One Identity Safeguard for Privileged Sessions 6.0

REST API Reference Guide

Copyright 2020 One Identity LLC.

ALL RIGHTS RESERVED.

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

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

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

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

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

Patents

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

Trademarks

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

Legend

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

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

SPS REST API Reference Guide
Updated - February 2020
Version - 6.0

Contents

Introduction **9**
Message format 9
How to configure SPS using REST 13
How to configure SPS using REST: a sample transaction 14

Using the SPS REST API **18**
Authenticate to the SPS REST API 18
Authenticate to the SPS REST API using X.509 certificate 21
Retrieve user information 24
Checking the transaction status 27
Open a transaction 28
Commit a transaction 30
Delete a transaction 32
Reviewing the changelog of a transaction 34
Application level error codes 36
Navigating the configuration of SPS 39
Modifying the configuration of SPS 42
Delete an object 42
Create a new object 44
Change an object 47

Basic settings **51**
Retrieve basic firmware and host information 51
Network settings 54
Web interface 54
Network configuration options 56
DNS servers 59
Routing between interfaces 61
Naming options 64
Network addresses 67
Routing table 72
Local services of SPS 74
Local services: Web login for administrators 78

Local services: Web login for users 82
Local services: cluster interface 87
Date and time 89
Date & time 89
NTP servers 91
Timezone 94
Logs, monitoring and alerts 102
Management options 102
Syslog server settings 105
Disk fill-up prevention 111
Mail settings 114
Health monitoring 121
SNMP settings 124
SNMP traps 126
Local services: access for SNMP agents 132
Alerting 140
System alerts 142
Traffic alerts 149

User management and access control 157
User management and access control 157
Authentication and user database settings 159
Privileges of usergroups 180
Audit data access rules 186
Active sessions 189
Manage users and usergroups locally on SPS 190
Manage usergroups locally on SPS 192
Manage users locally on SPS 196

Managing SPS 200
Troubleshooting options 200
Internal certificates 203
Passwords stored on SPS 206
Private keys stored on SPS 210
Certificates stored on SPS 213
Local services: enabling SSH access to the SPS host 218

RPC API 224
Manage the SPS license 226
Change contact information 230
Splunk integration 232
Splunk integration 236
Manage Safeguard for Privileged Sessions clusters 242
Promote a Safeguard for Privileged Sessions node to be the Central Management node in a new cluster 245
Join node(s) to the cluster 246
Query join status 249
Assign a role to a node 250
Query nodes 251
Query one particular node 253
Query the status of all nodes in the cluster 253
Query the status of one particular node 259
Upload and enable a configuration synchronization plugin 263
Disable a configuration synchronization plugin 268

General connection settings 270
Channel policy 270
Policies 274
Archive/Cleanup policy 279
Audit policies 283
Backup policy 292
Real-time content monitoring with Content Policies 300
LDAP servers 307
Signing CA policies 320
Time policy 324
Trusted Certificate Authorities 327
Local user databases 333
User lists 338

HTTP connections 344
HTTP connections 344
HTTP connection policies 346
HTTP authentication policies 362
Global HTTP options 370

HTTP settings policies 375

Citrix ICA connections 381

ICA connections 381

ICA connection policies 383

ICA channels 384

Global ICA options 386

ICA settings policies 392

RDP connections 397

RDP connections 397

RDP connection policies 399

RDP channels 419

Configuring domain membership 422

Global RDP options 425

RDP settings policies 431

SSH connections 438

SSH connections 438

SSH connection policies 440

SSH channels 460

SSH authentication policies 465

Global SSH options 477

SSH settings policies 483

SSH host keys and certificates 489

Telnet connections 494

Telnet connections 494

Telnet connection policies 496

Global Telnet options 497

VNC connections 503

VNC connections 503

VNC connection policies 505

Global VNC options 506

Search, download, and index sessions 512

Audited sessions 512

Download audit trails 532

Searching in the session database 532
Searching in connection content 549
Session statistics 552
Session histogram 562
Session alerts 565
Session events 569
Indexing sessions 574
Session audit trail downloads 576
Local services: configuring the indexer 579
Indexer policies 587

Reporting 594
Reporting 594
Reports 596
Built-in subchapters 605
Pre-defined reports 609
Content subchapters 613
Custom subchapters 621
Connection statistics subchapters 626

Health and maintenance 633
Monitor appliance health status 633

Advanced authentication and authorization 635
Usermapping policy 635
Plugins 640
Authentication and authorization plugins 642
Credential store plugins 645
Credential stores 648

Completing the Welcome Wizard using REST 656
Completing the Welcome Wizard using REST 656

Enable and configure analytics using REST 663
Enable One Identity Safeguard for Privileged Analytics 663
Configure One Identity Safeguard for Privileged Analytics 665

About us 671
Contacting us 671

Technical support resources 671

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 34.
	`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 details on setting the session timeout, see Web interface on page 54.
	`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)	Path of the first sibling of the current resource	

Element	Type	Description	Notes
`last`	string (relative path)	Path of the last sibling of the current resource	
`transaction`	string (`/api/transaction`)	The endpoint of the transaction log	For details on how SPS handles transactions, see How to configure SPS using REST on page 13.
`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-
10.40.0.28:22",
        "meta": {
        "href": "/api/ssh-host-
keys/ssh-rsa-
10.40.0.28:22"
        }
```

Element	Type	Description	Notes

			`}`
			`]`
			`}`

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 9.
- `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 36.
`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": {
                        "type": "NodeNotFound",
                        "message": "Unable to locate the requested path",
                        "details": {
                        "path": "no/such/path"
                        }
                        },
                        "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. 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 14.

1. Authenticate on the SPS REST server, and receive a `session_id`. For details, see Authenticate to the SPS REST API on page 18.
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 28.

   Note that opening a transaction locks the configuration of SPS similarly to accessing SPS from the web interface. For details, see "Multiple users and locking" in the Administration Guide.
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 39 and Modifying the configuration of SPS on page 42
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 30.

   If the **AAA > 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 **AAA > Accounting** page of the SPS web interface. Note that on the **AAA > Accounting** 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 32.

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 13.

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 **AAA > 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 **AAA > 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 **AAA > Accounting** page of the SPS web interface. Note that on the **AAA > Accounting** 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 21.

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

  - If the `types` field of the response includes the `x509` object, certificate-based authentication is permitted.
  - If it includes only the `basic` object, password authentication is permitted.
  - If it includes both fields, then certificate-based authentication is permitted for the users, but the `admin` user can authenticate with password as well. Note that in this case, SPS assumes that the `admin` user will authenticate with a password, and expects password-authentication on the `/api/authentication` endpoint. To authenticate with a certificate, use the `/api/authentication?type=x509` endpoint.
- You can access the REST server on the same IP address and port that you use to access the SPS web interface. Note that management (administrator) access must be enabled on the interface. For details on configuring management access, see "Configuring user and administrator login addresses" in the Administration Guide.
- For the user to have full access over the SPS REST API, they must have the **REST server** privilege. The user privileges on the web UI and REST API are synchronized. For example, if the user has the **ICA Control / Connections** privilege then they can access this page on the web UI and also the `/api/configuration/ica/connections` endpoint on the REST API.For details, see "Modifying group privileges" in the Administration Guide.

Note that the built-in **api** usergroup does not have this privilege by default, it is used to access the SOAP RPC API of SPS.

- 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 details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

Code	Description	Notes
200	OK	Successful authentication
400	InvalidAuthenticationRequest	Unable to authenticate: no valid credentials found.
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.
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 18.

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

  - If the `types` field of the response includes the `x509` object, certificate-based authentication is permitted.
  - If it includes only the `basic` object, password authentication is permitted.
  - If it includes both fields, then certificate-based authentication is permitted for the users, but the `admin` user can authenticate with password as well. Note that in this case, SPS assumes that the `admin` user will authenticate with a password, and expects password-authentication on the `/api/authentication` endpoint. To authenticate with a certificate, use the `/api/authentication?type=x509` endpoint.
- You can access the REST server on the same IP address and port that you use to access the SPS web interface. Note that management (administrator) access must be enabled on the interface. For details on configuring management access, see "Configuring user and administrator login addresses" in the Administration Guide.
- For the user to have full access over the SPS REST API, they must have the **REST server** privilege. The user privileges on the web UI and REST API are synchronized.

For example, if the user has the **ICA Control / Connections** privilege then they can access this page on the web UI and also the `/api/configuration/ica/connections` endpoint on the REST API.For details, see "Modifying group privileges" in the Administration Guide.

Note that the built-in **api** usergroup does not have this privilege by default, it is used to access the SOAP RPC API of SPS.

- 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 **AAA > 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 details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

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	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.
405	MethodNotAllowed	You tried using an unsupported HTTP method. Use the GET method for authentication.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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
```

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
  "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.
	`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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.

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 13.
- To check the configuration changes you made in the transaction, see Reviewing the changelog of a transaction on page 34.

URL

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

Sample request

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

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

Response

The following is a sample response received if opening the transaction is successful.

For details of the `meta` object, see Message format on page 9.

```
{
    "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

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 13.

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

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

POST body

Note that you must:

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

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

Sample request

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

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

Response

The following is a sample response received if opening the transaction is successful.

For details of the `meta` object, see Message format on page 9.

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

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

```
{
       "body": {
             "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 36.

