPServerWithNoBindPassword	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

HTTP response code	Status/Error	Description
		no credentials configured for that LDAP server. • valid: a `bind_dn` and a `bind_password` are both configured, or there is no `bind_dn` and no `bind_password` • invalid: there is a `bind_dn`, but no `bind_password`, or there is a `bind_password`, but no `bind_dn`
400	LDAPServerWithNoBindDn	See LDAPServer-WithNoBindPassword.
`500`	LDAPConnectionFailure	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.

For more information and a list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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**.

Cookie name	Description	Required	Values
			For more information 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.

```
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 more information on 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"
            }
          }
```

```
        },
        "channel": "#drawing"
      },
      {
        "actions": {
          "audit": true,
          "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"
            }
          }
        },
        "channel": "cliprdr"
      }
    ]
  }
}
```

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 464.

Element	Type	Description
		• For details on Citrix ICA-specific channels, see ICA channels on page 494. • For details on RDP-specific channels, see RDP channels on page 572. • For details on SSH-specific channels, see SSH channels on page 612. • For details on Telnet-specific channels, see Telnet channels on page 668. For example: `"channel": "#drawing",`

Element		Type	Description
`actions`		JSON object	The list of actions to perform when the Content policy matches the analyzed traffic. All actions are boolean values (`true` or `false`)
	`audit`	boolean	Set to `true` to record the activities of the channel into audit trails. Possible values: `true` or `false`

	`content_policy`	JSON object	Specifies the Content policy to use in the channel, otherwise its value is `null` (which is the default). For details on Content policies, see Real-time content monitoring with Content Policies For example:
			`"content_policy": {`
			`"key": "<object-id>",`
			`}`

	`four_eyes`	boolean	Set to `true` to require four-eyes authorization to access the channel. For details, see "Configuring four-eyes authorization" in the Administration Guide. Possible values: `true` or `false`

Element		Type	Description
`allowed_for`		JSON object	Specifies the access control rules of the channel.
	`clients`	list	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:

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 437.
`remote_groups`	list	You can control channel access during authentication to the remote server with blacklists or whitelists of

Element	Type	Description
		user groups. You can use local user lists on SPS, or LDAP groups. For example: `"remote_groups": ["group1", "group2"],` To configure local user lists, see User lists on page 437.

`servers`	list	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:
		`"servers": [`
		`  "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.
		Alternatively, you can configure a custom DNS server to be used for target selection `custom_dns` field of the Connection Policy.

Element		Type	Description
	`time_policy`	JSON object	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 422. For example: `"time_policy": { "key": "-100", }`

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
  "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"
      }
```

```
    },
    {
      "key": "indexing",
      "meta": {
        "href": "/api/configuration/policies/indexing"
      }
    },
    {
      "key": "ldap_servers",
      "meta": {
        "href": "/api/configuration/policies/ldap_servers"
      }
    },
    {
      "key": "signing_cas",
      "meta": {
        "href": "/api/configuration/policies/signing_cas"
      }
    },
    {
      "key": "time_policies",
      "meta": {
        "href": "/api/configuration/policies/time_policies"
      }
    },
    {
      "key": "trusted_ca_lists",
      "meta": {
        "href": "/api/configuration/policies/trusted_ca_lists"
      }
    },
    {
      "key": "user_databases",
      "meta": {
        "href": "/api/configuration/policies/user_databases"
      }
    },
    {
      "key": "userlists",
      "meta": {
        "href": "/api/configuration/policies/userlists"
      }
    },
    {
      "key": "usermapping_policies",
      "meta": {
        "href": "/api/configuration/policies/usermapping_policies"
      }
```

```
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa",
    "href": "/api/configuration/policies",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/private_keys",
    "parent": "/api/configuration",
    "previous": "/api/configuration/plugins",
    "transaction": "/api/transaction"
  }
}
```


Endpoint	Description
`aa_plugin_instances`	Authentication and Authorization plugin policies
`analytics`	Analytics.
`archive_cleanup_policies`	Archive/Cleanup policies.
`audit_policies`	Audit trail encryption, timestamping, and signing.
`backup_policies`	Backup policies.
`content_policies`	Actions for detected commands, screen content, credit card information, and window titles.
`credentialstores`	Local and external credential stores.
`indexing`	Languages for Optical Character Recognition (OCR).
`ldap_servers`	LDAP servers.
`signing_cas`	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.
`time_policies`	Time policies.
`trusted_ca_lists`	Trusted Certificate Authorities (CAs), and options for restricting the accepted certificates.
`user_databases`	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.

Endpoint	Description
`userlists`	Local white- or blacklists of usernames that allow fine-control over who can access a connection or a channel.
`usermapping_policies`	Usermapping policies describe who can use a specific username to access the remote server.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
      },
```

```
      "path": "/data/backup"
    },
    "start_times": [
      "10:10"
    ],
    "template": "PROTOCOL/CONNECTION/ARCHIVEDATE/",
    "retention_days": 30
  }
}
```

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.
`include_node_id_in_path`		boolean	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 archiving data to the same location, and so overwriting each other's data and resulting in data loss.
`notification_event`		Top level element	
	`type`	string (`all` | `errors-only` | `none`)	• `all`: Sends notification emails on all archive-related events. • `errors-only`: Sends notification emails only on archive-related errors. • `none`: Sends no archive-related notification emails.
	`send_filelist`	boolean	This is meaningful only if `notification_event` is set to `all`. `True` if the list of files are included in the notification e-mail.
	`file_count_limit`	integer	This is meaningful only if `notification_event` is set to `all` and `send_filelist` is set to `True`. The maximum number of files that are included in the notification e-mail.
`target`		Top level element	Defines the address of the archive server,

Element		Type	Description
			which protocol to use to access it, and other parameters. SPS can be configured to use the SMB/CIFS, and NFS protocols to access the archive server.
	`type`	string (`smb` | `nfs` | `none`)	• `smb`: Move data to a remote server using SMB/CIFS • `nfs`: Move data to a remote server using NFS • `none`: Cleanup data. Data is deleted from SPS forever and cannot be recovered.
	`server`	Top level element	
	`domain`	string	Only if `type` is set to `smb`. The domain name of the target server
	`protocol_version`	string	Only if `type` is set to `smb`. The SMB protocol to use when SPS connects to the server. Servers are usually backwards compatible with earlier protocol versions (for example, a server that supports version 2.1 supports versions 2.0 and 1.0 as well).
	`share`	string	Only if `type` is set to `smb`. The name and directory path of the share in the following format: `share_name/path/to/directory`
	`authentication`	Top level element	Only if `type` is set to `smb`.
	`path`	string	The path to the archive directory on the target server
`start_times`		list of strings	The time when the archive process starts in `H:MM` or `HH:MM` format.
`template`		string	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

Element	Type	Description
		possible: • `PROTOCOL/CONNECTION/ARCHIVEDATE/` • `ARCHIVEDATE/CONNECTION/PROTOCOL/` • `CONNECTIONDATE/PROTOCOL/CONNECTION/` • `ARCHIVEDATE/` • `CONNECTIONDATE/`
`retention_days`	integer (days)	Data older than this value is archived to the external server. The archived data is deleted from SPS.

Elements of `server`	Type	Description
`server`	Top level element	
`selection`	string (`ip` | `fqdn`)	• `ip`: IP address • `fqdn`: Hostname
`value`	string	The IP address or the hostname of the remote server

Elements of `authentication`	Type	Description
`authentication`	Top level element	Only if `type` is set to `smb`.
`selection`	string (`password` | `anonymous`)	• `password`: To log on using a username and password. • `anonymous`: To log on anonymously.
`username`	string	Only if `selection` is set to `password`. The username used to log on to the remote server
`password`	string	Only if `selection` is set to `password`. The password corresponding to the username

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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/78101850949e47437dd91d",
    "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/9161063345713f11489305",
    "transaction": "/api/transaction"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the policy.
body	Top level element	The configuration elements of the policy.

Element	Type	Description
	(string)	
encryption	Top level element	Audit trail encryption settings.
name	string	The name of the policy. This name is also displayed on the SPS web interface. It cannot contain whitespace.
signing	Top level element	Audit trail signing settings.
enabled	boolean	Set to `true` to enable audit trail signing. If signing is enabled, the `x509_identity` element is also required.
x509_identity	string	Required for signing audit trails. References the identifier of the X.509 certificate stored on SPS. You can configure certificates at the `/api/configuration/x509/` endpoint. 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).
timestamping	boolean	Set to `true` to timestamp the audit trail.

Elements of `encryption`	Type	Description
certificates	Top level list	Contains the encrypting certificates.
certificate	string	The encrypting certificate. You can replay an encrypted audit trail with the private key of the encrypting certificate.
four_eyes_certificate	string	Additional certificate for joint (4-eyes) encryption. You can only replay a jointly encrypted audit trail with the private keys of both certificates.
different_certificates_	Top level item	Configures encrypting upstream traffic separ-

Elements of `encryption`			Type	Description
`for_upstream`				ately.
	`certificates`		Top level list	The certificates for encrypting upstream traffic.
		`certificate`	string	The encrypting certificate. You can replay an encrypted upstream with the private key of the encrypting certificate.
		`four_eyes_certificate`	string	Additional certificate for joint (4-eyes) encryption. You can only replay a jointly encrypted upstream with the private keys of both certificates.
	`enabled`		boolean	Set to `true` to encrypt the upstream traffic with separate certificate(s). If upstream encryption is enabled, the `certificates` element is required.
`enabled`			boolean	Set to `true` to enable encrypting audit trails. If encryption is enabled, the `certificates` and `different_certificates_for_upstream` elements are required.

Examples:

Disable encryption, signing, and timestamping.

```
{
    "encryption": {
       "enabled": false
    },
    "name": "default",
    "signing": {
       "enabled": false
    },
    "timestamping_enabled": false
 }
```

Encrypt upstream traffic only (single certificate).

```
{
    "encryption": {
       "certificates": [],
       "different_certificates_for_upstream": {
          "certificates": [
             {
                "certificate": "<cert>",
                "four_eyes_certificate": null
             }
          ],
          "enabled": true
       },
       "enabled": true
    },
    "name": "Upstream_only",
    "signing": {
       "enabled": false
    },
    "timestamping_enabled": false
 }
```

Enable signing and timestamping, no traffic encryption.

```
{
    "encryption": {
       "enabled": false
    },
    "name": "Sign_and_timestamp",
    "signing": {
       "enabled": true,
       "x509_identity": {
          "key": "9508db81-4a3f-45a7-a2b1-a86f71c56416",
          "meta": {
             "href": "/api/configuration/x509/9508db81-4a3f-45a7-a2b1-
a86f71c56416"
          }
       }
    },
    "timestamping_enabled": true
 }
```

Enable signing and timestamping, and encrypt traffic with a single certificate (no separate upstream encryption).

```
{
  "encryption": {
    "certificates": [
      {
        "certificate": "<cert>",
```

```
        "four_eyes_certificate": null
      }
    ],
    "different_certificates_for_upstream": {
      "enabled": false
    },
    "enabled": true
  },
  "name": "API_audit_pol",
  "signing": {
    "enabled": true,
    "x509_identity": {
      "key": "d0286f64-41aa-45e1-ab19-830ac2f99f57",
      "meta": {
        "href": "/api/configuration/x509/d0286f64-41aa-45e1-ab19-830ac2f99f57"
      }
    }
  },
  "timestamping_enabled": true
}
```


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:


```
-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC
Q0ExEDAOBgNVBAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT
B0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi
aXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4XDTE2MDQyMjE2
MDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP
bnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD
VQQLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN
AQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS
59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw
0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc
3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs
X5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX
hJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
R5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP
AB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy
```

```
0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7XDB
uNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc
N5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt
qYyyD1ld89OUEYZ+aJQd/A==
-----END CERTIFICATE-----
```


The same certificate, as accepted by the SPS API:


```
"certificate": "-----BEGIN CERTIFICATE-----
\nMIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC\nQ0ExEDAOBgNV
BAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDU
RvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxh
Yml0Lmh1MB4XDTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDg
YDVQQIEwdP\nbnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQ
QLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxA
YmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7a
hTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY
6vDU9/4v9dDiQfEPlauw\n0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2
oSobPc\n3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv
0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4di
KU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh
6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57
IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7X
DB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc\nN5+4ImYnFN
xSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyyD1ld89OUEYZ+aJQd/A==
\n-----END CERTIFICATE-----\n"
```


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 32.

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:

```
{
    "key":  "1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
    "meta":  {
       "href":  "/api/configuration/policies/audit_policies/1e089e2a-76b4-
4079-94e3-c83ebc74dc2e",
       "parent":  "/api/configuration/policies/audit_policies",
       "transaction":  "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
403	Unauthorized	The requested resource cannot be retrieved because the

Code	Description	Notes
		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	NotFound	The requested object does not exist.

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.

```
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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
       ]
    }
 }
```

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.
`include_node_id_in_path`	boolean	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

Element		Type	Description
			the same location, and so overwriting each other's data and resulting in data loss.
`notification_event`		Top level element	
	`type`	string (`all` | `errors-only` | `none`)	• `all`: Sends notification emails on all backup-related events. • `errors-only`: Sends notification emails only on backup-related errors. • `none`: Sends no backup-related notification emails.
	`send_filelist`	boolean	This is meaningful only if `notification_event` is set to `all`. `True` if the list of files are included in the notification e-mail.
	`file_count_limit`	integer	This is meaningful only if `notification_event` is set to `all` and `send_filelist` is set to `True`. The maximum number of files that are included in the notification e-mail.
`target`		Top level element	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.
	`type`	string (`rsync` | `smb` | `nfs`)	• `rsync`: Rsync over SSH • `smb`: Copy data to a remote server using SMB/CIFS • `nfs`: Copy data to a remote server using NFS
	`server`	Top level element	
	`domain`	string	Only if `type` is set to `smb`. The domain name of the target server
	`protocol_version`	string	Only if `type` is set to `smb`.

Element		Type	Description
			The SMB protocol to use when SPS connects to the server. Servers are usually backwards compatible with earlier protocol versions (for example, a server that supports version 2.1 supports versions 2.0 and 1.0 as well).
	`share`	string	Only if `type` is set to `smb`. The name and directory path of the share in the following format: `share_name/path/to/directory`
	`authentication`	Top level element	Only if `type` is set to `smb`.
	`username`	string	Only if `type` is set to `rsync`. The username used to log on to the remote server
	`path`	string	The path to the backup directory on the target server
	`auth_key`	JSON object	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 280. For example: `"auth_key": { "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", "meta": { "href": "/api/-configuration/private_keys/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" } },`
	`host_key`	Top level element or string	Only if `type` is set to `rsync`.

Element	Type	Description
port	integer	Only if `type` is set to `rsync`. The port number of the SSH server running on the remote machine
start_times	list of strings	The time when the archive process starts in `H:MM` or `HH:MM` format.

Elements of `server`	Type	Description
server	Top level element	
selection	string (`ip` | `fqdn`)	• `ip`: IP address • `fqdn`: Hostname
value	string	The IP address or the hostname of the remote server

Elements of `authentication`	Type	Description
authentication	Top level element	Only if `type` is set to `smb`.
selection	string (`password` | `anonymous`)	• `password`: To log on using a username and password. • `anonymous`: To log on anonymously.
username	string	Only if `selection` is set to `password`. The username used to log on to the remote server
password	string	Only if `selection` is set to `password`. The password corresponding to the username

Elements of host_key	Type	Description
host_key	Type: string	Only if `type` is set to `rsync`. 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: `"host_key": "ssh-rsa AAAAB3Nz-aC1yc2EAAAADAQABAAABAQDmIDa1PuJFzgvZvPs9hzgvMd/9WIn4J7RBFuO769g/OgTvCRT-grF8IM/0iN0YzcUM3IGyPnJ1OlLE2Gb6CxVvEcjP6pme7JroAWo039wQHR3Rxl1KoEmC+0EO-ImQycIdAS7-grWNwD2VB2S7iyFErZhqRx-hGJPKbR/kF3lQ3dGt-t3pr4+R6wnU9lZ7RSETfB+N09FE4f5Nqy+VEShg-dc66ElFRXXVilmiTnIMAy-im3T7UVNgRdZYIUAZ79tkyTp6I+DZ7k7BG9TYwdBjh-wr0eVL56ILxpXylpzWONuMhHxLKsL42NfmeagjVUD1CJVOrfaGjCVGEeS3iQs6GVVxe78n"` When querying, the public key of the host will always be displayed in the `selection` and `value` elements.
selection	string	The algorithm the key is based on.

Elements of host_key	Type	Description
n	(dsa | dss | rsa)	
value	string	The public key of the host.

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:


```
{
    "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-XXXXXXXXXXXX",
             "meta": {
                "href": "/api/configuration/private_keys/XXXXXXXX-XXXX-
XXXX-XXXX-XXXXXXXXXXXX"
             }
          },
          "host_key": {
             "selection": "rsa",
             "value":
"AAAAB3NzaC1yc2EAAAADAQABAAAAYQCsU80IBrJbOlqCi03qZK+FtgS783VKE1TVZBtDQlsXJ
9FXu6KNBvqvSAjcXiWY+izqn+P14UVRY1vOdz7WwLIWOUoTKHfPMqv3bdjwM4Bhd26POWSFyDf
46yx1YzvMwgc="
          },
          "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

Cookie name	Description	Required	Values
session_id	Contains the authentication	Required	The value of the session ID cookie received from the REST server in the authentication

Cookie name	Description	Required	Values
	token of the user		response, for example, **a1f71d030e657634730b9e887cb59a5e56162860**. For more information 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 Content policy object.

```
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 more information on the `meta` object, see Message format on page 10.


```
{
  "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": [],
```

```
        "remote_groups": []
      }
    ]
  }
}
```

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		JSON object	Top level element, contains the configuration properties of the object.
	actions	JSON object	The list of actions to perform when the Content policy matches the analyzed traffic. All actions are boolean values (true or false)
	event	JSON object	Specifies the event that triggers an action.
	gateway_groups	list	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: `"gateway_groups": ["group1", "group2"],`
	remote_groups	list	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: `"remote_groups": ["group1", "group3"],`

Element		Type	Description
actions		JSON object	The list of actions to perform when the Content policy matches the analyzed traffic. All actions are boolean values (true or false)
	log	boolean	Log the event in the system logs. Possible values: true or false

Element	Type	Description
`terminate`	boolean	Immediately terminate the connection. Possible values: `true` or `false`
`notify`	boolean	Send an e-mail or SNMP alerts about the event. Possible values: `true` or `false`
`store_in_connection_database`	boolean	Store the event in the connection database of SPS. Possible values: `true` or `false`

Element	Type	Description
`event`	JSON object	Specifies the event that triggers an action.

`ignore`	list	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:
		`"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

Element	Type	Description
		afterward.

`match`	list	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:
		`"match": [`
		`"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 afterward.

`selection`	string	The type of event that you want to monitor. • `command`: The commands executed in the `session-shell` channel of SSH connections, or in Telnet connections.

Element	Type	Description
		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 32.

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:

```
{
    "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 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	The requested resource cannot be retrieved because the client is not authenticated and the resource requires author-

Code	Description	Notes
		ization 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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 available LDAP server configurations.

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

The following command retrieves the properties of a specific LDAP server.

```
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 more information on the `meta` object, see Message format on page 10.


```
{
    "items": [
       {
          "key": "3548834825727acc530407",
          "meta": {
             "href": "/api/configuration/policies/ldap_
servers/3548834825727acc530407"
          }
       }
    ],
    "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.


```
{
    "key": "posix-simple",
    "body": {
        "name": "posix-simple",
        "schema": {
            "selection": "posix",
            "membership_check": {
                enabled": true,
                "member_uid_attribute": "memberUid"
            },
            "memberof_check": {
```

```
                "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
    }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the LDAP server configuration.
body	Top level element (string)	Contains the properties of the LDAP server.
user_base_dn	string	Name of the DN to be used as the base of queries regarding users. 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.

Element	Type	Description
`group_base_dn`	string	Name of the DN to be used as the base of queries regarding groups. 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.
`bind_dn`	string	The Distinguished Name that SPS should use to bind to the LDAP directory.
`bind_password`	string	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).
`encryption`	Top level item	Configuration settings for encrypting the communication between SPS and the LDAP server.
`generated_publickey_attribute`	string	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.
`name`	string	Top level element, the name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.
`publickey_attribute`	string	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.
`schema`	Top level item	Contains the configuration settings for the AD schema.
`servers`	Top level list	Contains the addresses and ports of the LDAP servers.

Elements of `encryption`		Type	Description
`selection`		string	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 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. • `starttls` Opportunistic TLS.
`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.

Elements of `encryption`	Type	Description
		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).
`selection`	string	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 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. • `starttls` Opportunistic TLS.
`server_certificate_check`	Top level item	Must be used with the `enabled` child element. Configuration settings for verifying the LDAP server's certificate.
`enabled`	boolean	Must be used with the `server_certificate_check` parent element. Set to `true` to verify the LDAP server's certificate using the certificate of a Certificate Authority (CA).

Elements of `encryption`	Type	Description
`server_certificate_ca`	string	Must be used if the `enabled` element is set to `true`. The certificate of the CA.

Elements of `servers`	Type	Description
`host`	Top level item	Contains the address of the LDAP server.
`selection`	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.
`value`	string	The address of the LDAP server.
`port`	int	The port of the LDAP server.

Elements of `schema`	Type	Description
`selection`	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.
`membership_check`	Top level element	
`enabled`	boolean	POSIX: Enables POSIX primary and supplementary group membership checking. AD: Enables Active Directory specific non-primary group membership checking.
`nested_`	boolean	Must be used if the `selection` element is set to

Elements of schema		Type	Description
	groups		ad. Enable nested groups allows AD nested group support.
	member_uid_attribute	string	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.
memberof_check		Top level element	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.
	enabled	boolean	To enable memberof_check, set it to true.
	memberof_user_attribute	string	Must be used if the memberof_check is set it to true. The name of the user attribute (for example, memberOf) that contains the group DNs.
username_attribute		string	Must be used if the selection element is set to posix. Attribute name of the username (user ID).
user_dn_in_groups		Top level list	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. For example: `"user_dn_in_groups": [ { "object_class": "groupOfNames", "attribute": "member" }, { "object_class": "groupOfU-niqueNames", "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

```
{
  "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>"
```

```
      },
      "port": <server-port>
    }
  ],
  "user_base_dn": "<basedn>",
  "group_base_dn": "<basedn>",
  "bind_dn": "<binddn>",
  "bind_password": "<bind-password>",
  "encryption": {
    "client_authentication": {
      "enabled": false
    },
    "selection": "ssl",
    "server_certificate_check": {
      "enabled": false
    }
  },
  "publickey_attribute": "<sshPublicKey>",
  "generated_publickey_attribute": null
}
```

Example: Configure a Microsoft Active Directory server with mutual authentication, and the verification of the server's X.509 certificate

```
{
  "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:

```
-----BEGIN CERTIFICATE-----
MIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC
Q0ExEDAOBgNVBAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT
B0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi
aXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4XDTE2MDQyMjE2
MDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP
bnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD
VQQLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN
AQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS
59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw
0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc
3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs
X5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX
hJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
R5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP
AB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy
0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7XDB
uNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc
N5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt
qYyyD1ld89OUEYZ+aJQd/A==
-----END CERTIFICATE-----
```


The same certificate, as accepted by the SPS API:


```
"certificate": "-----BEGIN CERTIFICATE-----
\nMIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC\nQ0ExEDAOBgNV
BAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDU
RvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxh
Yml0Lmh1MB4XDTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDg
YDVQQIEwdP\nbnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQ
QLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxA
YmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7a
hTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY
6vDU9/4v9dDiQfEPlauw\n0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2
oSobPc\n3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv
0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4di
KU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh
6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57
IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7X
DB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc\nN5+4ImYnFN
xSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyyD1ld89OUEYZ+aJQd/A==
\n-----END CERTIFICATE-----\n"
```


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 32.

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 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
    "items":  [
       {
          "key":  "991699365727ac4eb4606",
          "meta":  {
             "href":  "/api/configuration/policies/signing_
cas/991699365727ac4eb4606"
          }
       }
    ],
    "meta":  {
       "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"
  }
}
```

Element	Type	Description
key	string	Top level element, contains the ID of the signing CA.
body	Top level element (string)	Contains the properties of the signing CA.
ca	string	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).
name	string	The name of the signing CA. This name is also displayed on the SPS web interface. It cannot contain whitespace.

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 32.

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:

```
{
    "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 35.

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 32.

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 35.

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 41.

HTTP response code	Status/Error	Description
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
400	Bad Request	The referenced certificate is not a valid CA certificate.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
session_id	Contains the authentication token of the	Required	The value of the session ID cookie received from the REST server in the authentication response, for example,

Cookie name	Description	Required	Values
	user		**a1f71d030e657634730b9e887cb59a5e56162860**. For more information 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 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 more information on 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": [
```

```
      [
        "0:00",
        "23:59"
      ]
    ],
    "Thu": [
      [
        "0:00",
        "23:59"
      ]
    ],
    "Tue": [
      [
        "0:00",
        "23:59"
      ]
    ],
    "Wed": [
      [
        "0:00",
        "23:59"
      ]
    ],
    "name": "7x24"
}
```

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.

`Fri`	list	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:
		`"Sat": [],`
		To allow access for the whole day, use `0:00` for the starting time, and `23:59`for the end. For example:
		"Wed": [["0:00", "23:59"] You can list multiple intervals for a day, for example:
		`"Wed": [ [ "8:00", "18:00"`

Element	Type	Description
		```], ["19:00", "22:00"] ```
Sat	list	
Sun	list	
Tue	list	
Wed	list	
Thu	list	

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

Cookie name	Description	Required	Values
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 more information 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 trusted CAs.

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

The following command retrieves the properties of a specific CA.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
    "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.

```
{
  "body": {
    "authorities": [
      {
        "certificate": "<cert>",
        "crl": "<url-of-revocation-list>"
      }
    ],
```

```
    "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"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the CA.
body	Top level element (string)	Contains the properties of the CA.
authorities	Top level list	Contains the certificates and the Certificate Revocation Lists (CLR) of the trusted CAs. You can add multiple certificate and CRL pairs.
certificate	string	The certificate of the trusted CA.
crl	string	The URL of the Certificate Revocation List of the CA.

Element		Type	Description
dn_check		Top level item	Certificates are only accepted if their content matches the configured values.
	altEmailAddress	string	The certificate is only accepted if its alternative e-mail address matches the value of the altEmailAddress element.
	c	string	The certificate is only accepted if its country matches the value of the c element.
	cn	string	The certificate is only accepted if its common name matches the value of the cn element.
	emailAddress	string	The certificate is only accepted if its e-mail address matches the value of the emailAddress element.
	l	string	The certificate is only accepted if its locality matches the value of the l element.
	o	string	The certificate is only accepted if its organization name matches value of the o element.
	ou	string	The certificate is only accepted if its organization unit name matches value of the ou element.
	st	string	The certificate is only accepted if its state or province matches value of the st element.
dns_lookup		boolean	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.
name		string	The name of the trusted CA. This name is also displayed on the SPS web interface. It cannot contain whitespace.

Element	Type	Description
`strict_hostcheck`	boolean	Set to `true` to configure only accepting certificates where the Common Name of the certificate contains the hostname or the IP address of the host showing the certificate.

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-----
MIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC
Q0ExEDAOBgNVBAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT
B0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi
aXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4XDTE2MDQyMjE2
MDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP
bnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD
VQQLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN
AQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS
59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw
0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc
3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs
X5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX
hJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
R5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP
AB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy
0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7XDB
uNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc
N5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt
qYyyD1ld89OUEYZ+aJQd/A==
-----END CERTIFICATE-----
```


The same certificate, as accepted by the SPS API:


```
"certificate": "-----BEGIN CERTIFICATE-----
\nMIIDnDCCAoQCCQDc536Ob5tPQTANBgkqhkiG9w0BAQUFADCBjzELMAkGA1UEBhMC\nQ0ExEDAOBgNV
BAgTB09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDU
RvY3VtZW50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxh
Yml0Lmh1MB4XDTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDg
YDVQQIEwdP\nbnRhcmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQ
QLEw1Eb2N1bWVudGF0aW9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxA
```

```
YmFsYWJpdC5odTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7a
hTeyaIjK52FQUXqxG8okOSD+nV74ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY
6vDU9/4v9dDiQfEPlauw\n0qNRjPS1MLzjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2
oSobPc\n3mV2zkvtD61CXzbezAVdArDl6WnysRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv
0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3Fy2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4di
KU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh
6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjMLU2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57
IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp5USme4PmNhY6QfUlgjsFjPfoVG7X
DB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C8xstRSEG7oJc\nN5+4ImYnFN
xSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyyD1ld89OUEYZ+aJQd/A==
\n-----END CERTIFICATE-----\n"
```


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 32.

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 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "8235074425707e306abf39",
      "meta": {
        "href": "/api/configuration/policies/user_
databases/8235074425707e306abf39"
```

```
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/policies/audit_policies",
    "href": "/api/configuration/policies/user_databases",
    "last": "/api/configuration/policies/usermapping_policies",
    "next": "/api/configuration/policies/userlists",
    "parent": "/api/configuration/policies",
    "previous": "/api/configuration/policies/trusted_ca_lists",
    "transaction": "/api/transaction"
  }
}
```


When retrieving the endpoint of a specific local user database, the response is the following.


```
{
  "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,
```

```
    "parent": "/api/configuration/policies/user_databases",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

Element	Type	Description
key	string	Top level element, contains the ID of the local user database.
body	Top level element (string)	Contains the properties of the local user database.
name	string	The name of the local user database. This name is also displayed on the SPS web interface. It cannot contain whitespace.
users	Top level list	Contains the credentials (password, key) of each configured user.
passwords	Top level item	References the password of the user. 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).
public_keys	Top level list	Contains the pubic keys of the user.
username	Top level list, string	Name of the user.

Elements of `public_keys`	Type	Description
selection	string	Possible values are: • `rsa` The `value` element contains an RSA key. • `dss` The `value` element contains a DSS key.
value	string	The public key.

Examples:

Configure password authentication only for test_user. (New passwords can only be provided using the web interface of SPS.)

```
{
    "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.

```
{
    "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 32.

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:

```
{
    "key":  "c4e60325-971a-44bc-ac01-e353dc6320d6",
    "meta":  {
       "href":  "/api/configuration/policies/user_databases/c4e60325-971a-44bc-ac01-e353dc6320d6",
       "parent":  "/api/configuration/policies/user_databases",
       "transaction":  "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

Modify a local user database

To modify a local usre database, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 366.

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

Cookie name	Description	Required	Values
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`**.

Cookie name	Description	Required	Values
			For more information 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.

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

The following command retrieves the properties of a specific list.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/policies/userlists/<key-id>
```

Response

The following is a sample response received when retrieveing the user lists.

For more information on the `meta` object, see Message format on page 10.

The keys with negative ID values are the default user lists of SPS.

```
{
  "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"
      }
    },
```

```
    {
      "key": "-2",
      "meta": {
        "href": "/api/configuration/policies/userlists/-2"
      }
    },
    {
      "key": "-3",
      "meta": {
        "href": "/api/configuration/policies/userlists/-3"
      }
    },
    {
      "key": "-4",
      "meta": {
        "href": "/api/configuration/policies/userlists/-4"
      }
    },
    {
      "key": "20088200245706af301b1ba",
      "meta": {
        "href": "/api/configuration/policies/userlists/20088200245706af301b1ba"
      }
    }
  ]
}
```


When retrieving the endpoint of a specific user list, the response is the following.


```
{
  "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",
```

```
    "parent": "/api/configuration/policies/userlists",
    "previous": "/api/configuration/policies/userlists/-3",
    "transaction": "/api/transaction"
  }
}
```

Element		Type	Description
key		string	Top level element, contains the ID of the user list
body		Top level element (string)	The elements of the user policy.
	allow	string	The default policy of the user list. Possible values are: • all_users creates a blacklist, where every user is permitted, except the ones listed in the except field. • no_user creates a whitelist, where only the users listed in the except field are allowed access.
	name	string	The name of the user list.
	except	list	The usernames added to the list.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Examples

The following defines a blacklist called `no_root` that permits every username except root.

```
{
        "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.

```
{
        "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 32.

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:

   ```
   {
     "key": "321314dc-eca0-4e97-b445-0612fedc0165",
     "meta": {
       "href": "/api/configuration/policies/userlists/321314dc-eca0-4e97-
   ```

```
b445-0612fedc0165",
    "parent": "/api/configuration/policies/userlists",
    "transaction": "/api/transaction"
  }
}
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

7

HTTP connections

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

Cookie name	Description	Required	Values
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 more information 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 available settings for configuring for HTTP connections.

```
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 more information on 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"
    }
}
```

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 the HTTP connection policies.
`options`	List of global HTTP 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 HTTP connection policies.

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

The following command retrieves the properties of a specific policy.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
  "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"
}
```

```
      },
      "settings": {
        "key": "-3040010",
        "meta": { "href": "/api/configuration/http/settings_policies/-3040010" }
      },
      "audit_policy": {
        "key": "78101850949e47437dd91d",
        "meta": { "href": "/api/configuration/policies/audit_
policies/78101850949e47437dd91d" }
      },
      "ldap_server": null,
      "backup_policy": null,
      "authentication_policy": {
        "key": "-304002001",
        "meta": { "href": "/api/configuration/http/authentication_policies/-
304002001" }
      },
      "usermapping_policy": null,
      "archive_cleanup_policy": null,
      "analytics_policy": null
    }
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the connection policy.
body	Top level element (string)	Contains the properties of the connection policy.
name	string	The name of the connection policy
active	boolean	Set to `false` to suspend the connection policy. Connection settings are preserved.
network	Top level element	
clients	list, string	List of client ("from") IP addresses.
ports	list, integers	List of target ports.
targets	list,	List of target IP addresses.

Element		Type	Description
		string	
server_address		Top level item	Defines the address where the clients connect to.
source_address		Top level element	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.
	selection	string	Configures Source Network Address Translation. Possible values are: • box_address Default. Uses the network address of the logical interface of SPS. • original Uses the IP address of the client, as seen by SPS. • fix Uses a fixed address when connecting to the remote server. Must be used with the address element.
	address	string	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.
web_proxy		Top level element	This will allow the clients to use SPS as an HTTP web proxy.
	enabled	boolean	When set to true This will allow the clients to use SPS as an HTTP web proxy.
	transport_security	Top level element	Configures the transport security (TLS) of the web proxy connection, between the client and SPS. Note that this setting requires a compatible client application

Element		Type	Description
			that is capable of using TLS-secured web proxy connections.
transport_security		Top level element	Configures the end-to-end encryption used in the sessions.
access_control		Top level list	Collection of access policies. Access policies define who can authorize and audit a connection.
indexing		Top level item	Configures indexing for the connection policy.
	enabled	boolean	Set to `true` to enable indexing the connections.
	policy	string	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).
	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. • `1` Very high priority. • `0` Near real-time priority.

Element		Type	Description
`rate_limit`		Top level element	Connection rate limit.
	`enabled`	boolean	Set to true to provide a connection rate limit.
	`value`	int	The number of connections (per minute) that are allowed in the connection policy.
`log_audit_trail_downloads`		boolean	Set to `true` to log audit trail downloads.
`channel_database_cleanup`		Top level item	Configures cleanup of the connection metadata on the connection policy's level.
	`days`	int	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 HTTP protocol are available at the `api/configuration/http/options` endpoint.
	`enabled`	boolean	Set to true to enable periodical cleanup of the connection metadata.
`override_log_level`		Top level item	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/<protocol>/options` endpoint is used. • 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

Element		Type	Description

			option and set the log level to use:
			`"override_log_level": {`
			`  "enabled": true,`
			`  "log_level": 5`
			`},`

`policies`		Top level item	List of policies referenced by the connection policy.
	`channel_policy`	string	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 HTTP channel policies at the `/api/configuration/http/channel_policies/` endpoint.
	`settings`	string	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 HTTP settings policies at the `/api/configuration/http/settings_policies/` endpoint.
	`audit_policy`	string	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

Element		Type	Description
			value of the returned `key` as the value of the `audit_policy` element, and remove any child elements (including the key).
	`ldap_server`	string	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).
	`backup_policy`	string	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).
	`authentication_policy`	string	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).
	`usermapping_policy`	string	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

Element		Type	Description
			value of the `usermapping_policies` element, and remove any child elements (including the key).
	`archive_cleanup_policy`	string	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).
	`analytics_policy`	string	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).

Elements of `server_address`	Type	Description

custom_dns	string	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.
		
		• To use the default DNS, disable this option:
		
		`"server_address": {`
		`  "custom_dns": {`
		`  "enabled": false`
		`  },`
		`  ...`

Elements of `server_address`	Type	Description

		`},`
		• 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"`
		`},`
		`...`
		`},`

`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

Elements of `server_address`	Type	Description
		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 provided order, and uses the first successfully resolved address to establish the connection.
`domains`	Top level list	Must be used if `selection` is set to `inband`.
`domain`	Top level item	Lists the address ranges that are included in the connection policy.
`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 targer server(s).
`exception_domains`	Top	Can only be used if `selection` is set to `inband`.

Elements of `server_address`	Type	Description
	level list	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` 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.

Elements of `web_proxy.transport_security`	Type	Description

`selection`	string	Configures the encryption used in the sessions.
		• `disabled`: Use unencrypted web proxy connection between the HTTP client and .
		`"transport_security": { "selection": "disabled" },`
		• `tls`: Enables TLS-encryption.
		`"transport_security": { "selection": "tls" }`

Elements of `web_proxy.transport_security`	Type	Description
`host_certification_method`	JSON object	Selects the certificate to show to the peers. You have the following options: • **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 300. `"host_certification_method": { "selection": "fix", "x509_identity": "893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a" }` • **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 418.

Elements of `web_proxy.transport_security`	Type	Description
		`"host_certification_method": { "selection": "generate", "signing_ca": "1904188625a843f11d30a-5" },`
`selection`	string	Possible values: • `fix`: if you want to use the same certificate for every peer. • `generate`: if you want to generate a certificate for the target requested by the client.
`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 300.
`signing_ca`	string	Reference the Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 418.

Elements of `body.transport_security`	Type	Description
`selection`	string	Configures the encryption used in the sessions. • `disabled`: Use unencrypted connection between the HTTP client and server. `"transport_security": { "selection": "disabled" },` • `client-only`: Enables half-

Elements of `body.transport_security`	Type	Description

		sided TLS encryption. Require HTTPS on client side, and HTTP on server side.
		`"transport_security": {`
		`"selection": "client-only"`
		`}`
		• `client-server`: Enables end-to-end TLS-encryption. To allow unencrypted HTTP requests in addition to HTTPS requests, set `allow_non_encrypted` to `true`.
		`"transport_security": {`
		`"selection": "client-server",`
		`"allow_non_encrypted": true`
		`"server_certificate_check": {}`
		`}`

`allow_non_encrypted`	boolean	Only if `selection` is set to `client_server`. To allow unencrypted HTTP requests in addition to HTTPS requests, set `allow_non_encrypted` to `true`.

`server_certificate_check`	Top level item	By default, SPS accepts any certificate shown by the server.
		`"server_certificate_check": {`
		`"enabled": false`
		`},`
		To verify the certificate of the destination server, configure and reference a Trusted CA list.
		`"server_certificate_check": {`

Elements of `body.transport_security`		Type	Description
			`"enabled": true, "trusted_ca": "9106862955a844051d7bf6" },`
	`enabled`	boolean	To verify the certificate of the destination server, set to `true`. In this case, you will also have to reference a `trusted_ca`.
	`trusted_ca`	string	Reference a Trusted CA list.
`host_certification_method`		JSON object	Selects the certificate to show to the peers. You have the following options: • **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 300. `"host_certification_method": { "selection": "fix", "x509_identity": "893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a" }`

Elements of `body.transport_security`	Type	Description
		• **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 418. `"host_certification_method": { "selection": "generate", "signing_ca": "1904188625a843f11d30a-5" },`
`selection`	string	Possible values: • `fix`: if you want to use the same certificate for every peer. • `generate`: if you want to generate a certificate for the target requested by the client.
`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 300.
`signing_ca`	string	Reference the Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 418.

Elements of `access_control`	Type	Description
`authorizer`	string	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.
`permission`	string	Defines the permissions of the `authorizer` usergroup. Possible values are: • `audit` The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection. • `authorize` The usergroup with the authorize permission can authorize connection requests. • `audit_and_authorize` The usergroup with the audit_and_authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.
`require_different_ip`	boolean	Set to `true` to require the authorizing user and its subject to have different IP addresses.
`require_different_username`	boolean	Set to `true` to require the authorizing user and its subject to have different usernames.
`subject`	Top level item	Defines the subjects of the access control policy.
`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`

Elements of `access_control`	Type	Description
		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.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
404	NotFound	The requested object does not exist.

HTTP channels

The available HTTP channel types and their functionalities are described below. For details on configuring channel policies, see Channel policy.

Channel	Special options	Description
`http`	No	**http**: Enables access to the server. This channel must be enabled for HTTP connections to work.
`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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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/http/authentication_policies/-
304002001"  }

          }
       }
    ],
    "meta":  {
       "first":  "/api/configuration/http/authentication_policies",
       "href":  "/api/configuration/http/authentication_policies",
       "last":  "/api/configuration/http/settings_policies",
       "next":  "/api/configuration/http/channel_policies",
       "parent":  "/api/configuration/http",
       "previous":  null,
       "transaction":  "/api/transaction"
    }
 }
```


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
    }
  }
```

```
}
```

Element	Type	Description
key	string	Top level element, contains the ID of the policy.
body	Top level element	Contains the elements of the policy.
name	string	The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.
gateway_authentication	Top level item	Client-side gateway authentication settings. The value of `selection` defines which authentication method is used.
selection	string	Defines the authentication method for client-side gateway authentication. Possible values are: • `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

Element		Type	Description
			also configure the `user_database` element. • `radius` Uses one or more Radius servers for authentication. To use this option, you must also configure the `authentication_protocol` and `servers` elements.
	`servers`	Top level list	Only if `selection` is set to `radius` 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.
	`authentication_protocol`	Top level item	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`.
	`user_database`	string	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).
	`timeout`	integer (seconds)	Specify the time remaining until a successful gateway authentication times out.
	`keepalive`	boolean	Set to `true` to avoid interruptions for active HTTP sessions. Active HTTP sessions can extend the gateway authentication beyond the configured timeout.

Elements of `servers`		Type	Description
`address`		Top level element	Defines the address of a RADIUS server.
	`selection`	string	Required child of the address element. Possible values are: • `ip` The `value` element contains the IP of the RADIUS server. • `fqdn` The `value` element contains the FQDN of the RADIUS server.
	`value`	string	The IP or the FQDN address of the RADIUS server.
`port`		int	The port number of the RADIUS server.
`shared_secret`		string	References the key of the shared secret for the RADIUS server. You can configure shared secrets at the `/api/configuration/passwords/` 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. `"shared_secret": { "plain": "<new-password>" }`

Examples:

Querying base authentication policy without gateway authentication:

```
{
    "key":  "-304002001",
    "body":  {
       "name":  "base",
       "gateway_authentication":  {
          "selection":  "none"
       }
    }
 }
```

Querying authentication policy with LDAP backend:

```
{
    "key":  "http-auth-pol-2",
    "body":  {
       "name":  "http_ldap",
       "gateway_authentication":  {
          "selection":  "ldap",
          "timeout":  3600,
          "keepalive":  true
       }
    }
 }
```


Querying authentication policy with local backend:


```
{
    "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:


```
{
  "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
    }
  }
}
```


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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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:

```
{
    "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 35.

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 32.

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 35.

Global HTTP options

List of options that affect all HTTP connections.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 more information on the `meta` object, see Message format on page 10.

```
{
  "body": {
    "channel_database_cleanup": {
      "enabled": false
    },
    "service": {
      "enabled": false
    }
  }
  "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"
  }
}
```

Element		Type	Description
key		Top level item	Contains the ID of the endpoint.
body		Top level item	Contains the elements of the global HTTP options.
	channel_database_cleanup	Top level item	Contains settings for database cleanup.
	service	Top level item	Global setting to enable HTTP connections, and specify the logging detail.

Elements of `channel_database_cleanup`	Type	Description
days	integer	Applies 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 connection-specific database cleanup times (if configured).
enabled	boolean	To enable the global cleanup of HTTP connection metadata, set `enabled` to **`true`**.

Elements of `service`	Type	Description
log_level	integer	Applies only if `enabled` is set to `true`. Defines the logging detail of HTTP connections.
enabled	boolean	Set to **`true`** to enable HTTP connections.

Examples

Querying the full list of global HTTP options:

```
{
    "body": {
       "channel_database_cleanup": {
          "enabled": true,
          "days": 365
       },
       "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"
    }
 }
```

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 32.

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.

   For more information about the elements of the `channel_database_cleanup` item, see Elements of `channel_database_cleanup`.

   For more information about the elements of the `service` item, see Elements of `service`.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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 the SPS REST API attempted to access, but could not retrieve.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section contains the `path` that the SPS REST API attempted to access, but could not retrieve.
404	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information on authentication, see

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).

Sample request

The following command lists HTTP settings policies.

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

The following command retrieves the properties of a specific policy.

```
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 more information on the `meta` object, see Message format on page 10.


```
{
    "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,
       "parent": "/api/configuration/http",
       "previous": "/api/configuration/http/options",
       "transaction": "/api/transaction"
    }
 }
```

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
       "webapp_session_cookies": [
          "PHPSESSID",
          "JSESSIONID",
          "ASP.NET_SessionId"
       ]
    },
    "key": "-3040010",
    "meta": {
       "first": "/api/configuration/http/settings_policies/-3040010",
       "href": "/api/configuration/http/settings_policies/-3040010",
       "last": "/api/configuration/http/settings_policies/-3040010",
       "next": null,
       "parent": "/api/configuration/http/settings_policies",
       "previous": null,
       "transaction": "/api/transaction"
    }
 }
```

When retrieving the default settings policy with a built-in HTTP proxy error template, the response is the following.

```
{
     "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"
      },
      "enable_disclaimer": false,
      "preconnect_channel_check": false
    }
  }
```

When you create a new settings policy with a custom error template, the response is the following.