Code	Description	Notes
200	OK	Transaction opened successfully.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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. 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.
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 13.

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 details on authentication, see Authenticate to the SPS REST API on page 18.

Cookie name	Description	Required	Values
			Note that this session ID refers to the connection between the REST client and the SPS 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 **AAA > 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 **AAA > Accounting** page of the SPS web interface. Note that on the **AAA > Accounting** 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 details of the `meta` object, see Message format on page 9.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 13. 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

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

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

Response

The following is a sample response received if deleting the transaction is successful.

For details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

Code	Description	Notes
200	OK	Transaction deleted successfully.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 13.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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.
400	SyntacticError	A value to be set is not accepted syntactically. The `details` section contains the path that was

Code	Description	Notes
		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 configuration subtree in an unsupported way. The method <method> is not allowed for this node.
409	MidAirCollisionSemanticError	This error occurs when the configuration has

Code	Description	Notes
		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.
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 44.

To modify or delete an object, you need the ID of the object. For details, see Change an object on page 47 and Delete an object on page 42.

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": "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 30.

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 details on authentication, see Authenticate to the

Cookie name	Description	Required	Values
			SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 36.

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`

Code	Description	Notes
		that was attempted to be accessed, but could not be retrieved.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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",
  ```

- 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 details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
409	Conflict	No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction on page 28.
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",
```

- 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`

Header 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 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 details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
409	Conflict	No open Transaction is available. Open a transaction before using this request. For details, see Open a transaction on page 28.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command displays the information about SPS that is available without authentication.

```
curl https://10.40.255.171/api/info
```

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

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

Response

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

For details of the `meta` object, see Message format on page 9.


```
{
       "body":  {
        "domainname":  "example",
              "hostname":  "scbwriter",
              "nickname":  null,
              "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,
       "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 64.
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 64.
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 64.
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 114.
firmware_version	The version number of the firmware running on SPS, for 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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.

Network settings

Web interface

Configuration options for the web interface of SPS.

URL

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

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 the SPS web interface.

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

Response

The following is a sample response received when listing the configuration options of the SPS web interface.

For details of the `meta` object, see Message format on page 9.

```
{
    "body": {
       "timeout": 10
    },
    "key": "webinterface",
    "meta": {
       "first": "/api/configuration/management/certificates",
       "href": "/api/configuration/management/webinterface",
       "last": "/api/configuration/management/webinterface",
       "next": null,
       "parent": "/api/configuration/management",
       "previous": "/api/configuration/management/syslog",
       "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.
timeout	int	Session timeout, in minutes. SPS terminates sessions that are idle for this period. This setting applies sessions that access the SPS web interface and the SPS REST interface.

Modify the configuration of the web interface

To modify the configuration of the web interface, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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

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

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

Status and error codes

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists network configuration options.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
  "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.
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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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_id	Contains the authentication token of the user	Required	The value of the session ID cookie received from the REST server in the authentication response, for example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configured DNS servers.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
    "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 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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists interface routing rules.

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

Response

The following is a sample response received when listing interface routing rules.

For details of the `meta` object, see Message format on page 9.


```
{
    "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	Contains the rules for routing between the network inter-

Element	Type	Description
	level element (list)	faces.
`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 `/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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a rule for routing between the network interfaces

To modify a rule, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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	Required	The value of the session ID cookie received from the REST server in the authentication response, for

Cookie name	Description	Required	Values
	user		example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
    "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	Contains the naming settings.

Element	Type	Description
	level element (string)	
`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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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`

Element	Type	Description
		Negotiate the network speed automatically. This is the default value. • `10-half` 10BaseT/Half. • `100-half` 100BaseT/Half. • `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

Elements of `interfaces`	Type		Description
			SPS web UI.
	`source_based_routes`	list	Contains details of the network 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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

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

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
  "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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify the routing table

To modify the routing table, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the local services.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
    "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, depending on their access privileges, modify the configuration of SPS, and perform authentication-related activities (gateway authentication, 4-eyes

Element	Description
	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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configuration options.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
	"body": {
		"access_restriction": {
			"allowed_from": [
				"10.40.0.0/16"
			],
			"enabled": true
		},
		"bruteforce_protection": true,
		"listen": [
			{
				"address": {
					"key":
"nic1.interfaces.ff7574025754b3df1647001.addresses.1",
					"meta": {
						"href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/
1"
					}
				},
				"http_port": 80,
				"https_port": 443
			}
		]
	},
	"key": "admin_web",
	"meta": {
		"first": "/api/configuration/local_services/admin_web",
		"href": "/api/configuration/local_services/admin_web",
		"last": "/api/configuration/local_services/user_web",
		"next": "/api/configuration/local_services/indexer",
		"parent": "/api/configuration/local_services",
		"previous": null,
		"transaction": "/api/transaction"
	}
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the endpoint.
body	Top level element	Contains the configuration options of the SPS web interface.

Element		Type	Description
		(string)	
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.
	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
```

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.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_`	integ-	The port number where SPS accepts HTTP connections. Note

Element	Type	Description
port	er	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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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_	Contains the	Required	The value of the session ID cookie received from

Cookie name	Description	Required	Values
id	authentication token of the user		the REST server in the authentication response, for example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "body": {
        "access_restriction": {
            "allowed_from": [
                "10.40.0.0/16"
            ],
            "enabled": true
        },
        "bruteforce_protection": true,
        "listen": [
            {
                "address": {
                    "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1",
                    "meta": {
                        "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/
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.

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

Element	Type	Description

		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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 Safeguard for Privileged Sessions (SPS) cluster. Complete the following steps on each and every 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configuration options.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
        "body": {
                "enabled": true,
                "listen_address": {
                        "key":
"nic1.interfaces.ff7574025754b3df1647001.addresses.2553887595ce3ca7f1eae4",
                        "meta": {
                                "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/2
553887595ce3ca7f1eae4"
                        }
                }
        },
        "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 36.

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
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists NTP server addresses.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
    "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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify an NTP server address

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

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
  "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`

Element	Type	Description

		`Africa/Bamako`
		`Africa/Bangui`
		`Africa/Banjul`
		`Africa/Bissau`
		`Africa/Blantyre`
		`Africa/Brazzaville`
		`Africa/Bujumbura`
		`Africa/Cairo`
		`Africa/Casablanca`
		`Africa/Ceuta`
		`Africa/Conakry`
		`Africa/Dakar`
		`Africa/Dar_es_Salaam`
		`Africa/Djibouti`
		`Africa/Douala`
		`Africa/El_Aaiun`
		`Africa/Freetown`

Element	Type	Description
		Kigali Africa/Kinshasa Africa/Lagos Africa/Libreville Africa/Lome Africa/Luanda Africa/Lubumbashi Africa/Lusaka Africa/Malabo Africa/Maputo Africa/Maseru Africa/Mbabane Africa/Mogadishu Africa/Monrovia Africa/Nairobi Africa/Ndjamena Africa/Niamey Africa/Nouakchott Africa/Ouagadougou

Element	Type	Description

		`entina/Tucuman`
		`America/Argentina/Ushuaia`
		`America/Aruba`
		`America/Asuncion`
		`America/Atikokan`
		`America/Atka`
		`America/Bahia`
		`America/Barbados`
		`America/Belem`
		`America/Belize`
		`America/Blanc-Sablon`
		`America/Boa_Vista`
		`America/Bogota`
		`America/Boise`
		`America/Buenos_Aires`
		`America/Cambridge_Bay`
		`America/Campo_Grande`
		`America/Cancun`
		`America/Caracas`
		`America/Catamarca`
		`America/Cayenne`

Element	Type	Description
		``` ca/Miquelon America/Moncton America/Monterrey America/Montevideo America/Montreal America/Montserrat America/Nassau America/New_York America/Nipigon America/Nome America/Noronha America/North_Dakota/Center America/North_Dakota/New_Salem America/Panama America/Pangnirtung America/Paramaribo America/Phoenix America/Port-au-Prince America/Port_of_Spain America/Porto_Acre America/Porto_Velho America/Puerto_Rico America/Rainy_River America/Rankin_Inlet America/Recife America/Regina ```

Element	Type	Description
		ope/Bucharest Europe/Budapest Europe/Chisinau Europe/Copenhagen Europe/Dublin Europe/Gibraltar Europe/Guernsey Europe/Helsinki Europe/Isle_of_Man Europe/Istanbul Europe/Jersey Europe/Kaliningrad Europe/Kiev Europe/Lisbon Europe/Ljubljana Europe/London Europe/Luxembourg Europe/Madrid Europe/Malta Europe/Mariehamn Europe/Minsk Europe/Monaco Europe/Moscow Europe/Nicosia Europe/Oslo Europe/Paris Europe/Podgorica Europe/Prague Europe/Riga Europe/Rome Europe/Samara Europe/San_Marino Europe/Sarajevo Europe/Simferopol Europe/Skopje Europe/Sofia Europe/Stockholm Europe/Tallinn Europe/Tirane

Element	Type	Description

Modify the time zone

To modify time zone, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
  "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": "health_monitoring",
      "meta": {
        "href": "/api/configuration/management/health_monitoring"
      }
    },
    {
      "key": "snmp",
      "meta": {
        "href": "/api/configuration/management/snmp"
      }
    },
    {
      "key": "soap",
      "meta": {
        "href": "/api/configuration/management/soap"
      }
    },
    {
      "key": "syslog",
      "meta": {
        "href": "/api/configuration/management/syslog"
      }
    },
    {
      "key": "webinterface",
      "meta": {
        "href": "/api/configuration/management/webinterface"
```

```
      }
    }
  ],
  "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.
health_monitoring	Configuration settings for monitoring the utilization of SPS.
snmp	SNMP settings.
soap	Configuration settings for the RPC API.
syslog	Syslog server address and authentication.
webinterface	Configuration settings for the SPS web 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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.

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

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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/soap",
       "transaction": "/api/transaction"
    }
 }
```


Element	Type	Description
key	string	Top level element, contains the ID of the endpoint.
body	Top level element	Contains the syslog server configuration settings.