```
{
            "name":  "custom_http_settings_policy",
            "timeout":  400,
            "session_timeout":  1000,
            "webapp_session_cookies":  [
               "PHPSESSID",
               "JSESSIONID",
               "ASP.NET_SessionId"
            ],
            "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":  "custom",
               "reference":  "123456789"
            }
         }
```

Element		Type	Description
key		string	Top level element, contains the ID of the policy.
body		Top level element (string)	The elements of the HTTP settings policy.
	client_tls_security_settings	JSON object	Configures TLS security settings on the client side.
	name	string	Name of the HTTP settings policy. Cannot contain whitespace.
	server_tls_security_settings	JSON object	Configures TLS security settings on the server side.
	session_timeout	int	Session timeout, in seconds.
	timeout	int	Idle timeout, in seconds. Note that the SPS web UI displays the same value in seconds.
	webapp_session_cookies	list (string)	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, JSESSIONID, 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.
	error_template	object	
	error_template.selection	enum	The type of the error template. Possible values: builtin, custom
	error_template.reference	number	The identifier of the error template.
	enable_disclaimer	boolean	
	preconnect_channel_check	boolean	

Elements of client_tls_security_settings and server_tls_security_settings		Type	Description
cipher_strength		JSON object	Specifies the cipher string OpenSSL will use.
	custom_cipher	string	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.
	selection	string	Specifies the cipher string OpenSSL will use. The following settings options are possible: • recommended: this setting only uses ciphers with adequate security level. • 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. For example: ALL:!aNULL:@STRENGTH
minimum_tls_version		string	Specifies the minimal TLS version SPS will offer during negotiation. The following settings options are possible: • TLSv1_2: this setting only offers TLS version 1.2 during the negotiation. This is the recommended setting. • TLSv1_1: this setting offers TLS version 1.1 and later versions during the negotiation. • TLSv1_0: this setting offers TLS version 1.0 and later versions during the negotiation.

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 32.

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:

```
{
    "key":  "3848c708-2e1d-4463-b232-0c8c5875ff55",
    "meta":  {
       "href":  "/api/configuration/http/settings_policies/3848c708-2e1d-
4463-b232-0c8c5875ff55",
       "parent":  "/api/configuration/http/settings_policies",
       "transaction":  "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Creating custom HTTP error templates

Configure HTTP error templates to create custom error pages when HTTP proxy errors occur.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 `/http/error_templates` endpoint include:

Operation	HTTP method	URL	Notes
Querying existing error templates	GET	`/api/configuration/http/error_templates`	
Creating a new error template with a custom logo	POST	`/api/configuration/http/error_templates`	
Updating an existing error template	PUT	`/api/configuration/http/error_templates/<key>`	
Querying error templates info	GET	`/api/configuration/http/error_templates?info=`	
Querying custom error template preview	GET	`/api/configuration/http/error_templates?preview=&error_type=<type-of-the-error-template>`	

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 more information on the `meta` object, see Message format on page 10.

```
{
     "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.",
```

```
      "internal": "Custom internal error markdown content.",
      "invalidurl": "Custom invalid URL markdown content.",
      "ioerror": "Custom I/O error markdown content.",
      "policysyntax": "Custom policy syntax error markdown content.",
      "policyviolation": "Custom policy violation error markdown content.",
      "redirect": "Custom redirect markdown content.",
      "serversyntax": "Custom server syntax error markdown content.",
      "servertimeout": "Custom server timeout 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=`.

```
{
        "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:

Element	Type	Description	Notes
`name`	string	The name of the template.	
`color`	number	The color of the brand and links appearing in the error template.	The value is given in hex color code.
`logo`	string? union?	The identifier of the logo.	The value of the logo can be 'null'.
`brand_name`	string	The name of the brand.	

Element	Type	Description	Notes
`md_contents`	object	Contains a list of error templates written in Markdown.	
`md_contents.auth`	string	The content of the **Authentication failed** error template in Markdown.	
`md_contents.badcontent`	string	The content of the **Bad content** error template in Markdown.	
`md_contents.clientsyntax`	string	The content of the **Client syntax** error template in Markdown.	
`md_contents.clienttimeout`	string	The content of the **Client timeout** error template in Markdown.	
`md_contents.connecterror`	string	The content of the **Connect error** error template in Markdown.	
`md_contents.ftperror`	string	The content of the **FTP error** error template in Markdown.	
`md_contents.internal`	string	The content of the **Internal error** error template in Markdown.	
`md_contents.invalidurl`	string	The content of the **Invalid URL** error template in Markdown.	
`md_contents.ioerror`	string	The content of the **IO error** error template in Markdown.	
`md_contents.policysyntax`	string	The content of the **Policy syntax** error template in Markdown.	
`md_contents.policyviolation`	string	The content of the **Policy violation** error template in Markdown.	
`md_contents.redirect`	string	The content of the **Redirect error** template in Markdown.	
`md_contents.serversyntax`	string	The content of the **Server syntax** error template in Markdown.	

Element	Type	Description	Notes
`md_contents.servertimeout`	string	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.

HTTP response code	Status/Error	Description
400	NotSupportedProxyErrorType	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**.

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

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

Cookie name	Description	Required	Values
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 more information 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 `/http/proxy_error_logo` endpoint include:

Operation	HTTP method	URL	Notes
Uploading a custom logo	POST	`/api/upload/http/proxy_error_logo`	If you upload an oversized logo, for example 800x800 pixels, your image will be automatically resized to 128x128 pixels, keeping the original aspect ratio.

NOTE: GET / PUT / DELETE methods are not allowed on logo upload.

Sample request

The following command uploads a custom logo.

```
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.

```
{
        "logo_id":  "rest_XXXXXXXX",
        "meta":  {
           "href":  "/api/upload/http/proxy_error_logo",
           "parent":  "/api/upload/http"
        }
     }
```

The response of the query without a unique logo ID is the following.

```
{
        "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 more information on the `meta` object, see Message format on page 10.

Elements of the response message include:

Element	Type	Description	Notes
`logo_id`	string	The identifier of the custom logo.	
`supported_formats`	enum	The supported image formats.	Possible values: PNG, JPEG, GIF.
`actual_resolution`	number	The actual resolution of the uploaded custom logo.	
`max_resolution`	number	The maximal resolution of the uploaded custom logo.	The maximum accepted size is 4096x4096 pixels.
`file_size`	number	The actual size of the file.	
`file_limit`	number	The maximum size of the file.	The file size limit is 16 megabyte.

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.

HTTP response code	Status/Error	Description
400	ResolutionTooLarge	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.
413	FileTooLarge	The file in the request exceeds the file size limitation.
415	InvalidImageFormat	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).

For more information and a complete list of standard HTTP response codes, see Application level error codes on page 41.

8

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

Cookie name	Description	Required	Values
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 more information 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 available settings for configuring for ICA connections.

```
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 more information on the `meta` object, see Message format on page 10.


```
{
    "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"
    }
 }
```


Item	Description
`channel_policies`	List of the default and custom channel policies.

Item	Description
`connections`	List of ICA connection policies.
`options`	List of global ICA options that affect all connections.
`settings_policies`	List of protocol-level settings (timeout, reliability). 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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.

Channel	Special options	Description
CTXTW	Yes	**Drawing (Thinwire)**: Enables access to the server's desktop

Channel	Special options	Description
		(screen). 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: `"actions": { "audit": true, "four_eyes": true, "content_policy": { "key": "433849548566ab327522e6" "meta": { "href": "/api/configuration/policies/content_policies/44287216854f482e7f2b24" } }, }`
`CTXCAM`	None	**Audio Mapping**: Enable access to the sound device of the server.
`CTXCDM`	None	**Drive Mapping**: Enable access to the client's hard drives on the server.
`CTXCLIP`	None	**Clipboard**: 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.
`CTXSCRD`	None	**Smartcard**: Enable using client side installed smartcards in server-side applications.
`CTXCOM1`	None	**Printer (COM1)**: Enable access to the serial port COM1.
`CTXCOM2`	None	**Printer (COM2)**: Enable access to the serial port COM2.
`CTXLPT1`	None	**Printer (LPT1)**: Enable access to the parallel port LPT1.
`CTXLPT2`	None	**Printer (LPT2)**: Enable access to the parallel port LPT2.
`CTXCPM`	None	**Printer Spooler**: Enable access to the client's printer from the remote desktops and applications.

Channel	Special options	Description
`CTXFLSH`	None	**HDX Mediastream**: 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.
`CTXUSB`	None	**USB**: Enable using client side installed USB devices in server-side applications.
`CTXTWI`	None	**Seamless**: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.
`SPDBRS`	None	**Speedbrowse**: Speeds up web browsing. Not currently supported by Safeguard Desktop Player, should be disabled by default.
`custom`	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, configure the `channels` field. Channel-specific access control rules: • `channels`: To permit only specific custom channels, configure this field. For example:. `"channels": { "selection": "restricted", "restrictions": [ "CUSTOM1", "CUSTOM2" ] }`

Global ICA options

List of options that affect all ICA connections.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 ICA options.

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

Response

The following is a sample response received when listing the list of default global ICA options.

For more information on the `meta` object, see Message format on page 10.


```
{
  "body": {
      "channel_database_cleanup": {
        "enabled": false
      },
      "service": {
        "enabled": false
      }
    }
  }
  "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"
  }
}
```

Element		Type	Description
key		Top level item	Contains the ID of the endpoint.
body		Top level item	Contains the elements of the global ICA options.
	channel_database_cleanup	Top level item	Contains settings for database cleanup.
	service	Top level item	Global setting to enable ICA connections, and specify the logging detail.

Elements of `channel_database_cleanup`	Type	Description
days	integer	Applies only if `enabled` is set to `true`. 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).
enabled	boolean	To enable the global cleanup of ICA connection metadata, set `enabled` to **`true`**.

Elements of `service`	Type	Description
log_level	integer	Applies only if `enabled` is set to `true`. Defines the logging detail of ICA connections.
enabled	boolean	To enable ICA connections, set to **`true`**.

Examples

Querying the full list of global ICA options:

```
{
    "body": {
       "channel_database_cleanup": {
          "enabled": true,
          "days": 365
        },
       "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"
    }
 }
```

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 32.

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.

   For more information about the elements of the `channel_database_cleanup` item, see Elements of `channel_database_cleanup`.

   For more information about the elements of the `service` item, see Elements of `service`.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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 the SPS REST API attempted to access, but could not retrieve.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section contains the `path` that the SPS REST API attempted to access, but could not retrieve.
404	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information on authentication, see

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).

Sample request

The following command lists ICA settings policies.

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

The following command retrieves the properties of a specific policy.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
    "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.

```
{
    "body": {
       "name": "default",
       "timeout": 600,
       "inactivity_timeout": {
          "enabled": true
          "value": 13000
       },
       "preconnect_channel_check": false,
       "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"
    }
 }
```

Element	Type	Description
key	string	Top level element, contains the ID of the policy.
body	Top level element (string)	The elements of the ICA settings policy.
name	string	Name of the ICA settings policy. Cannot contain whitespace.
preconnect_channel_check	boolean	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

Element		Type	Description
			access the server. To enable this function, set the parameter to `true`.
`reliability`		Top level item	Settings for ICA connection attempts.
`timeout`		int	Connection timeout, in seconds.
`inactivity_timeout`		Top level element	
	`enabled`	boolean	• `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`

Elements of `reliability`	Type	Description
`reconnect_attempts`	int	The number of times SPS attempts to connect to the target server.
`reconnect_sleep`	int	The number of seconds SPS waits between connection attempts.
`reconnect_timeout`	int	The number of seconds SPS waits after exhausting the number of `reconnect_attempts`.

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 32.

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:

```
{
    "key":  "dcd58077-98b3-4c73-8f0b-b34147863028",
    "meta":  {
       "href":  "/api/configuration/ica/settings_policies/dcd58077-98b3-
4c73-8f0b-b34147863028",
       "parent":  "/api/configuration/ica/settings_policies",
       "transaction":  "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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`

Code	Description	Notes
		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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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": {
```

```
        "href": "/api/configuration/mssql/connections"
      }
    },
    {
      "key": "options",
      "meta": {
        "href": "/api/configuration/mssql/options"
      }
    },
    {
      "key": "settings_policies",
      "meta": {
        "href": "/api/configuration/mssql/settings_policies"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/aaa",
    "href": "/api/configuration/mssql",
    "last": "/api/configuration/x509",
    "next": "/api/configuration/network",
    "parent": "/api/configuration",
    "previous": "/api/configuration/management",
    "remaining_seconds": 600,
    "transaction": "/api/transaction"
  }
}
```


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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
```

```
        }
      },
      "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"
  }
}
```

Element			Type	Description
key			string	Top level element, contains the ID of the connection policy.
body			Top level element (string)	The elements of the connection policy.
	access_control		Top level list	Collection of access policies. Access policies define who can authorize and audit a connection.
	active		boolean	Set to `false` to suspend the connection policy. Connection settings are preserved.
	channel_database_cleanup		Top level item	Configures cleanup of the connection metadata on the connection policy's level.
		days	int	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.
		enabled	boolean	Set to true to enable periodical cleanup of the connection metadata.
	indexing		Top level item	Configures indexing for the connection policy.
		enabled	boolean	Set to `true` to enable indexing the connections.

Element			Type	Description
		policy	string	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).
		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. • 1 Very high priority. • 0 Near real-time priority.
	log_audit_trail_downloads		boolean	Set to true to log audit trail downloads.
	name		string	The name of the connection policy.
	network			
		clients	list, string	List of client ("from") IP addresses.
		ports	list, integer-s	List of target ports.
		targets	list, string	List of target IP addresses.
	override_log_level		Top level	Specifies the verbosity level of sessions

Element		Type	Description

		item	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/<protocol>/options` endpoint is used.
			• 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 option and set the log level to use:
			`"override_log_level": {`
			`"enabled": true,`
			`"log_level": 5`
			`},`

`policies`		Top level item	List of policies referenced by the connection policy.
	`aa_plugin`	string	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).
	`analytics_policy`	string	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).
	`archive_`	string	References the identifier of the

Element		Type	Description
	`cleanup_policy`		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).
	`audit_policy`	string	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).
	`authentication_policy`	string	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. 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).
	`backup_policy`	string	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).
	`channel_policy`	string	References the identifier of the channel policy. The value of this option cannot be `null`.

Element		Type	Description
			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 MSSQL channel policies at the `/api/configuration/mssql/channel_policies/` endpoint.
	`credential_store`	string	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).
	`ldap_server`	string	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).
	`settings`	string	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).
	`usermapping_policy`	string	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,

Element		Type	Description
			use the value of the returned `key` as the value of the `usermapping_policies` element, and remove any child elements (including the key).
`rate_limit`		Top level element	Connection rate limit.
	`enabled`	boolean	Set to true to provide a connection rate limit.
	`value`	int	The number of connections (per minute) that are allowed in the connection policy.
`server_address`		Top level item	Defines the address where the clients connect to.
`source_address`		Top level element	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.
	`selection`	string	Configures Source Network Address Translation. Possible values are: • `box_address` Default. Uses the network address of the logical interface of SPS. • `original` Uses the IP address of the client, as seen by SPS. • `fix` Uses a fixed address when connecting to the remote server. Must be used with the `address` element.
	`address`	string	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.
`transport_`		Top	Configures the encryption used in the

Element		Type	Description
`security`		level element	sessions.

	`certificate`	JSON object	Selects the certificate to show to the peers. You have the following options:
			
			• **Use the same certificate for each client:**
			
			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 300.
			
			`"certificate": { "selection": "fix", "x509_identity": "893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a" }`
			
			• **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 418.
			
			`"certificate": { "selection": "generate", "signing_ca": "1904188625a843f11d30a5" },`

	`selection`	disabled | tls	Configures the encryption used in the sessions.

Element		Type	Description

			• `disabled`: Disables TLS encryption for MSSQL connections completely.
			`"transport_security": {`
			`"selection": "disabled"`
			`},`
			• `tls`: Enables TLS-encryption. Note that you must also set the `certificate` and `server_certificate_check` options.
			`"transport_security": {`
			`"certificate": {`
			`"selection": "generate",`
			`"signing_ca": "19605948865d07511f09eca"`
			`},`
			`"selection": "tls",`
			`"server_certificate_check": {`
			`"enabled": true,`
			`"trusted_ca": "1241814345d074efd1ded7"`
			`}`
			`}`

	`server_certificate_check`	JSON object	By default, SPS accepts any certificate shown by the server.
			`"server_certificate_check": {`
			`"enabled": false`
			`},`
			To verify the certificate of the destination server, configure and reference a Trusted CA list.
			`"server_certificate_check": {`
			`"enabled": true,`
			`"trusted_ca": "9106862955a844051d7bf6"`
			`},`

`web_gateway_`		Top	When gateway authentication is required

Element		Type	Description
authentication		level item	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	Type	Description
authorizer	string	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.
permission	string	Defines the permissions of the authorizer usergroup. Possible values are: • audit The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection. • authorize The usergroup with the authorize permission can authorize connection requests. • audit_and_authorize The usergroup with the audit_and_

Elements of `access_control`	Type	Description
		authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.
`require_different_ip`	boolean	Set to `true` to require the authorizing user and its subject to have different IP addresses.
`require_different_username`	boolean	Set to `true` to require the authorizing user and its subject to have different usernames.
`subject`	Top level item	Defines the subjects of the access control policy.
`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

Elements of `server_address`	Type	Description
custom_dns	string	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

Elements of `server_address`	Type	Description

		`/api/configuration/network/dns` endpoint.
		• 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" }, ... },`

`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`

Elements of `server_address`	Type	Description
		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.
`port`	int	Must be used if `selection` is set to `fix`. The port of the target server.
`domains`	Top level list	Must be used if `selection` is set to `inband`.
`domain`	Top level item	Lists the address ranges that are included in the connection policy.
`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 targer 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	Contains the excluded address range.

Elements of `server_address`	Type	Description
	item	
`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 provided order, and uses the first successfully resolved address to establish the connection.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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:

   ```
   {
       "key": "a99be49b-b0a2-4cf9-b70d-fea1f9ea188f",
       "meta": {
          "href": "/api/configuration/mssql/connections/a99be49b-b0a2-4cf9-b70d-fea1f9ea188f",
          "parent": "/api/configuration/mssql/connections",
          "transaction": "/api/transaction"
       }
    }
   ```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

MSSQL channels

The available MSSQL channel types and their functionalities are described below. For details on configuring channel policies, see Channel policy.

Channel	Special options	Description

`mssql`	Yes	**mssql**: Enables access to the MSSQL server. This channel must be enabled for MSSQL to work.
		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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
    "items": [
       {
          "key": "-200",
          "meta": {
             "href": "/api/configuration/mssql/authentication_policies/-200"
          }
       },
       {
          "key": "-304002001",
          "meta":   {
             "href": "/api/configuration/mssql/authentication_policies/-
304002001"
          }
       }
    ],
    "meta": {
       "first": "/api/configuration/mssql/authentication_policies",
       "href": "/api/configuration/mssql/authentication_policies",
       "last": "/api/configuration/mssql/settings_policies",
       "next": "/api/configuration/mssql/channel_policies",
       "parent": "/api/configuration/mssql",
       "previous": null,
       "transaction": "/api/transaction"
    }
 }
```


When retrieving the endpoint of a specific policy, the response is the following.


```
{
    "body": {
       "backend": {
          "selection": "ldap"
       },
       "name": "mssql_auth_policy_with_ldap"
    }
 }
```


Element	Type	Description
key	string	Top level element, contains the ID of the policy.
body	Top level element	Contains the elements of the policy.
name	string	The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.

Element		Type	Description
backend		Top level item	Client-side gateway authentication settings. The value of `selection` defines which authentication method is used.
	selection	string	Defines the authentication method for client-side gateway authentication. Possible values are: • `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). To use this option, you must also configure the `certificate`, `password`, and `public_key` elements. • `local` Uses the local user database configured in the `/api/configuration/policies/user_databases/` endpoint. To use this option, you must also configure the `user_database` element. • `radius` Uses one or more Radius servers for authentication. To use this option, you must also configure the `authentication_protocol` and `servers` elements.
	servers	Top level list	Only if `selection` is set to `radius` 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.

Element		Type	Description
	`authentication_protocol`	Top level item	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`.
	`user_database`	string	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).
	`timeout`	integer (seconds)	Specify the time remaining until a successful gateway authentication times out.
	`keepalive`	boolean	Set to `true` to avoid interruptions for active HTTP sessions. Active HTTP sessions can extend the gateway authentication beyond the configured timeout.

Elements of `servers`		Type	Description
`address`		Top level element	Defines the address of a RADIUS server.
	`selection`	string	Required child of the address element. Possible values are: • `ip` The `value` element contains the IP of the RADIUS server. • `fqdn` The `value` element contains the FQDN of the RADIUS server.
	`value`	string	The IP or the FQDN address of the RADIUS server.
`port`		int	The port number of the RADIUS server.
`shared_`		string	References the key of the shared secret for the

Elements of `servers`	Type	Description
`secret`		RADIUS server. You can configure shared secrets at the `/api/configuration/passwords/` 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. `"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":  "mssql-auth-pol-2",
    "body":  {
       "name":  "mssql_ldap",
       "backend":  {
          "selection":  "ldap",
          "timeout":  3600,
          "keepalive":  true
       }
    }
 }
```

Querying authentication policy with local backend:

```
{
    "key": "mssql-auth-pol-3",
    "body": {
       "name": "mssql_local",
       "backend": {
          "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:


```
{
    "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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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:

```
{
  "key": "6f924f39-e4c9-4b0f-8018-8842e2115ebd",
  "meta": {
    "href": "/api/configuration/mssql/authentication_policies/6f924f39-
```

```
e4c9-4b0f-8018-8842e2115ebd",
    "parent": "/api/configuration/mssql/authentication_policies",
    "transaction": "/api/transaction"
  }
}
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

Global MSSQL options

List of options that affect all MSSQL connections.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
  "body": {
    "channel_database_cleanup": {
      "enabled": false
    },
    "service": {
      "enabled": false
    }
  }
  "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"
  }
}
```

Element		Type	Description
key		Top level item	Contains the ID of the endpoint.
body		Top level item	Contains the elements of the global MSSQL options.
	channel_database_cleanup	Top level item	Contains settings for database cleanup.
	service	Top level item	Global setting to enable MSSQL connections, and specify the logging detail.

Elements of `channel_database_cleanup`	Type	Description
days	integer	Applies only if `enabled` is set to `true`. 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).
enabled	boolean	To enable the global cleanup of MSSQL connection metadata, set `enabled` to **`true`**.

Elements of `service`	Type	Description
log_level	integer	Applies only if `enabled` is set to `true`. Defines the logging detail of MSSQL connections.
enabled	boolean	To enable MSSQL connections, set to **`true`**.

Examples

Querying the full list of global MSSQL options:

```
{
    "body": {
       "channel_database_cleanup": {
          "enabled": true,
          "days": 365
       },
       "service": {
          "enabled": true,
          "log_level": 4
          }
    }
    "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"
    }
 }
```

Modify global MSSQL settings

To modify global MSSQL settings,

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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.

   For more information about the elements of the `channel_database_cleanup` item, see Elements of `channel_database_cleanup`.

   For more information about the elements of the `service` item, see Elements of `service`.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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 the SPS REST API attempted to access, but could not retrieve.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section contains the `path` that the SPS REST API attempted to access, but could not retrieve.
404	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information on authentication, see

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).

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 more information on the `meta` object, see Message format on page 10.


```
{
    "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.

```
{
    "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"
       },
       "preconnect_channel_check": false,
       "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"
    }
 }
```

Element		Type	Description
key		string	Top level element, contains the ID of the policy.
body		Top level element (string)	The elements of the MSSQL settings policy.
	client_tls_security_settings	JSON object	Configures TLS security settings on the client side.
	name	string	Name of the MSSQL settings policy. Cannot contain whitespace.
	server_tls_security_	JSON object	Configures TLS security settings on the server side.

Element	Type	Description
settings		
timeout	int	Idle timeout, in seconds. Note that the SPS web UI displays the same value in seconds.

Elements of client_tls_security_settings and server_tls_security_settings	Type	Description
cipher_strength	JSON object	Specifies the cipher string OpenSSL will use.
custom_cipher	string	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.
selection	string	Specifies the cipher string OpenSSL will use. The following settings options are possible: • recommended: this setting only uses ciphers with adequate security level. • 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. For example: ALL:!aNULL:@STRENGTH
minimum_tls_version	string	Specifies the minimal TLS version SPS will offer during negotiation. The following settings options are possible: • TLSv1_2: this setting only offers TLS version 1.2 during the negotiation. This is the recommended setting. • TLSv1_1: this setting offers TLS version 1.1 and later versions during the negotiation. • TLSv1_0: this setting offers TLS version 1.0 and later versions during the negotiation.

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 32.

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:

```
{
    "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 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

RDP connections

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
    "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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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",
```

```
        "href": "/api/configuration/rdp/connections",
        "last": "/api/configuration/rdp/settings_policies",
        "next": "/api/configuration/rdp/domain_membership",
        "parent": "/api/configuration/rdp",
        "previous": "/api/configuration/rdp/channel_policies",
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
    }
}
```


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"
        },
        "source_address": {
            "selection": "box_address"
        },
        "transport_security": {
            "certificate": {
                "selection": "self_signed"
            },
            "server_certificate_check": {
               "enabled": false
            },
            "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"
    }
```

```
}
```

Element			Type	Description
key			string	Top level element, contains the ID of the connection policy.
body			Top level element (string)	The elements of the connection policy.
	access_control		Top level list	Collection of access policies. Access policies define who can authorize and audit a connection.
	active		boolean	Set to `false` to suspend the connection policy. Connection settings are preserved.
	channel_database_cleanup		Top level item	Configures cleanup of the connection metadata on the connection policy's level.
		days	int	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.
		enabled	boolean	Set to true to enable periodical cleanup of the connection metadata.
	indexing		Top level item	Configures indexing for the connection policy.
		enabled	boolean	Set to `true` to enable indexing the connections.
		policy	string	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).

Element			Type	Description
		`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. • `1` Very high priority. • `0` Near real-time priority.
	`log_audit_trail_downloads`		boolean	Set to `true` to log audit trail downloads.
	`name`		string	The name of the connection policy.
	`network`			
		`clients`	list, string	List of client ("from") IP addresses.
		`ports`	list, integer-s	List of target ports.
		`targets`	list, string	List of target IP addresses.
	`override_log_level`		Top level item	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/<protocol>/options` endpoint is used. • To use the default log level, disable this option:

Element		Type	Description

			`"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:
			
			`"override_log_level": {`
			`  "enabled": true,`
			`  "log_level": 5`
			`},`

`policies`		Top level item	List of policies referenced by the connection policy.
	`aa_plugin`	string	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).
	`analytics_policy`	string	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).
	`archive_cleanup_policy`	string	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).

Element		Type	Description
	`audit_policy`	string	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).
	`backup_policy`	string	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).
	`channel_policy`	string	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 `/api/configuration/rdp/channel_policies/` endpoint.
	`credential_store`	string	References the identifier of the credential store. You can configure credential stores at the `/api/configuration/policies/credential stores/` 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).
	`ldap_server`	string	References the identifier of the LDAP server. You can configure LDAP servers at

Element		Type	Description
			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).
	settings	string	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.
	usermapping_policy	string	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 usermapping_policies element, and remove any child elements (including the key).
rate_limit		Top level element	Connection rate limit.
	enabled	boolean	Set to true to provide a connection rate limit.
	value	int	The number of connections (per minute) that are allowed in the connection policy.
remote_desktop_gateway		Top level element	Configure SPS to act as a Remote Desktop Gateway. Otherwise, simply disable this option: `"remote_desktop_gateway": {`

Element		Type	Description
			`"enabled": false` `},`
`server_address`		Top level item	Defines the address where the clients connect to.
`source_address`		Top level element	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.
	`selection`	string	Configures Source Network Address Translation. Possible values are: • `box_address` Default. Uses the network address of the logical interface of SPS. • `original` Uses the IP address of the client, as seen by SPS. • `fix` Uses a fixed address when connecting to the remote server. Must be used with the `address` element.
	`address`	string	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.
`transport_security`		Top level element	Configures the encryption used in the sessions.
	`certificate`	JSON object	Selects the certificate to show to the peers. You have the following options: • **Use a self-signed certificate:**

Element	Type	Description

		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.
		`"certificate": { "selection": "self_signed" }`
		• **Use the same certificate for each client:**
		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 300.
		`"certificate": { "selection": "fix", "x509_identity": "893b7eb7-8c6f-403a-ba3a-1d09dc4b4c7a" }`
		• **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 418.
		`"certificate": { "selection": "generate", "signing_ca": "1904188625a843f11d30a5" },`

Element		Type	Description

	`server_certificate_check`	JSON object	By default, SPS accepts any certificate shown by the server.
			`"server_certificate_check": {`
			`  "enabled": false`
			`},`
			
			To verify the certificate of the destination server, configure and reference a Trusted CA list.
			`"server_certificate_check": {`
			`  "enabled": true,`
			`  "trusted_ca": "9106862955a844051d7bf6"`
			`},`

	`legacy_fallback`	boolean	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`	legacy | tls	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,

Element		Type	Description

			Windows Server 2003 or older).
			CAUTION:
			Security Hazard!
			Selecting this option can significantly reduce the strength of the encryption used!
			`"transport_security": { "selection": "legacy" },`
			• `tls`: Enables TLS-encryption. Note that you must also set the `certificate`, `server_certificate_check`, and `legacy_fallback` options.
			`"transport_security": { "certificate": { "selection": "self_signed" }, "server_certificate_check": { "enabled": false }, "legacy_fallback": false, "selection": "tls" }`

`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

Element		Type	Description
			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

Elements of access_control	Type	Description
authorizer	string	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.
permission	string	Defines the permissions of the authorizer usergroup. Possible values are: • audit The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection. • authorize The usergroup with the authorize permission can authorize connection requests. • audit_and_authorize The usergroup with the audit_and_authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.
require_different_ip	boolean	Set to true to require the authorizing user and its subject to have different IP addresses.
require_different_	boolean	Set to true to require the authorizing user and its subject to have different usernames.

Elements of `access_control`		Type	Description
`username`			
`subject`		Top level item	Defines the subjects of the access control policy.
	`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 `remote_desktop_gateway`	Type	Description
`enabled`	boolean	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 "Using One Identity Safeguard for Privileged Sessions (SPS) as a Remote Desktop Gateway" in the Administration Guide.
`certificate`	JSON object	To act as a Remote Desktop Gateway, SPS needs to display a certificate to the clients. • To automatically create new certificates on SPS for every client, set `"selection": "generate"`, and reference the Certificate Authority (CA) to sign the generated certificates. For example:

Elements of `remote_desktop_gateway`	Type	Description

		`"certificate": {`
		`"selection": "generate",`
		`"value": {`
		`"signing_ca": "53449998258a4ceba80fdc"`
		`},`
		`"common_name": "examplecn"`
		`}`
		
		For details on uploading certificates, see Certificates stored on SPS on page 300.
		
		• To always display the same certificate, set `"selection": "fix"`, and reference an X.509 certificate and the matching private key. For example:
		
		`"certificate": {`
		`"selection": "fix",`
		`"value": "1904188625a843f11d30a5"`
		`},`
		
		For details on creating a signing CA, see Signing CA policies on page 418.

	selection	generate | fix	Determines if SPS displays the same certificate to every client (`fix`), or generates a separate certificate (`generate`) for every client.
	value	JSON object or string	Contains the options and parameters related to the option set in `selection`. • If `selection` is set to `generate`, this is a JSON object.

Elements of `remote_desktop_gateway`			Type	Description
				• If `selection` is set to `fix`, this is a string containing the reference ID of the certificate that SPS displays to the clients.
		common_name	string	Available only if `selection` is set to `generate`. You can specify the Common Name of the generated certificates in this parameter. For example: `"common_name": "examplecn"` If set to null, the Common Name of the certificates will be `SPS-hostname.domainname`
		signing_ca	string	Available only if `selection` is set to `generate`. Contains the reference key of the signing CA used to sign the certificates that SPS shows to the clients. For example: `"signing_ca": "1904188625a843f11d30a5"` If set to null, the Common Name of the certificates will be `SPS-hostname.domainname`
`local_authentication`			JSON object	Determines how SPS authenticates the clients: either using Active Directory (SPS must be member of a domain), or using a Local User Database.
	selection		active_directory | local_user_database	Determines how SPS authenticates the clients: • using Active Directory (SPS must be member of a domain) `"local_authentication": {`

Elements of `remote_desktop_gateway`	Type	Description

		`"selection": "local_user_database", "value": { "domain": "example", "local_user_database": "15646962145a843f758501-d" }`
		 • using a Local User Database.
		 `"local_authentication": { "selection": "active_directory", "value": null }`

value	JSON object	Set to `null` if `selection` is set to `active_directory`. If `selection` is set to `local_user_database`, `value` contains a JSON object with the `domain` and `local_user_database` keys.
domain	string	Available only if `selection` is set to `local_user_database`.
local_user_database	string	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.

Elements of `server_address`	Type	Description
custom_dns	string	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`

Elements of `server_address`	Type	Description

		endpoint.
		• 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" }, ... },`

`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`

Elements of `server_address`	Type	Description
		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.
`port`	int	Must be used if `selection` is set to `fix`. The port of the target server.
`domains`	Top level list	Must be used if `selection` is set to `inband`.
`domain`	Top level item	Lists the address ranges that are included in the connection policy.
`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 targer 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.

Elements of `server_address`	Type	Description
`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` 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 provided order, and uses the first successfully resolved address to establish the connection.

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"
      }
   }
```

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
}
```

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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 35.

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 32.

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 35.

RDP channels

The available RDP channel types and their functionalities are described below. For details on configuring channel policies, see Channel policy.

Channel	Special options	Description
`#drawing`	Yes	**Drawing**: Enables access to the server's graphical desktop (screen). This channel must be enabled for RDP to work. Channel-specific actions:

Channel	Special options	Description

		• `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,`
		`  "ids": false`
		`}`

`cliprdr`	None	**Clipboard**: 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.
`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

Channel	Special options	Description
		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.
`seamrdp`	None	**Seamless**: Enable seamless channels that run a single application on the RDP server, instead of accessing the entire desktop.
`drdynvc`	Yes	**Dynamic virtual channel**: 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 `drdynvcs` field. Channel-specific access control rules: • `drdynvcs` (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.
`rdpdr-serial`	Yes	**Serial redirect**: Enables access to serial-port redirections. To restrict access to specific redirections, list the unique names of the channels in the `devices` field. 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.
`rdpdr-parallel`	Yes	**Parallel redirect**: Enables access to parallel-port redirections. To restrict access to specific redirections, list the unique names of the channels in the `devices` field. 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.
`rdpdr-printer`	Yes	**Printer redirect**: Enables access to printer-port redirections. To restrict access to specific redirections, list the unique names of the channels in the `devices` field. Channel-specific access control rules: • `devices` (list): To permit only specific redirections, list the

Channel	Special options	Description
		unique name of the redirection in this field. Leave it empty to permit access to every redirection available.

`rdpdr-disk`	Yes	**Disk redirect**: Enables access to shared disk drives. To restrict access to specific redirections, list the unique names of the channels in the `devices` field, for example:
		`"devices": [`
		`  "C:"`
		
		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.

`rdpdr-scard`	Yes	**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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "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"
  }
}
```

Element	Type	Description
`key`	string	Top level element, contains the ID of the endpoint.
`body`	Top level element (string)	Contains the domain membership configuration.
`domain`	string	The name of the domain. Must be used if `enabled` is set to `true`.
`enabled`	boolean	Set to `true` to configure domain membership.
`realm`	string	Name of the realm. Must be used if `enabled` is set to `true`.

Examples:

Configure domain membership for the "test" domain on the "config.api" realm:

```
{
    "domain":  "test",
    "enabled":  true,
    "realm":  "test.config.api"
 }
```

Disable domain membership.

```
{
    "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 32.

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_embership/` 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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Global RDP options

List of options that affect all RDP connections.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "body": {
    "channel_database_cleanup": {
      "enabled": false
    },
    "service": {
      "enabled": false
    }
  }
  "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"
  }
}
```

Element		Type	Description
key		Top level item	Contains the ID of the endpoint.
body		Top level item	Contains the elements of the global RDP options.
	channel_database_cleanup	Top level item	Contains settings for database cleanup.
	service	Top level item	Global setting to enable RDP connections, and specify the logging detail.

Elements of `channel_database_cleanup`	Type	Description
days	integer	Applies only if `enabled` is set to `true`. 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).
enabled	boolean	To enable the global cleanup of RDP connection metadata, set `enabled` to **`true`**.

Elements of `service`	Type	Description
log_level	integer	Applies only if `enabled` is set to `true`. Defines the logging detail of RDP connections.
enabled	boolean	To enable RDP connections, set to **`true`**.

Examples

Querying the full list of global RDP options:

```
{
    "body": {
       "channel_database_cleanup": {
          "enabled": true,
          "days": 365
       },
       "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"
    }
 }
```

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 32.

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.

   For more information about the elements of the `channel_database_cleanup` item, see Elements of `channel_database_cleanup`.

   For more information about the elements of the `service` item, see Elements of `service`.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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 the SPS REST API attempted to access, but could not retrieve.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section contains the `path` that the SPS REST API attempted to access, but could not retrieve.
404	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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`**.

Cookie name	Description	Required	Values
			For more information 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 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 more information on 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
      },
      "server_screen": 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"
  }
}
```

Element	Type	Description
key	string	Top level element, contains the ID of the policy.
body	Top level element (string)	The elements of the RDP settings policy.
autologon_domain_suffix	string	Enter the suffix that the client will append to the domain when using autologon in conjunction with Network Level Authentication (CredSSP).
name	string	Name of the RDP settings policy. Cannot contain whitespace.
permit_unreliable_usernames	boolean	Set to `true` to automatically terminate RDP connections if SPS cannot reliably extract the username.
preconnect_channel_check	boolean	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`.
protocol_features	Top level item	Settings for RDP protocol versions, and Network Layer Authentication.
screen	Top level item	Display size and depth settings.
timeout	int	Connection timeout, in seconds.
inactivity_timeout	Top level element	
enabled	boolean	• `true`: If no user activity is detected, it terminates the session after the

Element	Type	Description
		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`	Type	Description
`nla`	Top level item	Settings for Network Level Authentication.
`enabled`	boolean	Set to `true` to enable Network Level Authentication. If set to true, the require_domain_membership element is required in the JSON.
`require_domain_membership`	boolean	Set to true to require domain membership. Must be in the JSON if NLA is `enabled`.

Elements of `screen`	Type	Description
`maximum_bpp`	int	The maximum allowed color depth of the remote desktop, in bits. The following values are valid: 8, 15, 16, 24.
`maximum_height`	int	The maximum allowed height of the remote desktop, in pixels.
`maximum_width`	int	The maximum allowed width of the remote desktop, in pixels.

Examples:

Turn off NLA.

```
{
  "autologon_domain_suffix": "-AUTO",
  "name": "API_test",
  "permit_unreliable_usernames": true,
  "preconnect_channel_check": true,
  "protocol_features": {
```

```
    "nla": {
      "enabled": false
    },
    "server_screen": true
  },
  "screen": {
    "maximum_bpp": 24,
    "maximum_height": 2000,
    "maximum_width": 2000
  },
  "timeout": 600
}
```

Configure NLA.

```
{
    "autologon_domain_suffix": "-AUTO",
    "name": "API_test",
    "permit_unreliable_usernames": true,
    "preconnect_channel_check": true,
    "protocol_features": {
       "nla": {
          "enabled": true,
          "require_domain_membership": false
       },
       "server_screen": 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 32.

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 35.

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 32.

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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.

Code	Description	Notes
400	Bad Request "message": "RDP Settings Policy 'API_test': SPS must be a domain member to allow enabling Network Level Authentication."	You have set `require_domain_membership` to `true`, but SPS is not the member of a domain.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

SSH connections

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

Cookie name	Description	Required	Values
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 more information 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 available settings for configuring for SSH connections.

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

Response

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

For more information on the `meta` object, see Message format on page 10.


```
{
    "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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 SSH connection policies.

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

The following command retrieves the properties of a specific policy.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "8348340645707e2575e3c6",
      "meta": {
        "href": "/api/configuration/ssh/connections/8348340645707e2575e3c6"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/ssh/authentication_policies",
    "href": "/api/configuration/ssh/connections",
    "last": "/api/configuration/ssh/settings_policies",
```

```
    "next": "/api/configuration/ssh/options",
    "parent": "/api/configuration/ssh",
    "previous": "/api/configuration/ssh/channel_policies",
    "remaining_seconds": 600,
    "transaction": "/api/transaction"
  }
}
```


When retrieving the endpoint of a specific SSH 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
    },
     "client_side_hostkeys": [
         {
              "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
              "meta": {
                  "href": "/api/configuration/private_keys/XXXXXXXX-XXXX-XXXX-XXXX-
XXXXXXXXXXXX"
              }
          }
    ],
    "server_side_hostkey_settings": {
        "hostkey_check": "accept-first-time"
    },
    "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/512524636571b903540804"
        }
      },
      "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"
    },
    "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"
  }
```


Element	Type	Description
key	string	Top level element, contains the ID of the connection policy.
body	Top level element (string)	The elements of the connection policy.
access_	Top	Collection of access policies. Access

Element		Type	Description
control		level list	policies define who can authorize and audit a connection.
active		boolean	Set to false to suspend the connection policy. Connection settings are preserved.
channel_database_cleanup		Top level item	Configures cleanup of the connection metadata on the connection policy's level.
	days	int	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.
	enabled	boolean	Set to true to enable periodical cleanup of the connection metadata.
indexing		Top level item	Configures indexing for the connection policy.
	enabled	boolean	Set to true to enable indexing the connections.
	policy	string	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).
	priority	int	Specifies the indexing priority for the connection. Possible values are: • 5 Very low priority. • 4

Element		Type	Description
			Low priority. • `3` Normal (default) priority. • `2` High priority. • `1` Very high priority. • `0` Near real-time priority.
`log_audit_trail_downloads`		boolean	Set to `true` to log audit trail downloads.
`name`		string	The name of the connection policy.
`network`			
	`clients`	list, string	List of client ("from") IP addresses.
	`ports`	list, integers	List of target ports.
	`targets`	list, string	List of target IP addresses.

`override_log_level`		Top level item	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/<protocol>/options` endpoint is used.
			• 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 option and set the log level to use:

Element		Type	Description

			`"override_log_level": {`
			`  "enabled": true,`
			`  "log_level": 5`
			`},`

`policies`		Top level item	List of policies referenced by the connection policy.
	`aa_plugin`	string	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).
	`analytics_policy`	string	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).
	`archive_cleanup_policy`	string	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).
	`audit_policy`	string	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

Element		Type	Description
			value of the returned `key` as the value of the `audit_policy` element, and remove any child elements (including the key).
	`authentication_policy`	string	Cannot be `null`. 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).
	`backup_policy`	string	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).
	`channel_policy`	string	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.
	`credential_store`	string	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

Element		Type	Description
			the value of the returned `key` as the value of the `credential_store` element, and remove any child elements (including the key).
	`ldap_server`	string	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).
	`settings`	string	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 SSH settings policies at the `/api/configuration/ssh/settings_policies/` endpoint.
	`usermapping_policy`	string	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 `usermapping_policies` element, and remove any child elements (including the key).
`rate_limit`		Top level element	Connection rate limit.
	`enabled`	boolean	Set to true to provide a connection rate limit.

Element		Type	Description
	value	int	The number of connections (per minute) that are allowed in the connection policy.
server_address		Top level item	Defines the address where the clients connect to.
server_side_hostkey		Top level element	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.
source_address		Top level element	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.
	selection	string	Configures Source Network Address Translation. Possible values are: • box_address Default. Uses the network address of the logical interface of SPS. • original Uses the IP address of the client, as seen by SPS. • fix Uses a fixed address when connecting to the remote server. Must be used with the address element.
	address	string	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.
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-

Element		Type	Description
			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`	Type	Description
`authorizer`	string	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.
`permission`	string	Defines the permissions of the `authorizer` usergroup. Possible values are: • `audit` The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection. • `authorize` The usergroup with the authorize permission can authorize connection requests. • `audit_and_authorize` 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`	Type	Description
`require_different_ip`	boolean	Set to `true` to require the authorizing user and its subject to have different IP addresses.
`require_different_username`	boolean	Set to `true` to require the authorizing user and its subject to have different usernames.
`subject`	Top level item	Defines the subjects of the access control policy.
`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 `client_side_hostkeys`	Type	Description
key	string	Configures the keys that SPS shows to the clients.

Elements of `server_address`	Type	Description
custom_dns	string	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. • To use the default DNS, disable this

Elements of `server_address`	Type	Description

		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"`
		`},`
		`...`
		`},`

`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_`

Elements of `server_address`	Type	Description
		`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.
`port`	int	Must be used if `selection` is set to `fix`. The port of the target server.
`domains`	Top level list	Must be used if `selection` is set to `inband`.
`domain`	Top level item	Lists the address ranges that are included in the connection policy.
`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 targer 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.

Elements of `server_address`	Type	Description
`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 provided order, and uses the first successfully resolved address to establish the connection.

Elements of `server_side_hostkey_settings`	Type	Description
`hostkey_check`	string	Defines the method for checking the host keys of the target server. Possible values are: • `disabled`

Elements of `server_side_hostkey_settings`	Type	Description
		Disables host key verification. • `accept-first-time` Records the key shown for the first connection, and accepts only the same key for any subsequent connections. • `accept-known-keys` Only accepts host keys that are already stored on SPS. You can manage host keys at the `/api/ssh-host-keys` endpoint.

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_hostkeys": [
                        {
                            "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                            "meta": {
                                "href": "/api/configuration/private_keys/XXXXXXXX-XXXX-
XXXX-XXXX-XXXXXXXXXXXX"
                            }
                        }
                    ],
```

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_settings": {
      "hostkey_check": "accept-first-time"
  }
```

Server-side hostkey: only accept X.509 certificates that are verified by a trusted CA.

```
  "server_side_hostkey_settings": {
      "hostkey_check": "accept-known-keys"
  }
```

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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 35.

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 32.

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 35.

SSH channels

The available SSH channel types and their functionalities are described below. For details on configuring channel policies, see Channel policy.

Channel	Special options	Description
`auth-agent`	None	**Agent**: Forwards the SSH authentication agent from the client to the server.
`x11`	Yes	**X11 Forward**: 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:

```
"networks": [
  {
    "selection": "address",
    "value": "192.168.1.1"
  },
  {
```

Channel	Special options	Description

		`"selection": "address",`
		`"value": "192.168.1.2"`
		`}`
		
		NOTE: Certain client applications send the Target address as a hostname, while others as an IP address. If you are using a mix of different client applications, you might have to duplicate the channel rules and create IP-address and hostname versions of the same rule.
		
		Channel-specific access control rules:
		
		• `networks` (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-forwards`	Yes	**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:
		
		`"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

Channel	Special options	Description
		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-points of the forwarded traffic (that is, the local host that sends data to the remote host), and not the SSH server or the client. For example, to enable forwarding from the `192.168.20.20` host to the remote host `192.168.50.50`, enter `192.168.20.20` into the `originator_address`, and `192.168.50.50` into the `host_address` field. For details, see Limiting addresses in port forwarding.
`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 forwardins, 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

Channel	Special options	Description
		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.
`session-exec`	Yes	**Session Exec**: Execute a remote command (for example `rsync`) without opening a session shell. List the permitted command in the `execs` 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 (`{} ()*?\\\|[]`). **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: `"execs": [ "top", "ls"`
`session-exec-scp`	Yes	**Session Exec SCP**: Transfers files using the Secure Copy (SCP) protocol. 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:

Channel	Special options	Description

		`"actions": {`
		`  "audit": false,`
		`  "four_eyes": false,`
		`  "ids": false,`
		`  "log_transfer_to_db": true,`
		`  "log_transfer_to_syslog": true`
		`}`

`session-subsystem`	Yes	**Session Subsystem**: Use a subsystem. Enter the name of the permitted subsystem into the `subsystems` field.
		Channel-specific access control rules:
		• `subsystems` (list): List the permitted subsystems in this field.
		For example:
		`"execs": [`
		`  "top",`
		`  "ls"`

`session-exec-sftp`	Yes	**Session SFTP**: Transfers files using the Secure File Transfer Protocol (SFTP).
		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:
		`"actions": {`
		`  "audit": false,`
		`  "four_eyes": false,`
		`  "ids": false,`
		`  "log_transfer_to_db": true,`
		`  "log_transfer_to_syslog": true`
		`}`

`session-shell`	Yes	**Session Shell**: The traditional remote terminal session. Channel-specific actions:

Channel	Special options	Description
		• `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, "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.