Element		Type	Description
		(string)	
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.
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.

Element	Type	Description
		• `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.
	`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`

Elements of `receivers`	Type	Description
		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\nQ0ExEDAOBgNVBAgT
B09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW
50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4X
DTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP\nbnRh
cmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQQLEw1Eb2N1bWVudGF0aW
9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74
ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw\n0qNRjPS1ML
zjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc\n3mV2zkvtD61CXzbezAVdArDl6Wny
sRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3F
y2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjML
U2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp
5USme4PmNhY6QfUlgjsFjPfoVG7XDB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C
8xstRSEG7oJc\nN5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyy
D1ld89OUEYZ+aJQd/A==\n-----END CERTIFICATE-----\n"
```


Modify syslog server settings

To modify the syslog server settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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	Required	The value of the session ID cookie received from the REST server in the authentication response, for

Cookie name	Description	Required	Values
	user		example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists disk fill-up prevention options.

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

Response

The following is a sample response received when listing disk fill-up prevention settings.

For details of the `meta` object, see Message format on page 9.

```
{
    "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.

Element		Type	Description
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.

Code	Description	Notes
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists mail settings.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
    "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).

Element		Type	Description
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 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.

Elements of smtp_encryption		Type	Description
			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 213.
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.
	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\nQ0ExEDAOBgNVBAgT
B09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW
50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4X
DTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP\nbnRh
cmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQQLEw1Eb2N1bWVudGF0aW
9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74
ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw\n0qNRjPS1ML
zjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc\n3mV2zkvtD61CXzbezAVdArDl6Wny
sRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3F
y2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjML
U2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp
5USme4PmNhY6QfUlgjsFjPfoVG7XDB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C
8xstRSEG7oJc\nN5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyy
D1ld89OUEYZ+aJQd/A==\n-----END CERTIFICATE-----\n"
```


Modify mail settings

To modify mail settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
  "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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the endpoints for SNMP configuration settings.

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

Response

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

For details of the `meta` object, see Message format on page 9.


```
{
    "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/soap",
       "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 36.

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
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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_id	Contains the authentication token of the user	Required	The value of the session ID cookie received from the REST server in the authentication response, for example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configuration of the SNMP server.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
    "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

Elements of `version`		Type	Description
			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.
	`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_`

Elements of `version`	Type	Description
		`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. 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.

Elements of `version`	Type	Description
`value`	string	The address of the SNMP server.
`username`	string	Must be used if version 3 of the SNMP 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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configuration options.

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

Response

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

For details of the `meta` object, see Message format on page 9.


```
{
    "body": {
        "access_restriction": {
            "enabled": false
        },
        "enabled": true,
        "listen": [
            {
                "address": {
                    "key": "nic1.interfaces.ff7574025754b3df1647001.addresses.1",
                    "meta": {
                        "href":
"/api/configuration/network/nics/nic1#interfaces/ff7574025754b3df1647001/addresses/
1"
                    }
                },
                "port": 161
            }
```

```
        ],
        "system_contact": "mycontact",
        "system_description": "mydescription",
        "system_location": "mylocation",
        "version_2c": {
            "community": "mycommunity",
            "enabled": true
        },
        "version_3": {
            "enabled": true,
            "users": [
                {
                    "auth_method": "sha",
                    "auth_password": {
                        "key": "5476940c-ba38-4002-96d4-cb09d6921c68",
                        "meta": {
                            "href": "/api/configuration/passwords/5476940c-ba38-
4002-96d4-cb09d6921c68"
                        }
                    },
                    "encryption_method": "aes",
                    "encryption_password": {
                        "key": "99782a91-63de-4a5c-82ff-b82273894dc7",
                        "meta": {
                            "href": "/api/configuration/passwords/99782a91-63de-
4a5c-82ff-b82273894dc7"
                        }
                    },
                    "username": "myusername"
                }
            ]
        }
    },
    "key": "snmp_agent",
    "meta": {
        "first": "/api/configuration/local_services/admin_web",
        "href": "/api/configuration/local_services/snmp_agent",
        "last": "/api/configuration/local_services/user_web",
        "next": "/api/configuration/local_services/ssh",
        "parent": "/api/configuration/local_services",
        "previous": "/api/configuration/local_services/postgresql",
        "transaction": "/api/transaction"
    }
}
```

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": [
```

Element	Type	Description

```
                        "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.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": {
```

Element		Type	Description
			"href": "/api/configuration/network/nics/nic1#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": { "enabled": true, "users": [

Element	Type	Description

```
        {
            "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists alerting configuration endpoints.

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

Response

The following is a sample response received when listing alerting configuration endpoints. For details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

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

Code	Description	Notes
		retrieved.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
  "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 item	Too many successive failed login attempts.
	email	boolean	Set to true to enable e-mail alerts.

Element		Type	Description
	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.

Element		Type	Description
	email	boolean	Set to true to enable e-mail alerts.
	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.

Element		Type	Description
`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.
`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.
	`emai`	boolean	Set to `true` to enable e-mail alerts.

Element	Type	Description
`1`		
`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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

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

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
  "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.

Element		Type	Description
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 item	Connection denied.
	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.

Element		Type	Description
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.
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 hostkey learned.
	email	boolean	Set to true to enable e-mail alerts.
	snmp	boolean	Set to true to enable SNMP alerts.

Element		Type	Description
`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.
`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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the AAA configuration endpoints.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
    "items": [
       {
          "key": "acls",
          "meta": {
             "href": "/api/configuration/aaa/acls"
          }
       },
       {
          "key": "local_database",
          "meta": {
             "href": "/api/configuration/aaa/local_database"
          }
       },
       {
          "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.
`local_database`	Local users and usergroups.
`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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

Authentication and user database settings

Contains settings for authenticating to SPS. You can create a user database locally on SPS, or connect to an LDAP server to authenticate users. You can configure authentication with passwords, X.509 certificates, or against a RADIUS server.

URL

```
GET 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 authentication and user database settings.

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

Response

The following is a sample response received when listing authentication and user database settings.

For details of the `meta` object, see Message format on page 9.


```
{
    "key": "settings",
    "body": {
        "method": {
            "selection": "x509",
            "admin_fallback": true,
            "trusted_ca": {
                "key": "18610698755c8de61207a7",
                "meta": {"href": "/api/configuration/policies/trusted_ca_
lists/18610698755c8de61207a7"}
            },
            "username_attribute": "commonName"
        },
        "backend": {
            "selection": "ldap",
            "schema": {
                "selection": "ad",
                "membership_check": {
                    "enabled": true,
                    "nested_groups": false
                },
                "memberof_check": {
                    "enabled": true,
                    "memberof_user_attribute": "memberOf"
                },
                "user_dn_in_groups": []
            },
            "servers": [
                {
                    "host": {
                        "selection": "ip",
                        "value": "10.110.0.1"
                    },
                    "port": 389
                },
                {
                    "host": {
                        "selection": "fqdn",
                        "value": "my.example"
```