Element		Type	Description
`connected_address`, `host_address`, `network`, or `originator_address`		list of JSON objects	Container objects for limiting access to port-forwarding in SSH channel policies. For details, see SSH channels on page 612.
	`selection`	`address` or `network`	Specifies the type of the address. Possible values: `address` or `network`
	`value`	IPv4 address or network	The IP address, or the network in `IP-address:prefix` format. For example, `192.168.1.1` or `192.168.0.0/16`

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "-200",
      "meta": {
        "href": "/api/configuration/ssh/authentication_policies/-200"
```

```
      }
    },
    {
      "key": "1895203635707e3340262f",
      "meta": {
        "href": "/api/configuration/ssh/authentication_
policies/1895203635707e3340262f"
      }
    }
  ],
  "meta": {
    "first": "/api/configuration/ssh/authentication_policies",
    "href": "/api/configuration/ssh/authentication_policies",
    "last": "/api/configuration/ssh/settings_policies",
    "next": "/api/configuration/ssh/channel_policies",
    "parent": "/api/configuration/ssh",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```


When retrieving the endpoint of a specific policy, the response is the following.


```
{
  "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",
    "parent": "/api/configuration/ssh/authentication_policies",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

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.
name	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. • `ldap` Uses the LDAP server selected for the connection policy. LDAP servers can be configured in the `/api/configuration/policies/l`

Elements of backend		Type	Description
			dap_servers endpoint). To use this option, you must also configure the password and public_key elements. • 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.
	enabled	boolean	Set it to true to enable public key-based client-side authentication.
user_database		string	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).
servers		Top level list	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.
	address	Top level element	Defines the address of a RADIUS server.

Elements of backend			Type	Description
		`selection`	string	Required child of the address element. Possible values are: • `ip` The `value` element contains the IP of the RADIUS server. • `fqdn` The `value` element contains the FQDN of the RADIUS server.
		`value`	string	The IP or the FQDN address of the RADIUS server.
	`port`		int	The port number of the RADIUS server.

	`shared_secret`		string	References the key of the shared secret for the RADIUS server. You can configure shared secrets at the `/api/configuration/passwords/` 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.
				`"shared_secret": {`
				`"plain": "<new-password>"`
				`}`

`authentication_protocol`			Top level item	RADIUS setting. Set to `pap` to use the Password Authentication Protocol. To use the Challenge-Handshake Authentication Protocol, set it to `chap`.

Elements of `gateway_methods`	Type	Description
`kerberos`	boolean	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

Elements of `gateway_methods`	Type	Description
		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`).
`password`	boolean	Authentication based on username and password. Set it to `true` to enable password-based client-side authentication.
`public_key`	Top level item	Authentication based on public-private encryption keypairs.

Elements of `relayed_methods`	Type	Description
`kerberos`	boolean	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`).
`keyboard_interactive`	boolean	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.
`password`	boolean	Authentication based on username and password. Set to `true` to enable password-based authentication on the remote server.

Elements of `relayed_methods`	Type	Description
`public_key`	Top level item	Authentication based on public-private encryption keypairs. Use the `selection` child element to disable or configure authentication using public-private keypairs on the remote server.
`selection`	string	Configures authenticaton on the remote server using public-private keypairs. The following values are possible: • `disabled` Disables the authentication method. • `publish_to_ldap` 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. • `fix` Uses a private key in the server-side connection. You have to use the `private_key` element to reference the private key. • `agent` 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

Elements of `relayed_methods`	Type	Description
		fail. For details on enabling agent forwarding in your SSH application, see the documentation of the application.
`private_key`	string	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 `/api/configuration/private_keys/` endpoint. To modify or add a private key, use the value of the returned `key` as the value of the `private_key` element, and remove any child elements (including the key).

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>"
       },
       "gateway_methods":  {
              "kerberos":  false,
              "password":  true,
              "public_key":  true
       },
       "relayed_methods":  {
              "kerberos":  false,
              "keyboard_interactive":  false,
              "password":  true,
              "public_key":  {
                     "selection":  "disabled"
              }
       },
       "name":  "passwords",
 }
```

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.

```
{
    "backend": {
        "authentication_protocol": "pap",
        "selection": "radius",
        "servers": [
            {
                "address": {
                    "selection": "ip",
                    "value": "192.168.1.1"
                },
                "port": 1812,
                "shared_secret": <key-of-shared-secret>,
            }
        ]
    }
   "gateway_methods": {
        "kerberos": false,
        "password": true,
        "public_key": false
    },
    "relayed_methods": {
        "kerberos": false,
        "keyboard_interactive": true,
```

```
        "password": true,
        "public_key": {
            "selection": "agent"
        }
    },
    "name": "RADIUS"
}
```

Using Kerberos authentication both on the client side and on the remote server.

```
{
       "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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section

Code	Description	Notes
		contains the `path` that was attempted to be accessed, but could not be retrieved.
404	NotFound	The requested object does not exist.

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 32.

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:

   ```
   {
       "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 35.

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 32.

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 gateway_methods. The elements of `relayed_methods` are listed in Elements of relayed_methods.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

Global SSH options

List of options that affect all SSH connections.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
    "body": {
       "channel_database_cleanup": {
          "enabled": false
       },
       "service": {
          "enabled": false
       },
       "gssapi": {
          "enabled": false
       }
    }
    "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"
    }
 }
```


Element		Type	Description
key		Top level item	Contains the ID of the endpoint.
body		Top level item	Contains the elements of the global SSH options.
	`channel_database_cleanup`	Top level item	Contains settings for database cleanup.
	`service`	Top level item	Global setting to enable SSH connections, and specify the logging detail.

Element		Type	Description
gssapi	Top level item	Global option to configure Kerberos authentication with SPS. For more information about Kerberos authentication with SPS, see Kerberos authentication settings in the *SPSAdministration Guide*.	

Elements of `channel_database_cleanup`	Type	Description
`days`	integer	Applies only if `enabled` is set to `true`. 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).
`enabled`	boolean	To enable the global cleanup of SSH connection metadata, set `enabled` to **`true`**.

Elements of `service`	Type	Description
`log_level`	integer	Applies only if `enabled` is set to `true`. Defines the logging detail of SSH connections.
`enabled`	boolean	To enable SSH connections, set to **`true`**.

Elements of `gssapi`	Type	Description
`enabled`	boolean	To enable mapping hostnames to Kerberos realms, set to **`true`**. NOTE: Enabling this option is only required if you have more than one realms deployed on your network.
`domain_realm_mapping`	list of JSON objects	When you have more than one realms deployed on your network, and `enabled` is set to **`true`**, the list of JSON objects in `domain_realm_`

Elements of `gssapi`		Type	Description
			`mapping` specify the mapping between the servers' DNS domain names and the names of their respective connected Kerberos realms.
	`domain`	string	The name of your DNS domain.
	`realm`	string	The name of your Kerberos realm.

Examples

Querying the full list of global SSH options:

```
{
  "body": {
    "channel_database_cleanup": {
      "enabled": true,
      "days": 365
    },
    "service": {
      "enabled": true,
      "log_level": 4
    },
    "gssapi": {
      "enabled": true,
      "domain_realm_mapping": [
        {
        "domain": "server.example.com",
        "realm": "SERVER.EXAMPLE.COM"
        },
        {
        "domain": ".example.com",
        "realm": "EXAMPLE.COM"
        }
      ]
    }
    "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"
    }
}
```

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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 the SPS REST API attempted to access, but could not retrieve.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section contains the `path` that the SPS REST API attempted to access, but could not retrieve.
404	NotFound	The requested object does not exist.

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 32.

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.

   For more information about the elements of the `channel_database_cleanup` item, see Elements of `channel_database_cleanup`.

For more information about the elements of the `service` item, see Elements of `service`.

For more information about the elements of the `gssapi` item, see Elements of `gssapi`.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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

Cookie name	Description	Required	Values
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 more information 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 SSH settings policies.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/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 more information on 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",
"blowfish-cbc", "cast128-cbc", "aes192-cbc", "aes256-cbc", "3des-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",
"blowfish-cbc", "cast128-cbc", "aes192-cbc", "aes256-cbc", "3des-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"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the policy.
body	Top level element (string)	The elements of the SSH settings policy.
client_side_algorithms	Top level element (list)	Lists the permitted client-side encryption parameters.
cipher	list	Lists the permitted client-side cipher algorithms.
compression	list	Lists the permitted client-side compression algorithms.

Element		Type	Description
	kex	list	Lists the permitted client-side KEX algorithms.
	mac	list	Lists the permitted client-side MAC algorithms.
greeting		string	Greeting message for the connection.
name		string	Name of the SSH settings policy.
preconnect_channel_check		boolean	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`.
server_side_algorithms		Top level element (list)	Lists the permitted server-side encryption parameters.
	cipher	list	Lists the permitted server-side cipher algorithms.
	compression	list	Lists the permitted server-side compression algorithms.
	kex	list	Lists the permitted server-side KEX algorithms.
	mac	list	Lists the permitted server-side MAC algorithms.
software_version		string	Specifies additional text to append to the SSH protocol banner sent by the server upon connection.
strict_mode		boolean	When this option is enabled, SPS rejects connections that use unrealistic parameters (for example, 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

Element			Type	Description
				with certain client or server applications. Strict mode is allowed by default. To turn it off, set the parameter to `false`.
	`timeout`		int	Connection timeout, in seconds.
	`inactivity_timeout`		Top level element	
		`enabled`	boolean	• `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 (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.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
403	Unauthorized	The requested resource cannot be retrieved because the

Code	Description	Notes
		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	NotFound	The requested object does not exist.

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 32.

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:

```
{
    "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 35.

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 32.

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 35.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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.

```
{
  "key": "ssh-rsa-10.10.100.1:22",
  "meta": {
    "href": "/api/ssh-host-keys/ssh-rsa-10.10.100.1:22",
    "parent": "/api/ssh-host-keys"
  },
  "ssh-rsa-10.10.100.1:22": {
    "address": "10.10.100.1",
```

```
    "port": 22,
    "type": {
      "selection": "ssh-rsa",
      "value": "AAAAB3NzaC1yc2EAAAABIwAAAQEAxrtNxBZieXhBI2gJoAdsjKNq...=="
    }
  }
}
```

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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:

```
{
    "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 49. 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:

```
{
    "address": "10.110.0.1",
    "port": 22,
    "type": {
       "selection": "ssh-rsa",
       "value": "AAAAB3NzaC1yc2EAAAAD...zvMwgc=="
    }
 }
```

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 47. If the DELETE request is successful, the response includes only the meta object, for example:

```
{
       "meta":  {
             "href":  "/api/ssh-host-keys/ssh-rsa-10.10.20.35:22",
             "parent":  "/api/ssh-host-keys"
       }
 }
```

You must commit your changes to actually delete the object from SPS.

Telnet connections

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

Cookie name	Description	Required	Values
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 more information 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 available settings for configuring for Telnet connections.

```
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 more information on 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"
       }
 }
```

Item	Description
`connections`	List of Telnet connection policies.
`channel_policies`	List of available Telnet channel types.

Item	Description
`authentication_policies`	List of the configured authentication methods that can be used in a connection.
`pattern_sets`	List of the default and custom channel policies.
`options`	List of global Telnet options that affect all 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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",
```

```
    "href": "/api/configuration/telnet/connections",
    "last": "/api/configuration/telnet/options",
    "next": "/api/configuration/telnet/options",
    "order": "/api/configuration/telnet/connections/@order",
    "parent": "/api/configuration/telnet",
    "previous": "/api/configuration/telnet/channel_policies",
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
  }
}
```


When retrieving the endpoint of a specific Telnet Connection Policy, the response is the following.


```
{
    "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": {
```

```
            "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
key	string	Top level element, contains the ID of the connection policy.
body	Top level element (string)	The elements of the connection policy.
access_control	Top level list	Collection of access policies. Access policies define who can authorize and audit a connection.
active	boolean	Set to `false` to suspend the connection policy. Connection settings are preserved.
channel_	Top	Configures cleanup of the connection

Element		Type	Description
database_cleanup		level item	metadata on the connection policy's level.
client_side_transport_security		Top level item	Defines the Transport Layer Security (TLS) settings for the connection between SPS and the client. For example: `"client_side_transport_security": { "selection": "disabled" },`
	days	int	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 Telnet protocol are available at the `api/configuration/telnet/options` endpoint.
	enabled	boolean	Set to true to enable periodical cleanup of the connection metadata.
indexing		Top level item	Configures indexing for the connection policy.
	enabled	boolean	Set to `true` to enable indexing the connections.
	policy	string	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).
	priority	int	Specifies the indexing priority for the connection. Possible values are: • 5 Very low priority. • 4

Element		Type	Description
			Low priority. • `3` Normal (default) priority. • `2` High priority. • `1` Very high priority. • `0` Near real-time priority.
`log_audit_trail_downloads`		boolean	Set to `true` to log audit trail downloads.
`name`		string	The name of the connection policy.
`network`			
	`clients`	list, string	List of client ("from") IP addresses.
	`ports`	list, integers	List of target ports.
	`targets`	list, string	List of target IP addresses.

`override_log_level`		Top level item	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/<protocol>/options` endpoint is used.
			• 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 option and set the log level to use:

Element		Type	Description

			`"override_log_level": {`
			`  "enabled": true,`
			`  "log_level": 5`
			`},`

`policies`		Top level item	List of policies referenced by the connection policy.
	`aa_plugin`	string	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).
	`analytics_policy`	string	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).
	`archive_cleanup_policy`	string	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).
	`audit_policy`	string	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

Element		Type	Description
			the `audit_policy` element, and remove any child elements (including the key).
	`authentication_policy`	string	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).
	`backup_policy`	string	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).
	`channel_policy`	string	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.
	`credential_store`	string	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

Element		Type	Description
			remove any child elements (including the key).
	ldap_server	string	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).
	settings	string	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).
	usermapping_policy	string	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 usermapping_policies element, and remove any child elements (including the key).
rate_limit		Top level element	Connection rate limit.
	enabled	boolean	Set to true to provide a connection rate limit.
	value	int	The number of connections (per minute) that are allowed in the connection policy.
server_address		Top level item	Defines the address where the clients connect to.
server_side_		Top	Defines the Transport Layer Security

Element		Type	Description

transport_security		level item	(TLS) settings for the connection between SPS and the server. For example:
			`"server_side_transport_security": {`
			`  "selection": "disabled"`
			`},`

source_address		Top level element	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.
	selection	string	Configures Source Network Address Translation. Possible values are: • box_address Default. Uses the network address of the logical interface of SPS. • original Uses the IP address of the client, as seen by SPS. • fix Uses a fixed address when connecting to the remote server. Must be used with the address element.
	address	string	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.
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,	By default, any user can perform

Element		Type	Description
		string	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`	Type	Description
`authorizer`	string	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.
`permission`	string	Defines the permissions of the `authorizer` usergroup. Possible values are: • `audit` The usergroup with the audit permission can monitor ongoing connections, and download the audit trails of a closed and indexed connection. • `authorize` The usergroup with the authorize permission can authorize connection requests. • `audit_and_authorize` The usergroup with the audit_and_authorize permission can authorize connection requests, monitor connections, and download the audit trail of closed and indexed connections.
`require_different_ip`	boolean	Set to `true` to require the authorizing user and its subject to have different IP addresses.
`require_`	boolean	Set to `true` to require the authorizing user and its

Elements of `access_control`		Type	Description
`different_username`			subject to have different usernames.
`subject`		Top level item	Defines the subjects of the access control policy.
	`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 client_side_transport_security

Elements of `client_side_transport_security`	Type	Description

`peer_certificate_check`	Top level item	Sets how SPS authenticates the peers. To permit connections from peers without requesting a certificate, set `"enabled": false`, for example:
		`"peer_certificate_check": {`
		`  "enabled": false`
		`}`
		To validate the certificate of the peer, set `"enabled": true`, and reference a trusted certificate authority list, for example:

Elements of `client_side_transport_security`	Type	Description

		`"peer_certificate_check": {`
		`"enabled": true,`
		`"trusted_ca": "cfc815e5-dadb-4eb9-a628-12ae0c12d358"`
		`}`

`selection`	string	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: • `starttls` 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. • `tls` Require encryption.

`sps_certificate`	JSON object	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 fpr each session.
		• To use the same certificate for every session, set `selection: "fix"` and reference the certificate to use in the `x509_identity` option, for example:
		`"sps_certificate": {`
		`"selection": "fix",`
		`"x509_identity": "<'key' of an uploaded certificate>"`
		`},`
		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 `client_side_transport_security`	Type	Description
		`"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`	Type	Description

custom_dns	string	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.
		
		• 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"`

Elements of `server_address`	Type	Description

		`},`
		`...`
		`},`

`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.
`port`	int	Must be used if `selection` is set to `fix`. The port of the target server.
`domains`	Top level list	Must be used if `selection` is set to `inband`.
`domain`	Top level item	Lists the address ranges that are included in the connection policy.

Elements of `server_address`			Type	Description
		`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 targer 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` 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

Elements of `server_address`	Type	Description
		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 provided order, and uses the first successfully resolved address to establish the connection.

Elements of server_side_transport_security

Elements of `server_side_transport_security`	Type	Description

`peer_certificate_check`	Top level item	Sets how SPS authenticates the peers. To permit connections from peers without requesting a certificate, set `"enabled": false`, for example:
		`"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`	Type	Description
`selection`	string	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: • `none` Do not use encryption. • `starttls` 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. • `tls` Require encryption.
`sps_certificate`	JSON object	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 fpr each session. • If the server does not require a certificate from SPS, set `selection: "none"`. • To use the same certificate for every session, set `selection: "fix"` and reference the certificate to use in the `x509_identity` option, for example: `"sps_certificate": { "selection": "fix", "x509_identity": "<'key' of an uploaded certificate>" },` 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`	Type	Description

		`"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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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:

```
{
    "key": "a99be49b-b0a2-4cf9-b70d-fea1f9ea188f",
    "meta": {
       "href": "/api/configuration/telnet/connections/a99be49b-b0a2-4cf9-
b70d-fea1f9ea188f",
       "parent": "/api/configuration/telnet/connections",
       "transaction": "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

Telnet channels

The available Telnet channel types and their functionalities are described below.

Channel	Special options	Description
`telnet`	Yes	**telnet**: Enables access to the server's terminal. This channel must be enabled for Telnet to work. 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" } }, }`

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

Cookie name	Description	Required	Values
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`**.

Cookie name	Description	Required	Values
			For more information 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 Telnet authentication policies.

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

The following command retrieves the properties of a specific policy.

```
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 more information on 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",
```

```
    "href": "/api/configuration/telnet/authentication_policies",
    "last": "/api/configuration/telnet/settings_policies",
    "next": "/api/configuration/telnet/channel_policies",
    "parent": "/api/configuration/telnet",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```

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"
```

```
}
```

Element	Type	Description
`key`	string	Top level element, contains the ID of the policy.
`body`	Top level element	Contains the elements of the policy.
`name`	string	The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.
`active_pattern_sets`	JSON list	The list of patterns to use to extract the username from the sessions. For more information, see Extracting username from Telnet connections. For example: `"active_pattern_sets": ["-8000","-8001","-8002"]`
`backend`	Top level item	Client-side gateway authentication settings. The value of `selection` defines which authentication method is used.
`selection`	string	Defines the authentication method for client-side gateway authentication. Possible values are:

Element	Type	Description
		• `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 also configure the `user_database` element. • `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 `servers`		Type	Description
`address`		Top level element	Defines the address of a RADIUS server.
	`selection`	string	Required child of the address element. Possible values are: • `ip` The `value` element contains the IP of the RADIUS server. • `fqdn` The `value` element contains the FQDN of the RADIUS server.
	`value`	string	The IP or the FQDN address of the RADIUS server.
`port`		int	The port number of the RADIUS server.
`shared_secret`		string	References the key of the shared secret for the RADIUS server. You can configure shared secrets at

Elements of `servers`	Type	Description
		the `/api/configuration/passwords/` 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. `"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:

```
{
    "key": "telnet-auth-pol-3",
    "body": {
       "name": "telnet_local",
       "backend": {
          "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:


```
{
    "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-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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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:

```
{
  "key": "6f924f39-e4c9-4b0f-8018-8842e2115ebd",
  "meta": {
    "href": "/api/configuration/telnet/authentication_policies/6f924f39-
```

```
e4c9-4b0f-8018-8842e2115ebd",
    "parent": "/api/configuration/telnet/authentication_policies",
    "transaction": "/api/transaction"
  }
}
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

Global Telnet options

List of options that affect all Telnet connections.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 Telnet options.

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

Response

The following is a sample response received when listing the default global Telnet options.

For more information on the `meta` object, see Message format on page 10.

```
{
  "body": {
      "channel_database_cleanup": {
        "enabled": false
      },
      "service": {
        "enabled": false
      }
    }
  }
  "key": "options",
  "meta": {
    "first": "/api/configuration/telnet/channel_policies",
    "href": "/api/configuration/telnet/options",
    "last": "/api/configuration/telnet/options",
    "next": null,
```

```
    "parent": "/api/configuration/telnet",
    "previous": "/api/configuration/telnet/channel_policies",
    "transaction": "/api/transaction"
  }
}
```

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.
	channel_database_cleanup	Top level item	Contains settings for database cleanup.
	service	Top level item	Global setting to enable Telnet connections, and specify the logging detail.

Elements of `channel_database_cleanup`	Type	Description
days	integer	Applies only if `enabled` is set to `true`. 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 this element to `true`.

Elements of `service`	Type	Description
log_level	integer	Applies only if `enabled` is set to `true`. Defines the logging detail of Telnet connections.
enabled	boolean	Set to `true` to enable Telnet connections.

Examples

Querying the full list of global Telnet options:

```
{
    "body": {
       "channel_database_cleanup": {
          "enabled": true,
          "days": 365
        },
       "service": {
          "enabled": true,
          "log_level": 4
       }
    }
    "key": "options",
    "meta": {
       "first": "/api/configuration/telnet/channel_policies",
       "href": "/api/configuration/telnet/options",
       "last": "/api/configuration/telnet/options",
       "next": null,
       "parent": "/api/configuration/telnet",
       "previous": "/api/configuration/telnet/channel_policies",
       "transaction": "/api/transaction"
    }
 }
```

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 32.

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.

   For more information about the elements of the `channel_database_cleanup` item, see Elements of channel_database_cleanup.

   For more information about the elements of the `service` item, see Elements of `service`.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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 the SPS REST API attempted to access, but could not retrieve.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section contains the `path` that the SPS REST API attempted to access, but could not retrieve.
404	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information on authentication, see Authenticate to the SPS REST API on page

Cookie name	Description	Required	Values
			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 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 more information on 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",
    "previous": "/api/configuration/telnet/options",
    "remaining_seconds": 600,
    "transaction": "/api/transaction"
  }
}
```

Element	Type	Description
key	string	Contains the ID of the pattern set. The pattern set IDs can be used for specifying the `active_pattern_sets` JSON list at the configuration of Telnet authentication policies..
body	string	Contains the descriptive name of the pattern set.
name	string	Descriptive name of the pattern set.

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-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 41.

Code	Description	Notes
400	PatternSetValidationFailed	The validation of the telnet pattern set file failed.
401	Unauthenticated	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.
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.

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/configuratation/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 41.

Code	Description	Notes
404	NotFound	The requested object does not exist.
405	DeleteNotAllowedOnBuiltInPatternSet	The deletion of built-in telnet pattern sets is not allowed.

13

VNC connections

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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"
    }
 }
```

Item	Description
`channel_policies`	List of the default and custom channel policies.
`connections`	List of the VNC connection policies.
`options`	List of global VNC options that affect all 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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`**.

Cookie name	Description	Required	Values
			For more information 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

Cookie name	Description	Required	Values
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 more information on authentication, see

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).

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 more information on the `meta` object, see Message format on page 10.


```
{
    "body": {
          "channel_database_cleanup": {
             "enabled": false
          },
       "service": {
          "enabled": false
         }
      }
    }
    "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"
    }
 }
```


Element		Type	Description
key	Top	Contains the ID of the endpoint.	

Element	Type	Description
	level item	
body	Top level item	Contains the elements of the global VNC options.
channel_database_cleanup	Top level item	Contains settings for database cleanup.
service	Top level item	Global setting to enable VNC connections, and specify the logging detail.

Elements of `channel_database_cleanup`	Type	Description
days	integer	Applies only if `enabled` is set to `true`. 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).
enabled	boolean	To enable the global cleanup of VNC connection metadata, set this element to `true`.

Elements of `service`	Type	Description
log_level	integer	Applies only if `enabled` is set to `true`. Defines the logging detail of VNC connections.
enabled	boolean	Set to `true` to enable VNC connections.

Examples

Querying the full list of global VNC options:

```
{
    "body": {
       "channel_database_cleanup": {
          "enabled": true,
          "days": 365
        },
       "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"
    }
 }
```


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 32.

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.

   For more information about the elements of the `channel_database_cleanup` item, see Elements of `channel_database_cleanup`.

   For more information about the elements of the `service` item, see Elements of `service`.

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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 the SPS REST API attempted to access, but could not retrieve.
403	Unauthorized	The requested resource cannot be retrieved because the client is not authorized to access it. The `details` section contains the `path` that the SPS REST API attempted to access, but could not retrieve.
404	NotFound	The requested object does not exist.

14

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
    "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.


```
{
  "body": {
    "active": false,
    "alerts": {
      "href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq/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/rUhhQZ3jYsY1NDWYp9DEpq/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": "rUhhQZ3jYsY1NDWYp9DEpq",
  "meta": {
      "href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq",
      "parent": "/api/audit/sessions",
      "remaining_seconds": 594
  }
}
```


Element	Type	Description
key	string	Top level element, contains the key of the connection or audit trail.
body	Top level element (string)	Contains the properties of the connection.
active	boolean	If the returned value is `true`, the connection is ongoing.
alerts	Top level	Contains a link to the details of the alerts. For details, see Session alerts on page 747.

Element	Type	Description

	item	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.
		`"alerts": {`
		`"href": "/api/audit/sessions/7930f4308efe8aecd710202d815b76ff/alerts"`
		`},`

analytics	Top level item	Contains analytics details of the connection.

channels	Top level list	Contains a link to the details of the channel.
		`"channels": {`
		`"href": "/api/audit/sessions/svc-rUhhQZ3jYsY1NDWYp9DEpq-kecske-29/channels"`
		`},`

client	Top level item	The IP address and port number of the client.
creation_time	date	The time this document was created. In optimal cases this is near equal to the session's original `start_time`. However, it can be later than `start_time`.
duration	int	The duration of the session in seconds. Computed value.
end_time	ISO 8601 date	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.

events	Top level item	Contains a link to the details of the events. For details, see Session events on page 751.
		`"events": {`
		`"href": "/api/audit/sessions/7930f4308efe8aecd710202d815b76ff/events"`

Element		Type	Description
			`},`
hidden		boolean	True if this is a session that has not been displayed on the SPS GUI yet (due to fragmented data about the session).

indexer		Top level item	Contains the details of indexing. For details on configuring indexing, see Local services: configuring the indexer on page 761.
			`"indexer": {`
			`"href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq/indexer"`
			`},`

node_id		string	The node ID of the SPS machine where this session has been recorded.
origin		string	How SPA received this session. The following values are possible: • `PSM` for sessions based on an audit trail recorded by SPS. • `LOG` for sessions built from log data.
protocol		string	The protocol of the connection.
recording		Top level item	Contains the properties of the audit trail.
	archived	boolean	If the audit trail has been archived, this value is `true`, otherwise it is `false`. For details about the archiving, see the `archive object of the psm.audit_trail` field.
	audit_trail	Top level item	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 trail, see Download audit trails on page 713.
	auth_method	Top level item	**Authentication method**: The authentication method used in the connection. For example, `password`
	channel_policy	string	References the `name` of the channel policy. You can find the list of channel policies for each protocol at

Element		Type	Description
			the `/api/configuration/<protocol>/channel_policies/` endpoint.
	`command_extracted`	boolean	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.
	`connection_policy`	string	The name of the Connection Policy that handled the session, for example, `ssh_gateway_auth`. This is the name displayed on the **Control > 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/<protocol>/connections/` endpoint.
	`connection_policy_id`	string	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/<protocol>/connections/` endpoint.
	`content_reference_id`	long	The unique ID of the TCP connection.
	`has_accepted_channel`	boolean	True, if at least the connection has been built successfully, the authentication was successful, and there was actual traffic.
	`index_status`	string	**Channel's indexing status**: Shows if the channel has been indexed. The following values are possible: • `CHANNEL_OPEN (0)`: The connection of the channel is still open (indexer is waiting for the connection to close). • `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. • `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 are already indexed.

Element		Type	Description
			• `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.
	`network_id`	string	The ID of the Linux network namespace where the session originated from.
	`server_local`	Top level item	The IP address and port number of SPS.
	`session_id`	string	The identifier of the session.
	`target`	Top level item	The IP address and port number the client targeted for connection.
	`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`

Element		Type	Description
			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.
	`window_title_extracted`	boolean	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 `events object` field.
`revision`		int	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.
`server`		Top level item	The IP address and port number of the remote server.
`trail_downloads`		Top level item	Contains a link to the details of the audit-trail downloads in this session (if any). `"trail_downloads": { "href": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq/trail_downloads" },`
`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.

Element	Type	Description
`server_username`	string	The username used for authenticating on the remote server.
`verdict`	string	Indicates what SPS decided about the session. A session verdict that originates from log events or other external events.

Analytics elements	Type	Description

`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 avalialble, an error message (for example, the baseline has not been built yet).
`similar_sessions`	string	Collection of similar sessions from different sources.

Analytics elements	Type	Description
`tags`	string	The Analytics tags section in Search > Details.

Audit trail elements	Type	Description
`archive`	Top level element	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: `"audit_trail": { "archive": { "date": "2018-11-25T12:00:05.000Z", "path": "2018-11-23/", "policy": "8106930065bf7eb4c3cf59", "server": "\\\\10.20.30.40\\archive\\-abc123 (user: myuser)" }, "download": { "href": "/api/audit/ses-sions/10/audit_trail" } },`
`date`	ISO 8601 date	The date when the audit trail was archived in ISO 8601 date.
`server`	hostname or IP address	The address of the remote server where the audit trail was archived.
`path`	string	The path on the remote server where the audit trail was archived.
`policy`	string	The ID of the archiving policy that was used to archive the audit trail.
download	string	The download element allows downloading the audit trail.

Channel elements	Type	Description
`key`	string	Top level element, contains the ID of the channel.
`items`	Top level element (string)	The properties of the channel.

Channel elements	Type	Description
`active`	boolean	If the returned value is `true`, the session has not ended yet and the channel is active.
`audit_stream_id`	string	The identifier of the channel's audit stream. If the session does not have an audit trail, this element is not used.
`channel_id`	long	The unique ID of the channel.
`client_x509_subject`	string	The client's certificate in Telnet or VNC sessions. Available only if the **<Protocol name> Control > Connections > Client-side transport security settings > Peer certificate validation** is enabled in SPS.
`duration`	int	The duration of the connection. Computed value.
`end_time`	ISO 8601 date	The ISO 8601 date of the end of the connection. For ongoing connections, the value is `null`.
`rule_num`	string	The number of the line in the Channel policy applied to the channel.
`start_time`	ISO 8601 date	The ISO 8601 date of the start of the connection.
`type`	string	The type of the channel. Additional elements might be used with certain ICA, SSH and RDP channel types.
`verdict`	string	The channel's connection verdict. Possible values are: • `accept` The connection attempt was successful. • `deny` The connection attempt was denied. • `four-eyes-deferred` Four-eyes authorization is unable to progress as it is waiting for a remote username. • `four-eyes-error` An internal error occurred during four-eyes authorization. • `four-eyes-reject`

Channel elements	Type	Description
		The connection attempt was rejected by a four-eyes agent on SPS. • `four-eyes-timeout` Four-eyes authorization timed out.
`command`	string	Used with the `session exec` SSH channel type. The executed command.
`scp_path`	string	Used with the `session exec scp` SSH channel type. The folder used for Secure Copy.
`subsystem_name`	string	Used with the `session subsystem sftp` SSH channel type. The name of the used subsystem.
`originator.ip`	string	Used with the `local forward` and `remote forward` SSH channel types. The source address of the forwarded traffic.
`originator.name`	string	The source host name of the forwarded traffic. If this information is not available, the value is the IP address instead.
`originator.port`	int	Used with the `local forward` and `remote forward` SSH channel types. The source port of the forwarded traffic.
`connected.ip`	string	Used with the `local forward` and `remote forward` SSH channel types. The target address of the forwarded traffic.
`connected.name`	string	The target host name of the forwarded traffic. If this information is not available, the value is the IP address instead.
`connected.port`	int	Used with the `local forward` and `remote forward` SSH channel types. The target port of the forwarded traffic.
`dynamic_channel`	string	Used with the `dynamic virtual` RDP channel type. The name of the dynamic channel.
`device_name`	string	Used with the `serial redirect`, `parallel redirect`, `printer redirect`, `disk redirect`,

Channel elements	Type	Description
		and `scard redirect` RDP channel types. The name of the device.
`application`	string	Used with ICA connections. The name of the application accessed in a seamless Citrix ICA connection.
`four_eyes_authorizer`	string	The username of the user who authorized the session. Available only if four-eyes authorization is required for the channel.
`four_eyes_description`	string	The description of the session submitted by the authorizer of the session. Available only if four-eyes authorization is required for the channel.

Client elements	Type	Description

`client`	Top level element	The IP address and port number of the client. For example:
		`"client": {`
		`"ip": "10.20.30.40",`
		`"port": 59125`
		`},`

`ip`	string	The IP address of the client.
`name`	string	The host name of the client. If this information is not available, the value is the IP address instead.
`port`	int	The port number of the client.

Server elements	Type	Description

`server`	Top level element	The IP address and port number of the remote server. For example:
		`"server": {`
		`"ip": "10.20.30.40",`
		`"port": 55386`
		`},`

`ip`	string	The IP address of the remote server.

Server elements	Type	Description
name	string	The host name of the remote server. If this information is not available, the value is the IP address instead.
port	int	The port number of the remote server.

Server_local elements	Type	Description
server_local	Top level element	The IP address and port number of SPS. For example: `"server_local": { "ip": "10.20.30.40", "port": 55386 },`
ip	string	The IP address of SPS.
name	string	The host name of SPS. If this information is not available, the value is the IP address instead.
port	int	The port number of SPS.

Target elements	Type	Description
target	Top level element	The IP address and port number the client targeted for connection. For example: `"target": { "ip": "10.20.30.40", "port": 221 },`
ip	string	The IP address the client targeted for connection.
name	string	The host name of the client targeted for connection. If this information is not available, the value is the IP address instead.
port	int	The port number the client targeted for connection.

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 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": "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
```

```
    }
  },
  {
    "key": "4",
    "meta": {
      "href": "/api/audit/sessions/1/channels/4"
    },
    "body": {
      "type": "seamless",
      "verdict": "deny",
      "start_time": 1451902142,
      "end_time": 1451902142,
      "duration": 0
    }
  },
  {
    "key": "5",
    "meta": {
      "href": "/api/audit/sessions/1/channels/5"
    },
    "body": {
      "type": "dynamic virtual",
      "verdict": "accept",
      "start_time": 1451902145,
      "end_time": 1451902146,
      "duration": 1,
      "dynamic_channel": "Microsoft::Windows::RDS::Geometry::v08.01"
    }
  },
  {
    "key": "6",
    "meta": {
      "href": "/api/audit/sessions/1/channels/6"
    },
    "body": {
      "type": "custom",
      "verdict": "deny",
      "start_time": 1451902145,
      "end_time": 1451902145,
      "duration": 0
    }
  },
  {
    "key": "7",
    "meta": {
      "href": "/api/audit/sessions/1/channels/7"
    },
    "body": {
```

```
      "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:"
```

```
    }
  },
  {
    "key": "11",
    "meta": {
      "href": "/api/audit/sessions/1/channels/11"
    },
    "body": {
      "type": "scard redirect",
      "verdict": "accept",
      "start_time": 1451902149,
      "end_time": 1451902150,
      "duration": 1,
      "device_name": "SCARD"
    }
  }
```


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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 714. 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).

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/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 "Replaying 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`.

- `?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.

- `?q`

  Filter the list using one or more properties (elements) of the 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:

```
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:

```
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:

```
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": "2017-01-25T11:11:52.000+01:00"
			},
			"key": "2",
			"meta": {
				"href": "/api/audit/sessions/2"
			}
		},
		{
			"body": {
				"duration": 34,
				"name": "myname",
				"start_time": "2017-01-25T11:11:11.000+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
	}
```


Element	Type	Description
`items`	list	Top level element, a list containing the details of the matching sessions.

Element	Type	Description
body	JSON object	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: `"body": { "duration": 0, "name": null, "start_time": "2017-01-25T11:11:52.000+01:00" },`
key	string	A globally unique string that identifies the session. This session ID has the following format: *svc/<unique-random-hash>/<name-of-the-connection-policy>:<session-number-since-service-started>/<protocol>*, for example, *svc/5tmEaM7xdNi1oscgVWpbZx/ssh_console:1/ssh*. Log messages related to the session also contain this ID. For example: `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'` 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.

In addition to the usual meta elements of other endpoints, search results can contain the following additional elements.

Element	Type	Description
meta	JSON	Top level element, a list containing meta information

Element	Type	Description
	object	about the response.
`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 `?q=screen.content` option to search for keywords that appear in the screen 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 entered 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.

```
?q=screen.content:"my-search-expression"
```

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 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.

- Do not begin the expression with the `*` wildcard.

Examples:

Search for the word `example`

```
?q=screen.content:example
```

Search for the words `example`, `examples`, and so on:

```
?q=screen.content:example%3F
```

Search for the words `example`, `examine`, and so on:

```
?q=screen.content:exam%2A
```

Search in metadata and session content at the same time:

```
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.

```
?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.

```
?fields=protocol,name
```

To list all possible elements, use the `fields=*` expression.

```
?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.

```
?start=2017-01-25T11:11:52.000+01:00
?end=2017-01-25T11:41:52.000+01:00
?start=2017-01-24&end=2017-01-25
```

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:

```
?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
```

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:


```
{
  "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,
```

```
    "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:

```
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:

```
{
  "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",
```

```
    "analytics": {
      "interesting_events": ["sudo", "systemctl"],
      "similar_sessions": []
    },
    "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?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-paKzcMJwXghEFJ9UvsdqFU-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

Cookie name	Description	Required	Values
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 more information 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 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:

```
{
        "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
                }
 }
```


Element	Type	Description
`items`	list	Top-level element, a list containing the details of the matching session.
`channel.id`	integer	A reference to the ID of the channel in the session where the event occurred.
`end_time`	string	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.
`rank`	float	Indicates the relevance of the match. If there are several results, the order of them is based on their relevance.
`record_id`	integer	The content element's exact position in the audit trail file.
`begin`	integer	The identifier of the screenshot in the

Element		Type	Description
			audit trail file where the content element first appeared.
	`end`	integer	The identifier of the screenshot in the audit trail file where the content element last appeared.
	`for_screenshot`	integer	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.
`start_time`		string	The timestamp of when the content first appeared on the screen and recording started. 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.
`trail_id`		integer	The unique identifier of the trail that contains the event.

In addition, search results can contain the usual meta elements of other endpoints:

Element	Type	Description
`meta`	JSON object	Top-level element, a list containing meta information about the response. For details about the type of information returned, see Message format on page 10.

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:

   ```
   "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

Cookie name	Description	Required	Values
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 more information 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 retrieves statistical data about sessions.

```
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:

```
...
{
   "Alpha": {
      "protocol": "http"
   },
   "Bravo": {
      "protocol": "ssh"
   },
   "Charlie": {
      "protocol": "rdp"
   },
   "Delta": {
      "protocol": "rdp"
   },
   "Echo": {
      "protocol": "rdp"
   },
   "Foxtrot": {
      "protocol": "http"
   },
   "Golf": {
      "protocol": "http"
   }
}
...
```

The response contains:

```
...
{
  "meta": {
      "href": "/api/audit/sessions/stats",
      "parent": "/api/audit/sessions",
      "others": 1,
      "field": "protocol",
      "size": 2
  }
}
...
```

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.

```
...
[
  {"count": 3, "value": "http"},
  {"count": 3, "value": "rdp"}
...
```

- `?start`: 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`.
- `?q=screen.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 10000. When this limit is exceeded, the scope of statistics is limited to the first 10000 sessions.

Response

The following snippet is a sample response received when retrieving statistics about the `protocol` field.

For more information on 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 .

```
{
      "items": [
            {
                  "count": 7,
                  "value": "ssh"
            }
      ],
      "meta": {
            "field": "protocol",
            "href": "/api/audit/sessions/stats",
            "others": 0,
            "parent": "/api/audit/sessions",
            "remaining_seconds": 600,
            "size": 10
      }
}
```

Element	Type	Description
`body`, or `items` when a	Top-level element (string)	Contains the properties that are in the scope of the requested statistics.

Element		Type	Description
list is returned			
	`count`	integer	Indicates the number of sessions included in the scope of statistics.
	`value`	string	Contains the value of the field that you requested statistics about.
`meta`		Top-level element	Contains links to different parts of the REST service.
	`field`	string	Contains the name of the field that you requested statistics about.
	`sub_fields`	string	Contains the name of the sub field that you requested statistics about.
	`others`	integer	Some values of the field that you specified in your query are not included in the scope of statitics. 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`.
	`size`	integer	The size that you specified in your query.

Example 1:

If you query `"/api/audit/sessions/stats?field=protocol"`, and the following sessions were recorded:

```
...
{
  "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-`

02&end=2017-01-03&q=psm.content_reference_id%3A%5B3%20TO%206%5D", and the following sessions were recorded:

```
{
  "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:

```
| trail_id | rank   | hits_count |
|    1     | 1.555 | 1          |
|    2     | 1.555 | 1          |
|    3     | 1.555 | 1          |
|    4     | 1.555 | 1          |
|    6     | 1.555 | 1          |
```

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"}
...
```

Example 3:

If you query `"/api/audit/sessions/stats?field=user.gateway_username&?sub_fields=protocol&?size=1"`, and the following sessions were recorded:

```
...
{
  "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": [
         {
            "count": 3,
            "value": "ssh"
         }
      ],
      "count": 5,
      "others": 2,
      "value": "user-Alpha"
   }
]
...
```

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 41.

Code	Description	Notes
200	OK	The query was well-formed and statistics have been successfully retrieved.
400	InvalidQueryValue	The query is invalid, for example, it has an invalid value.
500	SearchUnavailable	The search backend is inaccessible.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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 .

```
{
"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":
```

```
62 },
        { "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 }
    ]
},

"key": "histogram",
"meta":
    {
        "bin_size": "month",
        "field": "recording.connection_policy",
        "href": "/api/audit/sessions/histogram",
        "parent": "/api/audit/sessions",
        "remaining_seconds": 599,
        "time_zone": "Etc/UTC",
        "size": "10"
    }
}
```


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.

Element	Type	Description
`end`	date	The date set in the `?end` parameter of the request. By default, this is the date of the request.
`start`	date	The date set in the `?start` parameter of the request. By default, this is one month before the date of the request.
`time_zone`	string	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 .
`size`	integer	The size that you specified in your query.

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 41.

Code	Description	Notes
200	OK	The query was well-formed and the histogram has been successfully retrieved.
400	TooMuchBucketsInResult	Using the requested `bin_size` would result in too many intervals for the queried period.
400	NotSupportedContentOption	This endpoint does not support filtering in the content of sessions.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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",
```

```
            "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
    }
}
```


Element		Type	Description
`items`		list	Top level element, a list containing the alerts of the session.
	`alert_type`	string	The type of the event that triggered the alert. Possible values: • `adp.event.command`: A command entered in SSH or Telnet. • `adp.event.screen.content`: Alert triggered by the screen content. • `adp.event.screen.creditcard`: Credit card numbers detected. Displayed only as an alert, not visible in the events. • `adp.event.screen.windowtitle`: The title of the window in graphic protocols.
	`channel_id`	string	The regular expression that matched the command line without prompt.
	`matched_action`	integer	A reference to the ID of the channel in the session where the event occurred.

Element	Type	Description
`matched_content`	text	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 `sudo` command, then the `psm.alerts.matched_content` value contains the entire command line, including the command prompt, for example, `myuser@examplehost:~$ man sudo`
`matched_regexp`	text	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.
`record_id`	integer	The ID number of the alert within the session.
`rule_name`	string	The name of the content policy rule that triggered the alert. Note that this is not the name of the Content Policy.
`time`	string	The timestamp when the alert was triggered, for example, `2017-04-25T13:26:39.144356`.

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
```

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 761.

URL

```
GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/events
```

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
    "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,
            "time": "2017-04-11T09:21:21.04",
            "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",
```

```
            "channels_id": "0",
            "content": "myuser@examplehost:/var$ man sudo",
            "record_id": 148,
            "time": "2017-04-11T09:21:27.017",
            "type": "command"
        }
    ],
    "meta": {
        "first":
"/api/audit/sessions/7930f4308efe8aecd710202d815b76ff/events?limit=500&offset=
0",
        "href": "/api/audit/sessions/7930f4308efe8aecd710202d815b76ff/events",
        "last":
"/api/audit/sessions/7930f4308efe8aecd710202d815b76ff/events?limit=500&offset=
0",
        "limit": 500,
        "next": null,
        "offset": 0,
        "parent": "/api/audit/sessions/7930f4308efe8aecd710202d815b76ff",
        "previous": null
    }
}
```


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

Element	Type	Description
		protocols.
`time`	string	The timestamp when the event occurred, for example, `2017-04-25T13:26:39.144356`.

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=screen.content:sudo
```

You can escape special characters using the backslash character.

```
?q=screen.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=screen.content:(sudo rm)
```

You can use the `*` (asterisk) and `?` (question mark) wildcards for string-type values.

```
?q=screen.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.
- `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`: Text that appears on the screen in the session.
  - `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 761.