```
                    },
                    "port": 389
                }
            ],
            "user_base_dn": "ou=People,dc=example",
            "group_base_dn": "ou=Groups,dc=example",
            "bind_dn": "cn=admin,dc=example",
            "bind_password": {
                "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                "meta": {"href": "/api/configuration/passwords#XXXXXXXX-XXXX-XXXX-
XXXX-XXXXXXXXXXXX"}
            },
            "encryption": {
                "selection": "starttls",
                "server_certificate_check": {
                    "enabled": false
                },
                "client_authentication": {
                    "enabled": true,
                    "x509_identity": {
                        "key": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
                        "meta": {"href": "/api/configuration/x509/XXXXXXXX-XXXX-
XXXX-XXXX-XXXXXXXXXXXX"}
                    }
                }
            }
        },
        "require_commitlog": true
    }
}
```


Element		Type	Description
key		string	Top level element, contains the ID of the endpoint.
body		Top level element (string)	Contains the authentication settings.
	backend	Top level item	Settings for the user database (local or LDAP), and password policy.
	method	Top level item	Settings for the authentication method (password, RADIUS server, or X.509 certificate).
	require_commitlog	boolean	Set to true to request the user to write an explanation to every configuration change.

Elements of backend	Type	Description
`selection`	string	Defines the user database back-end. Possible values are: • `ldap` Use an LDAP server (AD or POSIX) for authentication. • `local` Use a local user database for authentication.
`cracklib_enabled`	boolean	Password setting. Set to `false` if a RADIUS server or X.509 certificate is used for authentication. 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. NOTE: The strength of the password is determined by its entropy: the variety of numbers, letters, capital letters, and special characters used, not only by its length. To execute some simple dictionary-based attacks to find weak passwords, set **Cracklib (eg. dictionary) check on password** to **Enabled**.
`expiration_days`	int	Password setting. Set to `0` if a RADIUS server or X.509 certificate is used for authentication. Must be used if the value of the `selection` element is set to `local`. Configures the number of days the user passwords are considered valid. Expired passwords must be changed upon login. The `0` value means the passwords

Elements of backend	Type	Description
		do not expire. The highest value you can configure is 365.
`minimum_password_strength`	string	Password setting. Set to `disabled` if a RADIUS server or X.509 certificate is used for authentication. Must be used if the value of the `selection` element is set to `local`. Configures the required password strength for new passwords. Possible values are: • `disabled` Any password is accepted. • `good` Weak passwords are not accepted. • `strong` Only strong passwords are accepted.
`remember_previous_passwords`	int	Password setting. Set to `0` if a RADIUS server or X.509 certificate is used for authentication. Must be used if the value of the `selection` element is set to `local`. Configures the number of previous passwords to retain to prevent password reuse. The `0` value means passwords can be reused.
`user_base_dn`	string	Must be used if the value of the `selection` element is set to `ldap`. Name of the DN to be used as the base of queries regarding users.

Elements of backend	Type	Description
		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.
`group_base_dn`	string	Must be used if the value of the `selection` element is set to `ldap`. 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. Must be used if the value of the `selection` element is set to `ldap`. NOTE: SPS accepts both pre-win2000-style and Win2003-style account names (User Principal Names), for example `administrator@example.com` is also accepted.

Elements of backend	Type	Description
bind_ password	string	Must be used if the value of the selection element is set to ldap. 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). NOTE: One Identity Safeguard for Privileged Sessions (SPS) accepts passwords that are not longer than 150 characters. The following special characters can be used: !"#$%&' ()*+,-./:;<=>?@[\]^-`{|}
encryption	Top level item	Must be used if the value of the selection element is set to ldap. Configuration settings for encrypting the communication between SPS and the LDAP server.
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 If you set the address using a domain name ("host": { "selection": "fqdn"), and you use a TLS-encrypted with certificate verification to connect to the LDAP server, use the full domain

Elements of backend			Type	Description
				name (for example `ldap.example.com`), otherwise the certificate verification might fail. The name of the LDAP server must appear in the 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.
	`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 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).

Elements of backend			Type	Description
	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).
		server_certificate_ca	string	Must be used if the enabled element is set to true. The certificate of the CA.
schema			Top level item	Must be used if the value of the selection element is set to ldap. Schema settings for AD and POSIX servers.
	selection		string	Configures which LDAP schema to use: AD or POSIX. Possible values are: • ad: Microsoft Active Directory server. For details and examples, see Example: Microsoft Active Directory server. • 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 Example: POSIX LDAP server.
	membership_check		Top level element	
		enabled	boolean	POSIX: Enables POSIX primary and supplementary group membership checking.

Elements of backend			Type	Description
				AD: Enables Active Directory specific non-primary group membership checking.
		nested_groups	boolean	Must be used if the `selection` element is set to `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.
		memberof_group_objectclass	string	Must be used if the value of the `selection` element is set to `posix`. The objectClass of the referred groups that can be referred in the `memberof_user_attribute`.
	username_attribute		string	Must be used if the value of the `selection` element is set to `posix`.

Elements of backend			Type	Description
				Username (user ID) attribute name is the name of the attribute in the user object, which contains the user's plain username.
	`user_dn_in_groups`		Top level list	Check the user DN in these groups is a list of additional group object classes and their respective attributes where SPS will look for member user DNs. Add `object_class` / `attribute` pairs. SPS will search for the user DN in the group's attribute defined here. For example: `"user_dn_in_groups": [ { "object_class": "groupOfNames", "attribute": "member" }, { "object_class": "groupOfUniqueNames", "attribute": "uniqueMember" } ]`
		`object_class`	string	Consider groups of this objectClass.
		`attribute`	string	Name of the group attribute which contains the user DN.
`servers`			Top level list	Must be used if the value of the `selection` element is set to `ldap`. Contains the addresses and ports of the LDAP servers.
	`host`		Top level item	Contains the address of the LDAP server.
		`selection`	string	Defines the address type (IP or domain name). Possible values

Elements of backend	Type	Description
		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. • If you set the address using an IP address (`"selection": "ip"`), use an IPv4 address. • If you set the address using a domain name (`"host": { "selection": "fqdn"`), and you use a TLS-encrypted with certificate verification to connect to the LDAP server, use the full domain name (for example `ldap.example.com`), otherwise the certificate verification might fail. The name of the LDAP server must appear in the Common Name of the certificate.
`port`	int	The port of the LDAP server.

Elements of method	Type	Description
`selection`	string	Configures the authentication method. Possible values are: • `passwd`: Use passwords for authentication. • `radius`: Configure authentication against a RADIUS server.

Elements of `method`			Type	Description
				CAUTION: **The challenge/response authentication method is currently not supported. Other authentication methods (for example password, SecureID) should work.** • `x509`: Use X.509 certificates for authentication.
`servers`			Top level list	RADIUS setting. Must be used if the value of the `selection` element is set to `radius`. Contains the RADIUS server addresses and port numbers, and references the shared secrets.
	`address`		Top level item	RADIUS setting. Must be used if the value of the `selection` element is set to `radius`. The address and port number of the RADIUS server.
	`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`.
		`selection`	string	RADIUS setting. Must be used if the value of the `selection` element is set to `radius`. Defines the address type (IP or domain name). Possible values are: • `fqdn` The RADIUS server address is provided as a fully qualified domain name. • `ip` The RADIUS server address is provided as an IP address.
		`value`	string	RADIUS setting. Must be used if the

Elements of method	Type	Description
		value of the selection element is set to radius. The address of the RADIUS server.
port	int	RADIUS setting. Must be used if the value of the selection element is set to radius. The port number of the RADIUS server.
shared_secret	string	RADIUS setting. Must be used if the value of the selection element is set to radius. References the identifier of the shared secret. You can view or modify the list of 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).
admin_fallback	boolean	X.509 setting. Must be used if the value of the selection element is set to x509. Set to true to allow the admin user to use password for login.
dn	string	X.509 setting. Must be used if the value of the selection element is set to x509. X.509 DN field name of the username (case sensitive). In most cases, this value is either CN or UID.
trusted_ca	string	X.509 setting. Must be used if the value of the selection element is set to x509. References the identifier of the trusted CA. You can view or modify the list of trusted CAs at the /api/configuration/policies/trusted_ca_lists/ endpoint. To modify or add a trusted CA, use the value of the returned key as the value of the trusted_ca element, and remove any child elements (including the key).

Example: Local user database with password authentication

This example configures a local user database with a password policy to authenticate the users of SPS:

NOTE:

One Identity Safeguard for Privileged Sessions (SPS) accepts passwords that are not longer than 150 characters. The following special characters can be used: `!"#$%&'()*+,-./:;<=>?@[\]^-`{|}`

NOTE:

The strength of the password is determined by its entropy: the variety of numbers, letters, capital letters, and special characters used, not only by its length.

To execute some simple dictionary-based attacks to find weak passwords, set **Cracklib (eg. dictionary) check on password** to **Enabled**.

NOTE:

Changes to the password policy do not affect existing passwords. However, setting password expiry will require every user to change their passwords after the expiry date, and the new passwords must comply with the strength requirements set in the password policy.

```
{
    "backend": {
        "cracklib_enabled": false,
        "expiration_days": 0,
        "minimum_password_strength": "good",
        "remember_previous_passwords": 10,
        "selection": "local"
    },
    "method": {
        "selection": "passwd"
    },
    "require_commitlog": false
}
```

Example: Local user database with RADIUS server

This example configures a local user database with a RADIUS server to authenticate the users of SPS. Note that the password-related elements have to be disabled, as

the RADIUS server determines the password policy.

CAUTION:

The challenge/response authentication method is currently not supported. Other authentication methods (for example password, SecureID) should work.

CAUTION:

After you commit this configuration, the SPS web interface will be available only after successfully authenticating to the RADIUS server. Note that the default `admin` account of SPS will be able to login normally, even if the RADIUS server is unaccessible.

```
{
    "backend": {
       "cracklib_enabled": false,
       "expiration_days": 0,
       "minimum_password_strength": "disabled",
       "remember_previous_passwords": 0,
       "selection": "local"
    },
    "method": {
       "selection": "radius",
       "servers": [
          {
             "address": {
                "selection": "ip",
                "value": "<server-ip>"
             },
             "port": <port>,
             "shared_secret": "<id-of-the-password>"
          }
       ]
    },
    "require_commitlog": false
 }
```

Example: Local user database with X.509 certificates

This example configures a local user database with X.509 certificates to authenticate the users of SPS. Note that the password-related elements have to be disabled.