URL

```
GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/indexers
```

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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":
"/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": 500,
        "match_count": 1,
        "next": null,
```

```
        "offset": 0,
        "parent": "/api/audit/sessions/rUhhQZ3jYsY1NDWYp9DEpq",
        "previous": null,
        "remaining_seconds": 599
    }
}
```

Element	Type	Description
`items`	list	Top level element, a list containing the indexing-related details of the session. For details, see `indexer_info` section in"List of available search queries" in the Administration Guide.

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 761.

URL

```
GET https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/trail_downloads
```

Cookies

Cookie name	Description	Required	Values
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 more information 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 indexing-related details of a session.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
       "items": [
             {
                   "from_api": false,
                   "ip_address": "10.20.30.40",
                   "time": "2018-11-20T11:10:19.000Z",
                   "username": "admin"
             },
             {
                   "from_api": false,
                   "ip_address": "10.20.30.40",
                   "time": "2018-11-20T11:10:38.000Z",
                   "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/rUhhQZ3jYsY1NDWYp9DEpq",
             "previous": null,
             "remaining_seconds": 599
       }
 }
```


Element		Type	Description
`items`		list	Top level element, a list containing the indexing-related details of the session.
	`from_api`	boolean	True, if the audit trail was not downloaded from the GUI, but through SOAP or REST API.
	`ip_address`	string	The IP address of the client that downloaded the audit trail.
	`time`	boolean	The exact time when the user downloaded the audit trail file.
	`username`	string	The user name of the user who downloaded the audit trail.

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
```

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

Cookie name	Description	Required	Values
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 more information 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 configuration options.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
    "body": {
        "decryption_keys": [
            {
                "key": "e38d47bd-5374-4d7c-b683-e26ea77142e2",
                "meta": {
                    "href": "/api/configuration/x509/e38d47bd-5374-4d7c-b683-
e26ea77142e2"
                }
            }
        ],
```

```
        "number_of_workers": 1,
        "remote_access": {
            "enabled": false
        },
        "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"
    }
}
```


A sample response when external indexers are enabled:


```
{
    "body": {
        "decryption_keys": [],
        "number_of_workers": 1,
        "number_of_workers": 0,
        "remote_access": {
            "access_restriction": {
                "allowed_from": [
                    "10.40.0.0/16"
                ],
                "enabled": true
            },
            "enabled": true,
            "listen": [
                {
                    "address": {
                        "key":
"nic1.interfaces.ff7574025754b3df1647001.addresses.1",
                        "meta": {
                            "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/address
es/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"
    }
}
```


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 indexer service.
decryption_keys	list	Indexing encrypted audit trails requires the

Element	Type	Description
		X.509 certificates and the matching private keys. The certificates must 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 280.
key	reference	The ID of the referenced decryption key. You can upload private keys at the `/api/configuration/private_key` endpoint. For details, see Private keys stored on SPS on page 280.
number_of_near_realtime_workers	integer	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	integer	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 requires about 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	JSON object	Enables external indexers to access the SPS host, and configures access restrictions and other parameters.
selection	string	The value of this option must be `integrated`.

Element	Type	Description
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`.
enabl	boole	Set it to `true` to restrict access to the specified client

Element		Type	Description
	ed	an	addresses.
enabled		boole-an	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 "Configuring external indexers" in the Administration Guide. **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.**
listen		list	Selects the network interface, IP address, and port where the clients can access the web interface.
	addre ss	JSON object	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:

```
{
    "body": {
        "interfaces": {
            "@order": [
                "ff7574025754b3df1647001"
            ],
            "ff7574025754b3df1647001": {
                "addresses": {
                    "1": "10.40.255.171/24",
                    "@order": [
                        "1"
                    ]
                },
                "name": "default",
                "vlantag": 0
            }
        },
```

Element	Type	Description

```
        "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.

```
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.in-
terfaces.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.in-
terfaces.ff7574025754b3df1647001.addresses.1",
    "meta": {
        "href": "/api/-
config-
uration/net-
work/n-
ics/n-
ic1#interfaces/ff7574025754b3df1647001/addresses/1"
```

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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:

```
{
        "decryption_keys": ["216b33dd-a1cd-41b1-85c5-66290b7a043d"],
        "number_of_near_realtime_workers": 0,
        "number_of_workers": 2,
        "remote_access": {
              "access_restriction": {
                    "allowed_from": [
                          "10.40.0.0/16"
                    ],
                    "enabled": true
              },
              "enabled": true,
              "listen": [
                    {
                          "address":
"nic1.interfaces.ff7574025754b3df1647001.addresses.1",
                          "port": 12354
                    }
              ],
              "ssl_config": {
                    "ca": "773ed50d-3066-44f1-84ec-cbef59111702",
                    "service": "a8b6c791-c24a-466d-ac50-a425a5253d46",
                    "worker": "c54c436f-63c5-4a2e-a59e-7ad904bbf0f2"
                    }
        },
        "selection": "integrated"
 }
```


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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
       "items": [
             {
                   "key": "-50000",
                   "meta": {
                         "href": "/api/configuration/policies/indexing/-50000"
                   }
             },
             {
                   "key": "13442970955825a89b55e46",
                   "meta": {
                         "href":
"/api/configuration/policies/indexing/13442970955825a89b55e46"
                   }
             }
       ],

       "meta": {
             "first": "/api/configuration/policies/audit_policies",
             "href": "/api/configuration/policies/indexing",
             "last": "/api/configuration/policies/usermapping_policies",
             "next": "/api/configuration/policies/ldap_servers",
             "parent": "/api/configuration/policies",
             "previous": "/api/configuration/policies/credentialstores",
             "remaining_seconds": 599,
             "transaction": "/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"
    }
}
```


Element	Type	Description
`key`	string	Top level element, contains the ID of the policy.
`body`	Top level element (string)	Contains the configuration options of the indexer policy.
`index`	Top level element	Contains the indexed events of the indexer policy. Possible values: • `command`: A command entered in SSH or Telnet. • `keyboard`: Keyboard-related events, for example, pressing Enter. • `mouse`: Mouse-related events, for example, mouse clicks. • `screen_content`: Screen content elements, for example, commands, window titles, IP addresses, user names, and so on. • `window_title`: The title of the window in graphic protocols.
`name`	string	The name of the indexer policy.
`ocr`	JSON object	Configuration of the OCR engine.

Element		Type	Description
	accuracy	string	Accuracy level for Optical Character Recognition. Possible values: • fast: 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. • balanced: 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. • accurate: 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.
	custom_languages	Top level element	Configures what languages to detect.

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	Type	Description

		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.
		
		The following languages are supported: *English*: `eng`, *German*: `deu`, *French*: `fra`, *Dutch*: `nld`, *Norwegian*: `nor`, *Swedish*: `swe`, *Finnish*: `fin`, *Danish*: `dan`, *Icelandic*: `isl`, *Portuguese*: `por`, *Spanish*: `spa`, *Catalan*: `cat`, *Galician*: `glg`, *Italian*: `ita`, *Maltese*: `mlt`, *Greek*: `ell`, *Polish*: `pol`, *Czech*: `ces`, *Slovak*: `slk`, *Hungarian*: `hun`, *Slovenian*: `slv`, *Croatian*: `hrv`, *Romanian*: `ron`, *Albanian*: `sqi`, *Turkish*: `tur`, *Estonian*: `est`, *Latvian*: `lav`, *Lithuanian*: `lit`, *Esperanto*: `epo`, *Serbian(Latin)*: `qsl`, *Serbian*: `srp`, *Macedonian*: `mkd`, *Moldavian*: `mol`, *Bulgarian*: `bul`, *Byelorussian*: `bel`, *Ukrainian*: `ukr`, *Russian*: `rus`, *Chechen*: `che`, *Kabardian*: `kbd`, *Afrikaans*: `afr`, *Aymara*: `aym`, *Basque*: `eus`, *Bemba*: `bem`, *Blackfoot*: `bla`, *Breton*: `bre`, *Brazilian*: `qbp`, *Bugotu*: `bgt`, *Chamorro*: `cha`, *Tswana(Chuana)*: `tsn`, *Corsican*: `cos`, *Crow*: `cro`, *Eskimo*: `qes`, *Faroese*: `fao`, *Fijian*: `fij`, *Frisian*: `fry`, *Friulian*: `fur`, *Gaelic(Irish)*: `gle`, *Gaelic (Scottish)*: `gla`, *Ganda(Luganda)*: `lug`, *Guarani*: `grn`, *Hani*: `hni`, *Hawaiian*: `haw`, *Ido*: `ido`, *Indonesian*: `ind`, *Interlingua*: `ina`, *Kasub*: `csb`, *Kawa*: `wbm`, *Kikuyu*: `kik`, *Kongo*: `kon`, *Kpelle*: `kpe`,

Custom languages elements	Type	Description
		Kurdish: `kur`, *Latin*: `lat`, *Luba*: `lua`, *Luxembourgish*: `ltz`, *Malagasy*: `mlg`, *Malay*: `msa`, *Malinke*: `mlq`, *Maori*: `mri`, *Mayan*: `MYN`, *Miao*: `hmn`, *Minangkabau*: `min`, *Mohawk*: `moh`, *Nahuatl*: `NAH`, *Nyanja*: `nya`, *Occidental*: `ile`, *Ojibway*: `oji`, *Papiamento*: `pap`, *PidginEnglish*: `tpi`, *Provencal*: `oci`, *Quechua*: `que`, *Rhaetic*: `roh`, *Romany*: `rom`, *Rwanda*: `kin`, *Rundi*: `run`, *Samoan*: `smo`, *Sardinian*: `srd`, *Shona*: `sna`, *Sioux*: `dak`, *Sami*: `SMI`, *Sami(Lule)*: `smj`, *Sami(Northern)*: `sme`, *Sami (Southern)*: `sma`, *Somali*: `som`, *Sotho*: `sot`, *Sundanese*: `sun`, *Swahili*: `swa`, *Swazi*: `ssw`, *Tagalog*: `tgl`, *Tahitian*: `tah`, *Tinpo*: `qti`, *Tongan*: `ton`, *Tun*: `tug`, *Visayan*: `qis`, *Welsh*: `cym`, *Sorbian (Wend)*: `WEN`, *Wolof*: `wol`, *Xhosa*: `xho`, *Zapotec*: `zap`, *Zulu*: `zul`.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.
409	Conflict	No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction on page 32.

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 32.

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:

```
{
    "key":  "aa423b72-0d0f-4275-be30-494e9a99ffad",
    "meta":  {
       "href":  "/api/configuration/policies/indexing/aa423b72-0d0f-4275-
be30-494e9a99ffad",
       "parent":  "/api/configuration/policies/indexing",
       "transaction":  "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

15

Reporting

Reporting

List of endpoints for configuring reporting, and accessing the generated reports.

CAUTION:

Creating statistics from custom queries using the Reporting > View & edit subchapters > Advanced statistics page and the `/api/configuration/reporting/custom_subchapters` REST API endpoint has been deprecated in version 7.0.0. During the upgrade process, existing advanced statistics subchapters and their references are removed from the SPS configuration. Additionally, advanced statistics ACLs assigned to user groups are also removed from the SPS configuration. Consider that if a user group only had the advanced statistics ACL assigned under Users & Access Control > Appliance Access, the whole ACL entry is removed during the upgrade process.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information on authentication, see Authenticate to the SPS REST API on page 19.

Cookie name	Description	Required	Values
			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/reporting
```

Response

The following is a sample response received when listing the available endpoints.

For more information on the `meta` object, see Message format on page 10.

```
{
  "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": "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
content_subchapters	List of the reporting subchapters created from audit trail content (statistics of search keywords, and screenshots).
predefined_reports	List of the pre-defined reports available on SPS.
reports	List of the configured reports.
statistics_subchapters	List of the reporting subchapters created from connection statistics.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Reports

List of the configured reports.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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": [
       {
          "key": "7798770004e472c8576912",
          "meta": {
             "href":
"/api/configuration/reporting/reports/7798770004e472c8576912"
          }
       },
       {
          "key": "8292675195707f19d932af",
          "meta": {
             "href":
"/api/configuration/reporting/reports/8292675195707f19d932af"
          }
       }
    ]
 }
```


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",
    "last": "/api/configuration/reporting/reports/8292675195707f19d932af",
    "next": null,
    "parent": "/api/configuration/reporting/reports",
    "previous": "/api/configuration/reporting/reports/12046247915707e5d6a5c59",
    "transaction": "/api/transaction"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the report
body	Top level element (string)	The elements of the report.
access	list	Required. List of access control groups whose members can access the subchapter. To deny access to the report, use "admin" as the only value for the element.
chapters	Top level item	A chapter of the report.
email_recipients	Top level item	Contains the list of e-mails where the generated report is sent.
recipients	list	Custom list of e-mails where the generated report is sent. To use a custom list, the selection element must be set to other.

Element			Type	Description
		`selection`	string	This element can have two values: • `default` uses the e-mail address configured in the `reporting_address` element of the https://<IP-address-of-SPS>/api/configuration/management/email endpoint (or the **Basic Settings > Management > Mail settings > Send reports to** field on the web UI). • `other` uses the e-mails listed in the `recipients` element.
	`frequency`		Top level item	Contains the list of options for defining the frequency of generating the report.
		`day`	boolean	Set it to `true` to generate the report each day.
		`month`	boolean	Set it to `true` to generate the report each month.
		`week`	boolean	Set it to `true` to generate the report each week.
	`logo_id`		string	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 > Configuration > <report> > Choose new logo** button.
	`name`		string	The name of the report.
	`send_report_in_email`		boolean	Set it to `false` if you do not want to include the generated report in the e-mail.

Chapters elements	Type	Description
`name`		string	Name of the chapter.
`subchapters`		list	List of subchapters included in the chapter.
	`name`	string	Name of the built-in subchapter included in the chapter. For the list of the built-in subchapters, see Built-in subchapters on page 790.

Chapters elements	Type	Description	
			To include a built-in subchapter, use the value of its `name` element, not the key.
	reference	string	The `key` of the custom, content, or statistics subchapter. • For the keys of the reporting subchapters created from custom queries to the SPS connection database, see the `custom_subchapters` endpoint. • For the keys of the reporting subchapters created from audit trail content (statistics of search keywords, and screenshots), see the `reporting/content_subchapters` endpoint. • For the keys of the reporting subchapters created from connection statistics, see the `reporting/statistics_subchapters` endpoint. To include a custom, content, or statistics subchapter, use the value of its `key` element, not the name.
	`selection`	string	This element can have two values: • Set `builtin` for the default reporting subchapters. • Set `custom` for all other subchapters (custom, content or statistics).

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"
}
```

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
400	IncompleteConfigurationSubtreeError	Possible cause: PUT operation on the `reports` endpoint, instead of POST.
400	IncompleteConfigurationSubtreeError "missing_paths": ["email_recipients/recipients"]	You have selected `other` for the `selection` element under `email_recipients`, but did not provide a list using `recipients`.
400	IncompleteConfigurationSubtreeError `Syntax error: \"No such property; property='recipients'`	Do not provide `recipients` if you set the `selection` element under `email_recipients` to `default`.
400	IncompleteConfigurationSubtreeError "missing_paths": ["chapters/7/subchapters/0/name"]	Verify that the `selection` element of the subchapter is correctly set to `builtin` or `custom`.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Add a report

To add a report, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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.

```
{
    "key": "26ddf648-9a21-4a7f-af56-9cea575785a9",
    "meta": {
       "href": "/api/configuration/reporting/reports/26ddf648-9a21-4a7f-af56-9cea575785a9",
       "parent": "/api/configuration/reporting/reports",
       "transaction": "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

Modify a report

To modify a report, you have to:

1. **Open a transaction.**

   For more information, see Open a transaction on page 32.

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 35.

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 value of `key`.

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:

   ```
   {
       "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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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/content_subchapters",
    "transaction": "/api/transaction"
  },
  "items": [
    {
      "key": "pcidss",
      "meta": {
        "href": "/api/configuration/reporting/predefined_reports/pcidss"
      }
    }
  ]
}
```

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
    }
 }
```

Element	Type	Description
key	string	Top level element, contains the ID of the report.
<id-of-the-report>	Top level item	The elements of the pre-defined report.
access	list	List of access control groups whose members can access the report.

Element		Type	Description
email_recipients		Top level item	Contains the list of e-mails where the generated report is sent.
	recipients	list	Custom list of e-mails where the generated report is sent. To use a custom list, the `selection` element must be set to `other`.
	selection	string	This element can have two values: • `default` uses the e-mail address configured in the `reporting_address` element of the `https://<IP-address-of-SPS>/api/configuration/management/email` endpoint (or the **Basic Settings > Management > Mail settings > Send reports to** field on the web UI). • `other` uses the e-mails listed in the `recipients` element.
name		string	The name of the report.
send_report_in_email		boolean	Set it to `false` if you do not want to include the generated report in the e-mail.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
400	IncompleteConfigurationSubtreeError `Syntax error: \"No such property; property='recipients'`	Do not provide `recipients` if you set the `selection` element under `email_recipients` to `default`.
400	Bad Request "message": "New Ids are not allowed"	Error when committing your transaction. Creating new pre-defined reports is not allowed.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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 35.

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

Cookie name	Description	Required	Values
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 more information 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 available content subchapters.

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

The following command retrieves the properties of a specific subchapter.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
    "meta": {
       "first": "/api/configuration/reporting/content_subchapters",
       "href": "/api/configuration/reporting/content_subchapters",
       "last": "/api/configuration/reporting/statistics_subchapters",
       "next": "/api/configuration/reporting/predefined_reports",
       "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.

```
{
  "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"
    ]
  },
  "key": "13869311625707e0a3e0892",
  "meta": {
    "first": "/api/configuration/reporting/content_
subchapters/13869311625707e0a3e0892",
    "href": "/api/configuration/reporting/content_
subchapters/13869311625707e0a3e0892",
    "last": "/api/configuration/reporting/content_
subchapters/13869311625707e0a3e0892",
    "next": null,
    "parent": "/api/configuration/reporting/content_subchapters",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the subchapter.
body	Top level element (string)	The elements of the subchapter.
access	list	Required. List of access control groups whose members can access the subchapter. To deny access to the subchapter, use `"admin"` as the only value for the element.
filter	Top level element.	Filter options for narrowing the scope of the keyword search. See the corresponding table for more details.
channel_policy	string	References the key of the channel policy. You can configure channel policies at the `"/api/configuration/<protocol>/channel_policies/<policy-ID>"` endpoint. Note that the path is different for each protocol.

Element		Type	Description
			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).
	`connection_policy`	string	The key of the connection policy specified for the search. To use a connection policy, you must also set the protocol using the `protocol` element.
	`protocol`	string	The protocol of the connection or channel policy specified for the search.
	`server_address`	string	The target server's address. Use an IPv4 address.
	`server_port`	int	The port of the target server's address.
	`source_address`	string	The address from where the connection is initiated.
	`source_port`	int	The port of the address from where the connection is initiated.
	`username`	string	The username used to connect to the target server.
`name`		string	The name of the subchapter.
`search_words`		list	The list of search keywords to generate statistics and screenshots for in the subchapter.

Examples:

Create a content subchapter for the occurences 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.

```
{
  "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
  },
  "name": "Shell_access",
  "search_words": [
    "logout"
  ]
}
```

- 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.

```
{
    "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.

```
{
  "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"
  },
  "name": "Login_as_admin",
  "search_words": [
    "logout"
  ]
}
```

- Search connections made to a specific server address and port.

```
{
    "access": [
       "search",
       "report"
    ],
    "filter": {
       "channel_policy": null,
       "connection_policy": null,
       "protocol": "ssh",
       "server_address": "<server-ip>",
       "server_port": <port>,
       "source_address": null,
       "source_port": null,
       "username": null
    },
    "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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.

Code	Description	Notes
400	Path: <endpoint>/filter/channel_policy Type: SyntacticError	You have included the `key` and `meta` elements of a `channel_policy` in a PUT or POST request.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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...

     ```
     "channel_policy": "-10000"
     ```

     ...and the `protocol` element:

     ```
     "protocol": "ssh"
     ```

If the POST request is successful, the response includes the key of the new subchapter. For example:

```
{
    "key": "416bb324-b44e-4ed3-a49d-02e99e53e941",
    "meta": {
       "href": "/api/configuration/reporting/content_subchapters/416bb324-
b44e-4ed3-a49d-02e99e53e941",
       "parent": "/api/configuration/reporting/content_subchapters",
       "transaction": "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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:

```
"channel_policy": {
    "key": "-10200",
    "meta": {
       "href": "/api/configuration/ssh/channel_policies/-10200"
    }
 }
```

   You have to change it in your PUT request to:

```
"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 35.

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

Cookie name	Description	Required	Values
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 more information 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 available subchapters.

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

The following command retrieves the properties of a specific subchapter.

```
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 more information on the `meta` object, see Message format on page 10.

```
{
    "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": [
       {
          "key": "21111736175707f1df8bea1",
          "meta": {
             "href": "/api/configuration/reporting/statistics_
subchapters/21111736175707f1df8bea1"
          }
       }
    ]
 }
```

When retrieving the endpoint of a specific subchapter, the response is the following.

```
{
  "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"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the subchapter.
body	Top level element (string)	The elements of the subchapter.
access	list	Required. List of access control groups whose members can access the subchapter. To deny access to the subchapter, use "admin" as the only value for the element.
chart	Top level element	Defines the properties of the chart generated from the database query.
type	string	Defines the chart type. • Use bar to generate a bar chart.

Element	Type	Description
		• Use `pie` to generate pie a chart. • Use `list` to generate a list.
`name`	string	The name of the subchapter.
`query`	string	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 714.

Examples:

Create statistics about the 15 most common usernames used in SSH connections.

- Create a bar chart.

```
{
   "access": [
      "reporting",
      "search"
   ],
   "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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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:

```
{
    "key": "769e627d-515d-4d26-a03e-cb2ed0bbee04",
    "meta": {
       "href": "/api/configuration/reporting/statistics_
subchapters/769e627d-515d-4d26-a03e-cb2ed0bbee04",
       "parent": "/api/configuration/reporting/statistics_subchapters",
       "transaction": "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

Health and maintenance

Monitor appliance health status

To monitor the health status of an appliance, query the `/api/health_status` endpoint.

URL

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

Response

The following is a sample response received.