```
{
    "backend": {
        "cracklib_enabled": false,
        "expiration_days": 0,
        "minimum_password_strength": "disabled",
        "remember_previous_passwords": 0,
        "selection": "local"
    },
    "method": {
        "admin_fallback": true,
        "dn": "<CN>",
        "selection": "x509",
        "trusted_ca": "<id-of-the-trusted-ca>"
    },
    "require_commitlog": false
}
```

Example: POSIX LDAP server

NOTE:

- The `admin` user is available by default and has all privileges. It is not possible to delete this user.
- Enabling LDAP authentication automatically disables the access of every local user except for `admin`. The `admin` user can login to SPS even if LDAP authentication is used.
- SPS accepts both pre-win2000-style and Win2003-style account names (User Principal Names). User Principal Names (UPNs) consist of a username, the at (@) character, and a domain name, for example `administrator@example.com`.
- For the username of SSH users, only valid UTF-8 strings are allowed.
- The following characters cannot be used in:
  - usernames: `/\[]:;|=+*?<>"`
  - group names: `/\[]:;|=+*?<>"@,`
- When using RADIUS authentication together with LDAP users, the users are authenticated to the RADIUS server, only their group memberships must be managed in LDAP. For details, see "Authenticating users to a RADIUS server" in the Administration Guide.
- SPS treats user and group names in a case insensitive manner if the matching rule for the attribute in question is case insensitive in the LDAP database.

CAUTION:

Nested groups can slow down the query and cause the connection to timeout if the LDAP tree is very large. In this case, disable the Enable nested groups option.

NOTE:

You also have to configure the usergroups in SPS and possibly in your LDAP database. For details on using usergroups, see "Using usergroups" in the Administration Guide.

This example configures a POSIX LDAP server, communication between SPS and the LDAP server is not encrypted. Note that for password authentication, the password-related elements have to be omitted from the JSON, as the POSIX server determines the password policy.

```
{
    "backend": {
        "selection": "ldap",
        "user_base_dn": "<base-dn>",
        "group_base_dn": "<base-dn>",
        "bind_dn": "<bind-dn>",
        "bind_password": "<id-of-the-password>",
        "schema": {
            "selection": "posix",
            "username_attribute": "<uid-attr>",
            "membership_check": {
                "enabled": true,
                "member_uid_attribute": "<memberUid-attr>"
            },
            "memberof_check": {
                "enabled": true,
                "memberof_user_attribute": "<user-attr-of-group-dns>",
                "memberof_group_objectclass": "<object-class-of-groups>"
            },
            "user_dn_in_groups": []
        },
        "servers": [
            {
                "host": {
                    "selection": "ip",
                    "value": "<ip-of-server>"
                },
                "port": <port>
            }
        ],
        "encryption": {
            "selection": "disabled"
        }
    },
    "method": {
        "selection": "passwd"
    },
    "require_commitlog": false
}
```

Example: Microsoft Active Directory server

NOTE:

- The `admin` user is available by default and has all privileges. It is not possible to delete this user.
- Enabling LDAP authentication automatically disables the access of every local user except for `admin`. The `admin` user can login to SPS even if LDAP authentication is used.
- SPS accepts both pre-win2000-style and Win2003-style account names (User Principal Names). User Principal Names (UPNs) consist of a username, the at (@) character, and a domain name, for example `administrator@example.com`.
- For the username of SSH users, only valid UTF-8 strings are allowed.
- The following characters cannot be used in:
  - usernames: `/\[]:;|=+*?<>"`
  - group names: `/\[]:;|=+*?<>"@,`
- When using RADIUS authentication together with LDAP users, the users are authenticated to the RADIUS server, only their group memberships must be managed in LDAP. For details, see "Authenticating users to a RADIUS server" in the Administration Guide.
- SPS treats user and group names in a case insensitive manner if the matching rule for the attribute in question is case insensitive in the LDAP database.

CAUTION:

Nested groups can slow down the query and cause the connection to timeout if the LDAP tree is very large. In this case, disable the Enable nested groups option.

NOTE:

You also have to configure the usergroups in SPS and possibly in your LDAP database. For details on using usergroups, see "Using usergroups" in the Administration Guide.

This example configures a Microsoft Active Directory server with mutual authentication, and SPS verifies the certificate of the server. Note that for password authentication, the password-related elements have to be omitted from the JSON, as the AD server determines the password policy.

```
{
    "backend": {
        "selection": "ldap",
        "user_base_dn": "<base-dn>",
        "group_base_dn": "<base-dn>",
        "bind_dn": "<bind-dn>",
        "bind_password": "<id-of-the-password>",
        "schema": {
            "selection": "ad",
            "membership_check": {
                "enabled": true,
                "nested_groups": true
            },
            "memberof_check": {
                "enabled": true,
                "memberof_user_attribute": "<user-attr-of-group-dns>"
            },
            "user_dn_in_groups": []
        },
        "servers": [
            {
                "host": {
                    "selection": "ip",
                    "value": "<ip-of-server>"
                },
                "port": <port>
            }
        ],
        "encryption": {
            "selection": "starttls",
            "server_certificate_check": {
                "enabled": true,
                "server_certificate_ca": "<cert>"
            },
            "client_authentication": {
                "enabled": true,
                "x509_identity": "<id-of-the-cert-and-key>"
            }
        }
    },
    "method": {
        "selection": "passwd"
    },
    "require_commitlog": false
}
```

Modify the authentication and user database settings

To modify the authentication and user database settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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

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

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the local users.

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

Response

The following is a sample response received when querying the endpoint.

For details of the `meta` object, see Message format on page 9.

```
{
    "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": "api",
            "objects": [
                "/special/rpcapi"
            ],
            "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 36.

Code	Description	Notes
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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
       "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 **AAA > Access Control**, the `autogenerated-all-data-access-id` rule is automatically generated. Therefore, you can almost always query this audit data access rule.

1. **Query the `https://<IP-address-of-SPS>/api/acl/audit_data/<key-of-rule-to-be-queried>` endpoint.**

```
curl --cookie cookies https://<IP-address-of-SPS>/api/acl/audit_data/<key-of-rule-to-be-queried>
```

The following is a sample response received.

For details of the `meta` object, see Message format on page 9.

```
{
       "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 terminates the specified session.

URL

```
GET https://<IP-address-of-SPS>/api/active-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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 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 delete 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
```

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 36.

Code	Description	Notes
400	SessionIdMissing	No session id is given in the id query parameter.
500	SessionTerminationFailed	The session could not be terminated due to internal errors.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the endpoints of the local database.

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

Response

The following is a sample response received when listing the endpoint.

For details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 usergroups.

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

Response

The following is a sample response received when querying a particular usergroup endpoint.

For details of the `meta` object, see Message format on page 9.


```
{
    "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 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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 28).

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 44.

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 30.

Delete usergroup

To delete a usergroup, you have to:

1. Open a transaction (for details, see Open a transaction on page 28).
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 42. 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 30).

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 28).
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 42.
4. Commit your changes to actually delete the object from SPS (for details, see Commit a transaction on page 30).

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 Authentication and user database settings on page 159.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection

Cookie name	Description	Required	Values
			between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the local users.

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

The following command displays the parameters of a specific user.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/aaa/local_
database/users/<ID-of-the-user>
```

Response

The following is a sample response received when querying the list of users.

For details of the `meta` object, see Message format on page 9.


```
{
    "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 206.
	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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 28).

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
    "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	Settings for debug logging.

Element	Type	Description
`logging`	level 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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
  "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 value

Element	Type	Description
		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 213.
`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 213.
`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 213.

Modify a certificate

To modify a certificate, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 44. 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 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./"}'
```

Element	Type	Description
`hash`	string	Must contain the SHA-256 hash of the password to be created, for example, `"hash": "ddec437eeb1da25a146a24c432d1165bc646daa7fecc6aa14c636265c83caa14"`. The request must contain at least the `hash` or the `plain` attribute.
`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"`. The request must contain at least the `hash` or the `plain` attribute.

Response

The response to a successful POST message is a JSON object that includes the reference ID of the created password in its `key` attribute.

For details of the `meta` object, see Message format on page 9.

```
{
       "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 Element . 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 details, see Commit a transaction on page 30.

Change the root password

To change the password of the `root` user, complete the following steps.

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 Element . 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. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 44. 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 4: 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 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.

Element		Type	Description
	digest	string	The fingerprint of the key, 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`.
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 details of the `meta` object, see Message format on page 9.

```
{
       "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": "ef:d3:8e:d0:81:4f:a2:8f:3b:8b:0c:dd:c7:8f:8c:7e",
            "hash_algorithm": "md5"
        },
        "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 36.

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 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.
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.

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 44. 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 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 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.....\nI2SchDibk
/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .........
Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END RSA
PRIVATE KEY-----\n"}'
```

The body should be:

```
{
       "certificate": "-----BEGIN CERTIFICATE-----
\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2SchDibk
/Xj/ZvuEQ23LvzayWOVVuVHtH3JZX3SU4Sa0vpaeC+3oddVTwQOWRq0\n .........
Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END
CERTIFICATE-----",
       "private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEpAIBAAKCAQEAu3QMMhqeg9ZMLNfdvQoNN1deVRE2SR0VKY+ALnzPZF4fUoJy\n.....\nI2SchDibk
/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 ..... Qbn5W3xKz4vXDDQHEbEsvDQ9A7+uCEuHpO4s33IK9KEa0Zdp745AU0DSGXN4HFzc\n-----END RSA PRIVATE KEY-----`

Element	Type	Description
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 details of the `meta` object, see Message format on page 9.

```
{
       "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/emailA
ddress=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 36.

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 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.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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+GdHpgoaL8YsydxJBsYg9dYTD
zVVtYFVvdCVzBdcwCjyOuPwtZoYU3pLEFQ7OVoDUDPmVnl6idS/6tB2m89I5zdc02xUeCWTBpTGoOhNtc+YD
mxPGZ1FQIpXCw0MT91jviWm3JydDd5YKINwvdTh8zsRT/702ZD9uZslwkQA/b2B9/hidCAkQkvs5H1B3o4la
Td0JE9k90N+qbaQjVvoInr+jdXaWvrScwFVxZhb7Q1LvUL6oxW889bOWFMSa+/mnENarw6rpwfk9Ayi5uQQ2
imY/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.

Element		Type	Description
	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": {
        "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.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"`
		`}`
		`},`

`port`	integ-	The port number where this local service accepts

Element	Type	Description
	er	connections.
public_keys	list	Lists the public keys that can be used to authenticate on SPS. For example: `"public_keys": [ { "comment": "user@example.com anothercomment", "key": { "selection": "rsa", "value": "AADDB3NzaC1yc2EABBADAQA...../DuhON" } }, { "comment": "username@example.com", "key": { "selection": "rsa", "value": "ASFDFAB3NzaC1yc2EAAAABIwAAASdfASF/EuQh9zc2umxX...dU=" } } ]` 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": "ASFDFAB3NzaC1yc2EAAAABIwAAASdfASF/EuQh9zc2umxX...dU=" }`
selec-	rsa	The type of the public key. Must be `rsa`.

Elements of public_keys		Type	Description
	tion		
	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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

RPC API

The SPS RPC API allows you to access, query, and manage SPS from remote applications. You can access the API using the Simple Object Access Protocol (SOAP) protocol over HTTPS, meaning that you can use any programming language that has access to a SOAP client to integrate SPS to your environment.

URL

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

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

Cookie name	Description	Required	Values
			details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 RPC API settings.

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

Response

The following is a sample response received when listing the RPC API settings.

For details of the `meta` object, see Message format on page 9.

```
{
    "body": {
       "enabled": true
    },
    "key": "soap",
    "meta": {
       "first": "/api/configuration/management/certificates",
       "href": "/api/configuration/management/soap",
       "last": "/api/configuration/management/webinterface",
       "next": "/api/configuration/management/syslog",
       "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 RPC API configuration options.
enabled	boolean	Set to `true` to enable the RPC API.

Modify RPC API settings

To modify the RPC API settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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

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

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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.

CAUTION:

Accessing the One Identity Safeguard for Privileged Sessions (SPS) host directly using SSH is not recommended or supported, except for troubleshooting purposes. In such case, the One Identity Support Team will give you exact instructions on what to do to solve the problem.

For security reasons, disable SSH access to SPS when it is not needed. For details, see "Enabling SSH access to the One Identity Safeguard for Privileged Sessions (SPS) host" in the Administration Guide.

The following encryption algorithms are configured on the local SSH service of SPS:

- Key exchange (KEX) algorithms:

```
diffie-hellman-group-exchange-sha256
```

- Ciphers:

```
aes256-ctr,aes128-ctr
```

- Message authentication codes:

```
hmac-sha2-512,hmac-sha2-256
```

URL

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

Cookies

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 9.


```
{
        "body": {
                "customer": "Example",
                "limit": 5000,
                "limit_type": "host",
                "serial": "b937d212-db7d-0f2f-4c87-295e3c57024a",
                "valid_not_after": "2018-11-07",
                "valid_not_before": "2017-11-06"
        },
        "key": "license",
        "meta": {
                "first": "/api/configuration/management/certificates",
                "href": "/api/configuration/management/license",
                "last": "/api/configuration/management/webinterface",
                "next": "/api/configuration/management/root_password",
                "parent": "/api/configuration/management",
                "previous": "/api/configuration/management/health_monitoring",
                "remaining_seconds": 600,
                "transaction": "/api/transaction",
                "upload": "/api/upload/license"
        }
 }
```


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.
customer	string	The company permitted to use the license (for example, `Example Ltd.`).
limit	integer	The actual value of the session or host limit (see `limit_type`).
limit_type	host | session	• `host`: Limits the number of servers (individual IP addresses) that can be connected through SPS.

Element		Type	Description
			• `session`: Limits the number of concurrent sessions (parallel connections) that can pass through SPS at a time (for example 25). SPS will reject additional connection requests until an already established connection is closed.
	serial	string	The unique serial number of the license.
	valid_not_after	date	The date when the license expires. The dates are displayed in YYYY/MM/DD format.
	valid_not_before	date	The date after which the license is valid. The dates are displayed in YYYY/MM/DD 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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

3. **Upload the license file.**

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

```
curl --cookie cookies -F 'data=@/path/license.txt' -H "Expect:" --insecure
https://<IP-address-of-SPS>/api/upload/license
```

4. **Restart the traffic on SPS.**

   SPS will not use the new license to ongoing sessions. For the new license to take full effect, you must restart all traffic on the **Basic Settings > System > Traffic control** page of the SPS web interface.

```
curl --cookie cookies -F 'data=@/path/license.txt' -H "Expect:" --insecure
https://<IP-address-of-SPS>/api/upload/license
```

5. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 RPC API settings.

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

Response

The following is a sample response received when querying the endpoint.

For details of the `meta` object, see Message format on page 9.

```
{
        "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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the endpoints for SNMP configuration settings.

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

Response

The following is a sample response received when querying the endpoint.

For details of the `meta` object, see Message format on page 9.


```
{
 "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 Identity

Elements of `remote_desktop_gateway`	Type	Description

		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....\nr8lDCPoq\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 details, see Commit a transaction on page 30.

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 SiemMessage.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the endpoints for SNMP configuration settings.

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

Response

The following is a sample response received when querying the endpoint.

For details of the `meta` object, see Message format on page 9.

```
{
        "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.
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.

Elements	Type	Description
		• `cef`: if using CEF-compatible SIEMs, for example, Microsoft 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": "192.168.1.1" },`
		`"address": {`

Elements of `protocol`		Type	Description
			`"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_ref": "1241814345d074efd1ded7" }`
			`"tls": { "selection": "disabled" }`

Elements of `protocol`		Type	Description
`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` 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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Manage Safeguard for Privileged Sessions clusters

When you have a set of two or more 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 Safeguard for Privileged Sessions 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 **AAA > Access Control** 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
	"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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

Promote a Safeguard for Privileged Sessions node to be the Central Management node in a new cluster

You can build a cluster by promoting a Safeguard for Privileged Sessions 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 details, see Open a transaction on page 28.

2. **Create the Central Management node.**

   POST an empty request to the `https://<IP-address-of-node-to-become-Central-Management-node>/api/cluster/promote` endpoint.

   The following is a sample response received.

   For details of the `meta` object, see Message format on page 9.

```
{
        "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 details, see Commit a transaction on page 30.

Join node(s) to the cluster

Once you have a Central Management Safeguard for Privileged Sessions 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 details, see Open a transaction on page 28.

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 details of the `meta` object, see Message format on page 9.

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

Elements	Type	Description
		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","roles":
[]}'
```

   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 30.

Query join status

To find out whether a node has been joined to a cluster, complete the following steps.

1. Query the `/api/cluster/join_request` endpoint on the node whose join status you want to figure out.

```
curl GET --cookie cookies https://<IP-address-of-node-to-be-queried>/api/cluster/join_request
```

   The following is a sample response received.

   For details of the `meta` object, see Message format on page 9.

```
      "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.

Elements	Type	Description
		• `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 details, see Open a transaction on page 28.

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.

Role	Description
`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 details, see Commit a transaction on page 30.

Query nodes

To list the nodes available in a cluster, complete the following steps.

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

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

   The following is a sample response received.

For details of the `meta` object, see Message format on page 9.

```
{
        "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 details of the `meta` object, see Message format on page 9.

   ```
   {
         "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 details of the `meta` object, see Message format on page 9.


```
{
  "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": {
        "last_updated": "2018-02-08T09:59:00Z",
        "last_checked": "2018-02-08T09:59:00Z",
        "downloaded_xml_hash": "2853830f4aa0a90a63e75bab1b22e513",
        "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	Hard disk use

Elements		Type	Description
		point integer (percent)	
	swap	floating point integer (percent)	Swap use
	load1	floating point integer	The average system load during the last one minute. The values mean the following: • < 1: low system load • 1-5: high system load • > 5: extremely high system load
	load5	floating point integer	The average system load during the last five-minute period. The values mean the following: • < 1: low system load • 1-5: high system load • > 5: extremely high system load
	load15	floating point integer	The average system load during the last fifteen-minute period. The values mean the following: • < 1: low system load • 1-5: high system load • > 5: extremely high system load

	sessions	string	The protocol type and the number of ongoing sessions. For example:
			`"sessions": {`
			`  "ssh": 3,`
			`  "rdp": 4`
			`},`

	total_	integer	The total number of ongoing

Elements		Type	Description
	sessions	(number of)	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 configuration 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.

Elements			Type	Description
	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)	
		downloaded_xml_hash	string	The hash of the latest downloaded configuration file (used for configuration synchronization). If no configuration file has been downloaded yet, it says null.
		last_updated	string	The last time the node's configuration was synchronized, in ISO 8601 format.
		last_checked	string	The last time the node attempted to fetch a new configuration, in ISO 8601 format.
		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)	

Elements of issues	Type	Description
`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 details of the `meta` object, see Message format on page 9.