For more information on 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": 1.0,
                    "disk": 2.0,
                    "firmware": {
                        "tainted_files": {
                            "boot": 1,
                            "core": 2,
                            "ha_other": 0
                        },
                        "integrity": {
                            "boot": "OK",
                            "core": "CORRUPTED"
                        }
                    },
                    "indexer": {
                        "indexer_status": [
                            {
                                "capabilities": ["index"],
                                "idle_workers_count": 1,
                                "indexer_jobs": 0,
                                "ip": "127.0.0.1",
                                "screenshot_jobs": 0,
                                "video_jobs": 0,
                                "workers_count": 1
                            },
                            {
                                "capabilities": ["screenshot", "video"],
                                "idle_workers_count": 1,
                                "indexer_jobs": 0,
                                "ip": "127.0.0.1",
```

```
                "screenshot_jobs": 0,
                "video_jobs": 0,
                "workers_count": 1
            },
            {
                "capabilities": ["video"],
                "idle_workers_count": 1,
                "indexer_jobs": 0,
                "ip": "127.0.0.1",
                "screenshot_jobs": 0,
                "video_jobs": 0,
                "workers_count": 1
            }
        ],
        "processing_tasks": [],
        "remaining_tasks": [],
        "worker_warnings": []
    },
    "load1": 3.0,
    "load15": 4.0,
    "load5": 5.0,
    "memory": 7.0,
    "pipeline_queues": {
        "session_events": 64.3,
        "indexer_events": 40.0
    },
    "raid": {
        "disks": {
            "252:0": {"status": "OK"},
            "252:1": {"status": "OK"},
            "252:2": {"status": "OK"},
            "252:3": {"status": "OK"}
        },
        "status": {
            "detailed_status": "OK",
            "operation_progress": 0,
            "status": "OK"
        }
    },
    "sessions": {
        "http": 1,
        "ica": 2,
        "mssql": 3,
        "rdp": 4,
        "ssh": 5,
        "telnet": 6,
        "vnc": 7
    },
    "sessions_total": 28,
    "swap": 8.0,
    "system_details": {
        "cpu": {
            "guest": 0.0,
            "guest_nice": 0.0,
            "idle": 97.0,
            "iowait": 0.0,
            "irq": 0.0,
            "nice": 0.0,
            "softirq": 0.0,
```

```
                        "steal": 0.0,
                        "system": 2.0,
                        "user": 1.0
                    },
                    "disk": {
                        "free": 200000000,
                        "percent": 2.0,
                        "total": 10000000000,
                        "used": 9800000000
                    },
                    "memory": {
                        "active": 5523861504,
                        "available": 3288555520,
                        "buffers": 265920512,
                        "cached": 3269304320,
                        "free": 1188810752,
                        "inactive": 1109909504,
                        "percent": 60.6,
                        "shared": 1117671424,
                        "slab": 325619712,
                        "total": 8344498176,
                        "used": 3620462592
                    },
                    "swap": {
                        "free": 0,
                        "percent": 0.0,
                        "sin": 0,
                        "sout": 0,
                        "total": 0,
                        "used": 0
                    }
                }
            }
        }
```


Elements of the response message include:

Elements	Type	Description
`health_status`	null or object	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).
`health_status.memory`	floating point number	Memory usage (percent)
`health_status.disk`	floating point number	Hard disk usage (percent)
`health_status.swap`	floating	Swap usage (percent)

Elements	Type	Description
	point number	
`health_status.cpu`	floating point number	Overall CPU usage (percent)
`health_status.load1`	floating point number	The average system load during the last one minute.
`health_status.load5`	floating point number	The average system load during the last five-minute period.
`health_status.load15`	floating point number	The average system load during the last fifteen-minute period.
`health_status.sessions`	string	The protocol type and the number of ongoing sessions. For example: `"sessions": { "ssh": 3, "rdp": 4 }`
`health_status.total_sessions`	integer (number of)	The total number of ongoing sessions.
`health_status.system_details`	JSON object	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 One Identity Safeguard for Privileged Sessions without notice.
`health_status.pipeline_queues`	JSON object	Represents the fullness of the processing pipelines in percentages.
`health_status.indexer`	JSON object	
`health_status.indexer.indexer_`		Represents the current status of the indexer service.

Elements	Type	Description
status		
health_status.indexer.indexer_status capabilities	enum	Defines what various tasks the worker nodes of that group can handle. Possible values: index | screenshot | video
health_status.indexer.indexer_status idle_workers_count	number	The number of worker nodes without a task.
health_status.indexer.indexer_status indexer_jobs	number	The number of indexer tasks delegated to worker nodes within that worker group.
health_status.indexer.indexer_status ip	string	The IPv4 address of the indexer worker nodes.
health_status.indexer.indexer_status screenshot_jobs	number	The number of screenshot tasks delegated to worker nodes within that worker group.
health_status.indexer.indexer_status video_jobs	number	The number of video tasks delegated to worker nodes within that worker group.
health_status.indexer.indexer_status workers_count	number	The number of worker nodes with the same capabilities.
health_status.indexer.processing_tasks	array	The list of tasks in progress based on the connection policy.
health_status.indexer.remaining_tasks	array	The list of tasks queued for processing based on the connection policy. For example:

Elements	Type	Description
		[{ 'processing_tasks': { 'connection': 'ssh_connection', 'protocol': 'SSH', 'indexer_jobs': 1, 'screenshot_jobs': 3, 'video_jobs': 0 } }]
`health_status.indexer.worker_warnings`	array	The list of error and warning messages related to the worker nodes.
`health_status.raid`	JSON object	
`health_status.raid.status`	JSON object	
`health_status.raid.status status`	string	Possible values: • `OK [0]` - All disks are functioning properly. • `WARNING [1]` - All disks are functioning properly, however, there is a possibility of data loss, if the values increase further above the threshold. • `DEGRADED [2]` - There is no operation in progress due to disk failure. Data is preserved, but it is not at full redundancy. • `DEGRADED_SYNCING [3]` - Resync/rebuild is in progress due to disk failure. Data is preserved, but it is not at full redundancy. • `CRITICAL [4]` - Data loss occurred.
`health_status.raid.status detailed_status`	string	A short description of the type of RAID operation that is in progress (for example, rebuild, background

Elements	Type	Description
		initialization, or consistency check).
`health_status.raid.status operation_progress`	number	The progress of the RAID operation in percentages. Typically used if the disk is in a `REBUILDING` state. The default value is **`0`**.
`health_status.raid.disks`	JSON object	
`health_status.raid.disks <disk_id>`	JSON object	
`health_status.raid.disks <disk_id>.status`	string	Possible values: • `SPARE` - The Dedicated Hot Spare (DHS) disk that is on standby to take over, if a disk fails. • `REBUILDING` - Either a new disk has been inserted, or the DHS disk had to take over for a failed disk, and is now building. • `FAULTY` - The failed disk that must be replaced with a new one.

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.

16

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on 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": "1158115305570454488 3f77",
          "meta": {
             "href": "/api/configuration/policies/usermapping_
policies/1158115305570454488 3f77"
          }
       },
       {
          "key": "9328731525704545f5e3de",
          "meta": {
             "href": "/api/configuration/policies/usermapping_
policies/9328731525704545f5e3de"
          }
       }
    ]
 }
```


When retrieving the endpoint of a specific host key, the response is the following.

```
{
    "body": {
       "allow_other_remote_users_without_mapping": false,
       "mappings": [
          {
             "allowed_groups": [],
             "remote_user": "test"
          },
          {
             "allowed_groups": [
                "admins"
             ],
             "remote_user": "root"
          }
       ],
       "name": "Test"
    },
    "key": "9328731525704545f5e3de",
    "meta": {
       "first": "/api/configuration/policies/usermapping_
policies/277736452570454272e157",
       "href": "/api/configuration/policies/usermapping_
policies/9328731525704545f5e3de",
       "last": "/api/configuration/policies/usermapping_
policies/9328731525704545f5e3de",
       "next": null,
       "parent": "/api/configuration/policies/usermapping_policies",
       "previous": "/api/configuration/policies/usermapping_
policies/1158115305570454488 3f77",
       "transaction": "/api/transaction"
    }
 }
```


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

Element		Type	Description
			remote server.
mappings		Top level list	Contains the list of user groups and the corresponding remote usernames the group members can use to log in.
	`allowed_groups`	list	The usergroups allowed to log in as the `remote_user` on the remote server. Required element. Empty means all users.
	`remote_user`	string	The username on the remote server that the users configured in `allowed_groups` can use to log in. Required element. Must have a value.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
401	Unauthenticated	The requested resource cannot be retrieved because the

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	NotFound	The requested object does not exist.

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 32.

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:

   ```
   {
       "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 35.

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 32.

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 35.

Plugins

Contains the endpoints for configuring plugins.

URL

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

Cookies

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
	"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
`aa`	Endpoint for configuring authentication and authorization plugins.
`configuration_sync`	Endpoint for configuring plugins that synchronize the configuration of SPS clusters that receive their configuration from the Central Management node.
`credentialstore`	Endpoint for configuring credential store plugins.
`signingca`	Endpoint for configuring plugins to sign certificates.

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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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 32.

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:

```
{
       "meta": {
              "href": "/api/configuration/plugins/aa/aa423b72-0d0f-4275-
be30-494e9a99ffad",
              "parent": "/api/configuration/plugins/aa"
       },
       "key": "aa423b72-0d0f-4275-be30-494e9a99ffad",
       "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 35.

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 41.

Code	Description	Notes
201	Created	The plugin has been successfully uploaded. The response should include the key of the created object.
400		The plugin does not support this version of SPS.
400	InvalidPlugin	The type or some other value in the Manifest file of the plugin is invalid, or this version of SPS

Code	Description	Notes
		does not support this type of plugin. Check the `error` key in the response for details.
401	Unauthenticated	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.
422	TransactionProcessingError	The plugin was uploaded but deploying the plugin failed for some reason.

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 32.

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 47. If the DELETE request is successful, the response includes only the meta object, for example:

   ```
   {
         "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 35.

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 41.

Code	Description	Notes
200		The plugin has been successfully deleted
400	SemanticError	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).
401	Unauthenticated	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.
404		There is no plugin with the given key.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
    "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"
       }
    }
 }
```

Element			Type	Description
body			Top level element (string)	Contains the results of the response.
	zip_checksum		string	The checksum of the uploaded .zip file.
		verdict	string	The verdict of the integrity check.
		reason	string	The reason of the integrity check verdict.
	zip_content		string	The content of the .zip file.
		verdict	string	The verdict of the integrity check.
		reason	string	The reason of the integrity check verdict.
	unregistered		string	Whether SPS was joined to Starling for online checksum.
		verdict	string	The verdict of the integrity check.
		reason	string	The reason of the integrity check verdict.

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 41.

Code	Description	Notes
400	MissingMandatoryParameter	One of the following keys is missing: key, plugin_type, ops.
400	InvalidFormat	The key is not valid plugin key.
404	MissingPlugin	The plugin is not found in the configuration.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
  "items": [
    {
      "key": "2080160955734bb2a1ddf9",
      "meta": {
        "href": "/api/configuration/plugins/aa/2080160955734bb2a1ddf9"
      }
```

```
    }
  ],
  "meta": {
    "first": "/api/configuration/plugins/aa",
    "href": "/api/configuration/plugins/aa",
    "last": "/api/configuration/plugins/ticketing",
    "next": "/api/configuration/plugins/credentialstore",
    "parent": "/api/configuration/plugins",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```


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": "2080160955734bb2a1ddf9",
    "meta": {
       "first": "/api/configuration/plugins/aa/2080160955734bb2a1ddf9",
       "href": "/api/configuration/plugins/aa/2080160955734bb2a1ddf9",
       "last": "/api/configuration/plugins/aa/2080160955734bb2a1ddf9",
       "next": null,
       "parent": "/api/configuration/plugins/aa",
       "previous": null,
       "transaction": "/api/transaction"
    }
 }
```


Element	Type	Description
key	string	Top level element, contains the ID of the plugin.
body	Top level element (string)	Contains the properties of the plugin.

Element	Type	Description
`api`	string	The API version of the plugin.
`description`	string	The description of the plugin. This description is also displayed on the SPS web interface.
`default_configuration`	string	The default configuration of the plugin (an INI file as a string). For details, see the documentation of the particular plugin.
`entry_point`	string	The entry point of the plugin, for example, `main.py`
`name`	string	The name of the plugin. This name is also displayed on the SPS web interface. It cannot contain whitespace.
`path`	string	The path where the plugin is stored on SPS.
`scb_max_version`	string	The version number of the latest SPS release that is compatible with the plugin.
`scb_min_version`	string	The version number of the earliest SPS release that is compatible with the plugin.
`sha256sum`	string	The SHA-256 checksum of the plugin.
`version`	string	The version number of the plugin.

To configure a particular instance of a plugin, use the `/api/coniguration/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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.

```
{
    "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.

```
{
    "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",
```

```
        "previous": null,
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
    }
}
```

Element		Type	Description
key		string	Top level element, contains the ID of the plugin instance.
body		Top level element (string)	Contains the properties of the plugin instance.
	configuration	string	The configuration of the plugin instance (an INI file as a string). For details, see the documentation of the particular plugin.
	name	string	The name of the plugin instance. This field can contain only letters (a-z, A-Z), numbers (0-9) and the underscore (_) character, and must begin with a letter.
	plugin	JSON object	Contains the details of the plugin object that this instance refers to: the ID of the plugin and its endpoint, for example, `"plugin": { "key": "8876228625d67aa91e2253", "meta": { "href": "/api/-configuration/plugins/aa/8876228625d67aa91e2253" } }`

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 32.

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 35.

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 41.

Code	Description	Notes
400	SemanticError	The configuration of the instance is invalid. Check the `error` key in the response for details.

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 32.

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 35.

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 32.

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 35.

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

Cookie name	Description	Required	Values
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 more information 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 credential store plugins stored on SPS.

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

The following command retrieves the properties of a specific plugin.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/plugins/credentialstore/<plugin-id>
```

Response

The following is a sample response received when listing the credential store plugins used on SPS.

For more information on the `meta` object, see Message format on page 10.

```
{
    "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.

```
{
  "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,
```

```
    "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	Type	Description
`key`	string	Top level element, contains the ID of the plugin.
`body`	Top level element (string)	Contains the properties of the plugin.
`api`	string	The API version of the plugin.
`description`	string	The description of the plugin. This description is also displayed on the SPS web interface.
`default_configuration`	string	The default configuration of the plugin (an INI file as a string). For details, see the documentation of the particular plugin.
`entry_point`	string	The entry point of the plugin, for example, `main.py`
`name`	string	The name of the plugin. This name is also displayed on the SPS web interface. It cannot contain whitespace.
`path`	string	The path where the plugin is stored on SPS.
`scb_max_version`	string	The version number of the latest SPS release that is compatible with the plugin.
`scb_min_version`	string	The version number of the earliest SPS release that is compatible with the plugin.
`sha256sum`	string	The SHA-256 checksum of the plugin.
`version`	string	The version of the 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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information on authentication, see Authenticate to the SPS REST API on page 19.

Cookie name	Description	Required	Values
			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 more information on the `meta` object, see Message format on page 10.

```
{
  "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",
```

```
    "next": "/api/configuration/policies/indexing",
    "parent": "/api/configuration/policies",
    "previous": "/api/configuration/policies/content_policies",
    "transaction": "/api/transaction"
  }
}
```


When retrieving the endpoint of a specific credential store, the response is the following.


```
{
  "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/1580973975727acedd51b2",
    "href":
"/api/configuration/policies/credentialstores/935272738572bc2ec1dbdd",
    "last":
"/api/configuration/policies/credentialstores/935272738572bc2ec1dbdd",
    "next": null,
    "parent": "/api/configuration/policies/credentialstores",
    "previous":
"/api/configuration/policies/credentialstores/1580973975727acedd51b2",
    "transaction": "/api/transaction"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the credential store.
body	Top level element (string)	The configuration elements of the credential store.
name	string	The name of the credential store. This name is also displayed on the SPS web interface. It cannot contain whitespace.
type	Top level item	All elements for the configured type of credential store.
authenticator_name	string	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.
default_namespace	string	The default namespace of the accounts (for example, [Linux], [LDAP], [IPMI], W2003DOMAIN).
dns_servers	Top level	The IP addresses of the DNS servers to use

Element		Type	Description
		item	for resolving the hostnames provided in `domain_mappings`.
	`domain_mappings`	Top level list	Use for RDP connections only. In a domainless environment, use `default_namespace`.
	`encryption`	Top level item	Configures the encryption key for the local credential store.
	`login_mode`	Top level item	Configures the account SPS uses to login to the ERPM server.
	`plugin`	string	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.
	`proxy_server`	string	The IP address and port of the proxy server. Use the `http://` or `https://` prefix.
	`selection`	string	Configures the type of the credential store. Possible values are: • `local` Local credential store. Can only be configured using the web interface of SPS. • `external_plugin` Credential Store Plug-in. To upload or update a plugin, see Upload a plugin.
	`server_certificate_chec`	Top level item	To verify the certificate of the ERPM server, configure `server_certificate_check`.
	`web_interface_url`	string	Name of the DN of the ERPM server. Use the `http://` or `https://` prefix.

Elements of `dns_servers`	Type	Description
`primary`	string	The IP address of the primary DNS server.
`secondary`	string	The IP address of the secondary DNS server.

Elements of `domain_mappings`	Type	Description
`domain`	string	The domain name used for Domain/Host mapping.
`host`	Top level item	The host name or address of the domain controller used for Domain/Host mapping.
`selection`	string	Declares if the `value` element contains an IP or an FQDN. Possible values are: • `fqdn` The `value` element contains a hostname. • `ip` The `value` element contains an IP.
`value`	string	The IP address or hostname of the domain controller.

Elements of `encryption`	Type	Description
`selection`	string	Defines the encryption of the local credential store. Possible values are: • `basic` The local credential store uses the built-in protection of SPS. • `password` The local credential store is protected by one or more passwords.

Elements of `login_mode`	Type	Description
`password`	string	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

Elements of `login_mode`	Type	Description
		the value of the `password` element, and remove any child elements (including the key).
`selection`	string	Possible values are: • `fixed_username` SPS uses a fix username and password. Requires the `username` and `password` elements. • `gateway_auth_credentials` SPS uses the username and password that the user provided during the gateway authentication process. Can only be used for SSH connections.
`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`	Type	Description
`enabled`	boolean	Set to `true` to verify the ERPM server's certificate.
`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.

```
{
    "name":  "API_PLUGIN",
    "type":  {
       "plugin":  {
          "key":  "2534221015734bb18aaf32",
          "meta":  {
             "href":
"/api/configuration/plugins/credentialstore/2534221015734bb18aaf32"
          }
       },
       "selection":  "external_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 41.

Code	Description	Notes
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

Completing the Welcome Wizard using REST

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:

```
{
       "meta":  {
              "eula":  "https://www.oneidentity.com/legal/sta.aspx",
              "href":  "/api/setup",
              "parent":  "/api",
              "remaining_seconds":  0
       },
       "status":  "completed"
 }
```

Request body

Element	Type	Description
`accept_eula`	boolean	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.
`network`	JSON object	Contains the initial networking configuration of SPS.
`license`	string	Your SPS license as a string. You can download your license from support portal. Example: `"license": "<?xml version=\"1.0\" encoding=\"UTF-8\"?><LicenseFile><Licenses><License Generator=\"File\" GeneratorVersion=\"3.0\" Signature=\"FEEDCAFE\"><EEnterprise>true</Enterprise><LicenseNumber>123-456-789</LicenseNumber><LicenseType>Perpetual</LicenseType><ProductName>Safeguard for Privileged Sessions</ProductName><ProductVersion>7</ProductVersion><Property Name=\"Basic Proxies\">999999</Property><sudo-iolog>0</sudo-iolog><Analytics>True</Analytics></License></Licenses></LicenseFile>"` 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:

Element	Type	Description
		`"license": null,` To upload a license file, see Upload a new license.
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 272.
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" }`

Element	Type	Description
network	JSON object	The initial networking configuration of SPS.
hostname	string	Name of the machine running SPS. For example:

Element		Type	Description
			`"hostname": "psm",`
	`domainname`	string	Name of the domain used on the network. For example: `"domainname": "example.com",`
	`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. `"initial_address": "192.168.1.10/24",` 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)
	`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.**

Element	Type	Description
`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. `"primary_dns": "192.168.1.1",` Use an IPv4 address.

Element	Type	Description
`certificates`	JSON object	The internal certificates of SPS. 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. TIP: One Identity recommends using 2048-bit RSA keys (or stronger). For example: `"certificates": { "ca": { "certificate": "-----BEGIN CERTIFICATE----- \nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMA- kGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n" }, "webserver": { "certificate": "-----BEGIN CERTIFICATE----- \nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMA-`

Element		Type	Description
			`kGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n", "private_key": "-----BEGIN RSA PRIVATE KEY-----\nMIIEo-gIBAAKCAQEA/JERC+o1Uks-vUfbzS5Yp77CNlS6RkkdZLPjl2i9+ACzv/lOy\n...\n-----END RSA PRIVATE KEY-----\n" }, "tsa": { "certificate": "-----BEGIN CERTIFICATE----- \nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMA-kGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n", "private_key": "-----BEGIN RSA PRIVATE KEY-----\nMIIEo-gIBAAKCAQEA/JERC+o1Uks-vUfbzS5Yp77CNlS6RkkdZLPjl2i9+ACzv/lOy\n...\n-----END RSA PRIVATE KEY-----\n" } },`
	`ca`	JSON object	The certificate of SPS's internal Certificate Authority.:
	`webserver`	JSON object	The SSL certificate of SPS's web and REST interface.
	`tsa`	JSON 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 41.

Code	Description	Notes
401	WebGuiOrRpcApiConfigInProgress	Web based or RPC API configuration is in progress — another client is connected to SPS. You can see the IP address of the

Code	Description	Notes

		client in the details key of the response, for example:
		
		{
		"error": {
		"details": {
		"user": "admin@10.30.255.70"
		},
		"message": "Web based or RPC API configuration is in progress.",
		"type": "WebGuiOrRp-cApiConfigInProgress"
		},
		"meta": {
		"href": "/api/setup",
		"next": "/api/transaction",
		"parent": "/api",
		"remaining_seconds": 0
		}
		}

401	ConfigurationAlreadyInitialized	The Welcome Wizard was already successfully completed on this SPS.
		
		{
		"error": {
		"message": "The configuration of the system is already initialized.",
		"type": "ConfigurationAlreadyInitialized"
		},
		"meta": {
		"href": "/api/setup",
		"parent": "/api",
		"remaining_seconds": 0
		}
		}

18

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:

```
{
    "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",
```

```
        "parent": "/api/configuration/local_services",
        "previous": "/api/configuration/local_services/admin_web",
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
    }
}
```

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 32.

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 35.

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 41.

Code	Description	Notes
200	OK	Updating the resource was successful
201	Created	The new resource was successfully created.
401	Unauthenticated	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.
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	NotFound	The requested object does not exist.

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

Cookie name	Description	Required	Values
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 more information 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 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 more information on the `meta` object, see Message format on page 10.


```
{
    "items":  [
       {
          "key":  "9316362595a747b24d295e",
          "meta":  {"href":
"/api/configuration/policies/analytics/9316362595a747b24d295e"}
       },  {
          "key":  "9316362595a747b24d295f",
          "meta":  {"href":
"/api/configuration/policies/analytics/9316362595a747b24d295f"}
       }
    ]
 }
    "meta":  {
       "first":  "/api/configuration/policies/aa_plugin_instances",
       "href":  "/api/configuration/policies/analytics",
       "last":  "/api/configuration/policies/usermapping_policies",
       "next":  "/api/configuration/policies/audit_policies",
       "parent":  "/api/configuration/policies",
       "previous":  "/api/configuration/policies/aa_plugin_instances",
       "remaining_seconds":  599,
       "transaction":  "/api/transaction"
    }
```


When retrieving the endpoint of a specific analytics policy, the response is the following.


```
{
    "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"
    }
```


Element		Type	Description
body, or items when a list is returned		Top-level element (string)	Contains the properties of the analytics policy.
	name	string	The unique name of the policy. This name is also displayed on the SPS web interface. It cannot contain whitespaces.
	scoring	Top-level element	Scoring settings for analytics.
key		string	Top-level element, contains the ID of the policy.

Elements of scoring	Type	Description
command	string	Contains one of the following values: • disable: The algorithm is not used and is therefore not scoring session data. • 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. • 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.
fis	string	
hostlogin	string	
keystroke	string	
logintime	string	
mouse	string	
windowtitle	string	

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 32.

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:

   ```
   {
       "key": "1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
       "meta": {
          "href": "/api/configuration/policies/analytics/1e089e2a-76b4-4079-94e3-c83ebc74dc2e",
          "parent": "/api/configuration/policies/analytics",
          "transaction": "/api/transaction"
       }
    }
   ```

3. **Commit your changes.**

   For more information, see Commit a transaction on page 35.

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 32.

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 35.

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 32.

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 47.

   If the DELETE request is successful, when querying `/api/configuration/policies/analytics`, the response includes the key of the deleted analytics policy. For example:

```
{
    "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 35.

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 41.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	SemanticError	The request to create an object has failed due to semantic errors in the configuration.
401	Unauthenticated	The requested resource cannot be retrieved because the

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	NotFound	The requested object does not exist.

About us

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