For details of the other objects, see tables Cluster status details and "issues" object details.

```
{
    "fqdn": "managed-host.cluster",
    "key": "46f97a58-4028-467d-9a22-9cfe78ae3e1c",
    "configuration_sync": {
        "downloaded_xml_hash": "2853830f4aa0a90a63e75bab1b22e513",
        "last_updated": "2018-02-08T09:59:30Z",
        "last_checked": "2018-02-08T09:59:30Z",
        "issues": {}
    },
    "health_status": {
        "memory": 62.5,
        "disk": 1.9,
        "swap": 0,
        "load1": 0.53,
        "load5": 0.68,
        "load15": 0.37,
        "sessions": {
            "ssh": 3,
            "rdp": 4
        },
```

```
        "total_sessions": 7
    },
    "sync_status": "up-to-date",
    "last_seen": "2018-02-08T10:00:00Z",
    "meta": {
        "configuration": "/api/cluster/nodes/46f97a58-4028-467d-9a22-
9cfe78ae3e1c",
        "href": "/api/cluster/status/46f97a58-4028-467d-9a22-9cfe78ae3e1c"
    }
}
```


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 mean the following: • < 1: low system load

Elements		Type	Description
			• 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 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.

Elements		Type	Description
			• `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.
`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)	
	`downloaded_xml_hash`	string	The hash of the latest downloaded configuration file (used for configuration synchronization). If no configuration file

Elements		Type	Description
			has been downloaded yet, it says `null`.
	`last_updated`	string	The last time the node's configuration was synchronized, in ISO 8601 format.
	`last_checked`	string	The last time the node attempted to fetch a new configuration, in ISO 8601 format.
	`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.
	`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 as-is.
- Tailor certain parts of the central configuration to specific needs of individual nodes in the cluster (for example, your nodes may access external services at 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 SPS through the REST API. In other words, whatever you can configure through the REST API, you can configure the exact same settings using the plugin. One notable difference between the REST API and the plugin is that using the REST API, you can only read certain types of data (such as keys and passwords), while using the configuration synchronization plugin, you can write these types of data as well.

Data structures in the plugin are represented as nested JSON objects. For object references, the plugin uses keys.

The plugin works with the following key parameters:

- `local_config`: The current configuration of a Managed Host node (those parts that can be configured through the REST API).
- `merged_config`: The configuration of the Central Management node that is about to be synced to the Managed Host node (those parts that can be configured through the REST API), with settings related to networking, local services, management, and the license of SPS whitelisted. These settings are never overwritten by configuration synchronization.
- `node_id`: The unique ID of the Managed Host node in the cluster (you can retrieve this identifier by querying the `/api/cluster/nodes` endpoint through the REST API).
- `plugin_config`: The configuration of the plugin provided as free-form text. Specifying the configuration of the plugin is optional. It enables you to run configuration synchronization on each cluster with different parameters if you have multiple clusters.

Example: Customizing an IP address in a connection policy

For example, an RDP connection policy on a Managed Host node specifies the following client and target addresses:

```
$ 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"
                            ]
                },
        },
        ...
 }
```

Let's suppose that on the Central Management node, an RDP connection policy is configured with these details:

```
$ 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`.

In case you need assistance with writing customized transformations, contact our Professional Services Team, and a One Identity Service Delivery Engineer will be able to help you.

NOTE:

Configuration settings related to networking (`/api/configuration/network`), local services (`/api/configuration/local_services`), and the management of SPS (`/api/configuration/management`) are not overwritten on the nodes by configuration synchronization even when not using a plugin.

To upload a configuration synchronization plugin to the Central Management node, complete the following steps.

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details of the `meta` object, see Message format on page 9.

   ```
   {
       "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.
`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 details, see Commit a transaction on page 30.

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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

6

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`. For details on authentication, see Authenticate to the

Cookie name	Description	Required	Values
			SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
  "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 RDP-specific channels, see RDP channels on page 419. • For details on SSH-specific channels, see SSH channels on page 460.

Element	Type	Description
		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:
		`"clients": [`
		`"192.168.1.1/24",`
		`"2001:db8:85a3::8a2e:0:0/32"`

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

Element	Type	Description
		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 338.
`remote_groups`	list	You can control channel access during authentication to the remote server with blacklists or whitelists of 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 338.

`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"`

`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 324. 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available endpoints.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
  "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

Endpoint	Description
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.
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 36.

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
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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

Element		Type	Description
			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, 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.

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`.
	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 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

Elements of server	Type	Description
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the audit policies.

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

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/policies/audit_
policies/<policy-id>
```

Response

The following is a sample response received when listing audit policies.

For details of the `meta` object, see Message format on page 9.

```
{
  "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 (string)	The configuration elements of the policy.
	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

Element	Type	Description
		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_for_upstream`			Top level item	Configures encrypting upstream traffic separately.
	`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.

Elements of `encryption`	Type	Description
		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\nQ0ExEDAOBgNVBAgT
B09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW
50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4X
DTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP\nbnRh
cmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQQLEw1Eb2N1bWVudGF0aW
9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74
ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw\n0qNRjPS1ML
zjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc\n3mV2zkvtD61CXzbezAVdArDl6Wny
sRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3F
y2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjML
```

```
U2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp
5USme4PmNhY6QfUlgjsFjPfoVG7XDB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C
8xstRSEG7oJc\nN5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyy
D1ld89OUEYZ+aJQd/A==\n-----END CERTIFICATE-----\n"
```


Add an audit policy

To add an audit policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify an audit policy

To modify an audit policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the properties of a specific Backup policy object.

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

Response

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

For details of the `meta` object, see Message format on page 9.


```
{
  "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 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.

Element		Type	Description
	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. 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 210. For example:

Element	Type	Description

		`"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`.
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

Elements of `authentication`	Type	Description
		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`	T-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 AAAAB3NzaC1yc2EAAAADAQABAAABAQDmIDa1PuJFzgvZvPs9hzgvMd/9WIn4J7RBFuO769g/OgTvCRTgrF8IM/0iN0YzcUM3IGyPnJ1OlLE2Gb6CxVvEcjP6pme7JroAWo039wQHR3Rxl1KoEmC+0EOImQycIdAS7grWNwD2VB2S7iyFErZhqRxhGJPKbR/kF3lQ3dGtt3pr4+R6wnU9lZ7RSETfB+N09FE4f5Nqy+VEShgdc66ElFRXXVilmiTnIMAyim3T7UVNgRdZYIUAZ79tkyTp6I+DZ7k7BG9TYwdBjhwr0eVL56ILxpXylpzWONuMhHxLKsL42NfmeagjVUD1CJVOrfaGjCVGEeS3iQs6GVVxe78n"` When querying, the public key of the host will always be displayed in the `selection` and `value` elements.	o- p l- e- v- e- l e- l- e- m- e- n- t o- r s- t- r- i-

Elements of host_key	Type	Description
	ng	
selection	string (dsa | dss | rsa) -	The algorithm the key is based on.
value	string	The public key of the host.

Elements of host_key	Type	Description
	ng-	

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+FtgS783VKE1TVZBtDQlsXJ9FXu
6KNBvqvSAjcXiWY+izqn+P14UVRY1vOdz7WwLIWOUoTKHfPMqv3bdjwM4Bhd26POWSFyDf46yx1Yzv
Mwgc="
      },
      "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 token of the user	Required	The value of the session ID cookie received from the REST server in the authentication response, for example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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`

Element	Type	Description
		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`
`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",`

Element	Type	Description
		`"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.

`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.

Element	Type	Description
		• 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. **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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a content policy

To modify a content policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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.

Element	Type	Description
		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.
`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.

Element	Type	Description
		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_`	string	Must be used if the `enabled` element is set to

Elements of `encryption`		Type	Description
	`identity`		`true`. References the identifier of the X.509 certificate stored on SPS. You can configure X.509 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).
`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_groups`	boolean	Must be used if the `selection` element is set to `ad`.

Elements of schema		Type	Description
			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": "groupOfUniqueNames", "attribute": "uniqueMember" } ]`
	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\nQ0ExEDAOBgNVBAgT
B09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW
50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4X
DTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP\nbnRh
cmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQQLEw1Eb2N1bWVudGF0aW
9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74
ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw\n0qNRjPS1ML
zjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc\n3mV2zkvtD61CXzbezAVdArDl6Wny
sRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3F
y2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjML
U2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp
5USme4PmNhY6QfUlgjsFjPfoVG7XDB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C
8xstRSEG7oJc\nN5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyy
D1ld89OUEYZ+aJQd/A==\n-----END CERTIFICATE-----\n"
```


Add an LDAP server

To add an LDAP server, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify an LDAP server

To modify the configuration of an LDAP server, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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

Code	Description	Notes
		retrieved.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the configured signing Certificate Authorities (CAs).

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

The following command retrieves the properties of a specific policy.

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

Response

The following is a sample response received when listing signing CAs.

For details of the `meta` object, see Message format on page 9.

```
{
    "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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a signing CA

To modify a signing CA, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
400	Bad Request `"message": "Signing certificate is not`	The referenced certificate is not a valid CA certificate.

Code	Description	Notes
	`CA;`	
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.
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 user	Required	The value of the session ID cookie received from the REST server in the authentication response, for example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
  "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" ], [ "19:00", "22:00" ] ]`

Element	Type	Description
Sat	list	
Sun	list	
Thu	list	
Tue	list	
Wed	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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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.
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

Element		Type	Description
			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.
`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\nQ0ExEDAOBgNVBAgT
B09udGFyaW8xEDAOBgNVBAcTB1Rvcm9udG8xEDAOBgNVBAoT\nB0JhbGFiaXQxFjAUBgNVBAsTDURvY3VtZW
50YXRpb24xEDAOBgNVBAMTB2JhbGFi\naXQxIDAeBgkqhkiG9w0BCQEWEWNhdGFpbEBiYWxhYml0Lmh1MB4X
DTE2MDQyMjE2\nMDAyNloXDTE3MDQyMjE2MDAyNlowgY8xCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdP\nbnRh
cmlvMRAwDgYDVQQHEwdUb3JvbnRvMRAwDgYDVQQKEwdCYWxhYml0MRYwFAYD\nVQQLEw1Eb2N1bWVudGF0aW
9uMRAwDgYDVQQDEwdiYWxhYml0MSAwHgYJKoZIhvcN\nAQkBFhFjYXRhaWxAYmFsYWJpdC5odTCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCC\nAQoCggEBAOGa9I2jmVlVdVWEI/Wy7ahTeyaIjK52FQUXqxG8okOSD+nV74
ZFUuiS\n59X+2Ow1aDqVGrDMgPNhSVpYXUvDUAUOILJW4rAIoxDY6vDU9/4v9dDiQfEPlauw\n0qNRjPS1ML
zjSOQDSKqPkdivkS6HKZeX3+TFq3OxO+vIrF9zFfp9T+eDG2oSobPc\n3mV2zkvtD61CXzbezAVdArDl6Wny
sRyzxyH8WEhFwZepWxFD9Y5N1dzKody7Hncs\nX5kVIv0+Z6bBHfg/7wHWysJdwNuLr0ByTOvPM6WdA83k3F
y2gYNk7Rc0BbRFbQTX\nhJVfUzSUWHVhFQtAb4diKU5voqepfNMCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEA
\nR5DIwOHsEKoGkiI3cHC2VMnxP2rRhpTneh6El+DFnQPdjrXa+tnqV4TdnNaD+FvP\nAB1kqbmC4hJAsjML
U2b1ne6m+SLmzhRuMxcA6x+fnYvcQT57IbRdq2E/4oJGeyuy\n0jQE+nmoVD3lDytIOxCfQvZhl1tcbBE5hp
5USme4PmNhY6QfUlgjsFjPfoVG7XDB\nuNaUoWS6RvZPmL5IuvF9tqe96ES6DTjC8rBfQYvSoVNjjPnUMx0C
8xstRSEG7oJc\nN5+4ImYnFNxSG20hZpFy0OFDf2g7Fx+W50/NtXamUF1Sf8WlPZc03oVl1/Fzo7mt\nqYyy
D1ld89OUEYZ+aJQd/A==\n-----END CERTIFICATE-----\n"
```


Add a trusted CA

To add a trusted CA, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a trusted CA

To modify a trusted CA, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists local user databases.

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

The following command retrieves the properties of a specific local user database.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/configuration/policies/user_
databases/<object-id>
```

Response

The following is a sample response received when listing local user databases.

For details of the `meta` object, see Message format on page 9.


```
{
    "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.

Element		Type	Description
			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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a local user database

To modify a local usre database, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 270.

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`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a user list

To modify a user list, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "items": [
       {
          "key": "channel_policies",
          "meta": {
             "href": "/api/configuration/http/channel_policies"
          }
       },
       {
          "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
`channel_policies`	List of the default and custom channel 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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
  "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, string	List of target IP addresses.
	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

Element		Type	Description
			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 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

Element		Type	Description
			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.
`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

Element		Type	Description
			`api/configuration/http/options` endpoint.
	`enabled`	boolean	Set to true to enable periodical cleanup of the connection metadata.
`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 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.

Element		Type	Description
			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 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	Type	Description
		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
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 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

Elements of `server_address`			Type	Description
				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 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

Elements of `server_address`	Type	Description
		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"`
		`}`

`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 213.

Elements of `web_proxy.transport_security`	Type	Description

		`"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 320.
		
		`"host_certification_method": { "selection": "generate", "signing_ca": "1904188625a843f11d30a5" },`

	`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 213.
	`signing_ca`	string	Reference the Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 320.

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-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": {`

Elements of body.transport_security		Type	Description

			`"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"`
			`},`

	`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 213.
			
			`"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 320. `"host_certification_method": { "selection": "generate", "signing_ca": "1904188625a843f11d30a5" },`
	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 213.
	signing_ca	string	Reference the Signing CA that you have already configured on SPS. For details, see Signing CA policies on page 320.

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.

Elements of `access_control`		Type	Description
`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` 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 36.

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 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/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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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/http/authentication_policies
```

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/http/authentication_policies<object-id>
```

Response

The following is a sample response received when listing HTTP authentication policies.

For details of the `meta` object, see Message format on page 9.


```
{
    "items": [
       {
          "key": "-200",
          "meta": {
             "href": "/api/configuration/ssh/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`

Element		Type	Description
			Uses the LDAP server 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 `certificate`, `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.
	`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_`

Element		Type	Description
			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 (second-s)	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).

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify an HTTP authentication policy

To modify an HTTP authentication policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists global HTTP options.

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

Response

The following is a sample response received when listing global HTTP options.

For details of the `meta` object, see Message format on page 9.

```
{
  "body": {
    "audit": {
      "cleanup": {
        "enabled": false
      },
      "timestamping": {
        "selection": "local",
```

```
        "signing_interval": 30
      }
    },
    "service": {
      "enabled": true,
      "log_level": 4
    }
  },
  "key": "options",
  "meta": {
    "first": "/api/configuration/http/channel_policies",
    "href": "/api/configuration/http/options",
    "last": "/api/configuration/http/settings_policies",
    "next": "/api/configuration/http/settings_policies",
    "parent": "/api/configuration/http",
    "previous": "/api/configuration/http/channel_policies",
    "transaction": "/api/transaction"
  }
}
```


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.
audit	Top level item	Contains settings for timestamping and cleanup.
service	Top level item	Global setting to enable HTTP connections, and specify the logging detail.
enabled	boolean	Set to `true` to enable HTTP connections.
log_level	int	Defines the logging detail of HTTP connections.

Elements of `audit`	Type	Description
cleanup	Top level item	Global retention settings for HTTP connection metadata. To configure retention time for a specific connection policy, use the `archive_cleanup_policy` element at the endpoint of the policy instead.
channel_	int	Only if `enabled` is set to `true`.

Elements of `audit`	Type	Description		
	`database_cleanup_days`			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 this element to `true`.
`timestamping`			Top level item	Global timestamping settings for HTTP connections.
	`selection`		string	Configures local or remote timestamping. • Set `local` to use SPS for timestamping. • Set `remote` to configure a remote timestamping server.
	`server_url`		string	Required for `remote` timestamping. The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.
	`oid`		Top level item	The Object Identifier of the policy used for timestamping.
		`enabled`	boolean	Required for `remote` timestamping. Set to `true` to configure the Object Identifier of the timestamping policy on the timestamping remote server.
		`policy_oid`	string	Required if the `oid` is `enabled`. The Object Identifier of the timestamping policy on the remote timestamping server.
	`signing_interval`		int	Time interval for timestamping open connections, in seconds.

Examples:

Set SPS as the timestamping server:

```
{
    "audit": {
       "cleanup": {
          "enabled": false
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Enable cleanup, and set it to occur every 10 days:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Change timestamping to a remote server, without specifying a timestamping policy:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
        "oid": {
```

```
          "enabled": false
        },
        "selection": "remote",
        "server_url": "<url-of-timestamping-server>",
        "signing_interval": 30
      }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, and specify the `1.2.3` timestamping policy:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
             "oid": {
                "enabled": true,
                "policy_oid": "1.2.3"
             },
             "selection": "remote",
             "server_url": "<url-of-timestamping-server>",
             "signing_interval": 30
          }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Modify global HTTP settings

To modify global HTTP settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

2. **Modify the JSON object of the global HTTP settings endpoint.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/http/options` endpoint. You can find a detailed description of the available parameters listed in Element . The elements of the `audit` item are described in Elements of audit.

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
  "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"
    }
 }
```


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

Element	Type	Description
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.

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.

Elements of client_tls_security_settings and server_tls_security_settings	Type	Description
		• `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 in order 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 will only offer TLS version 1.2 during negotiation. This is the recommended setting. • `TLSv1_1`: this setting will offer TLS version 1.1 and later versions during negotiation. • `TLSv1_0`: this setting will offer TLS version 1.0 and later versions during negotiation.

Add HTTP settings policies

To add a settings policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify HTTP settings policies

To modify a settings policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "items": [
       {
          "key": "channel_policies",
          "meta": {
             "href": "/api/configuration/ica/channel_policies"
          }
       },
       {
          "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.
`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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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.

Channel	Special options	Description
`CTXTW`	Yes	**Drawing (Thinwire)**: Enables access to the server's desktop (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.

Channel	Special options	Description

		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.
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

Channel	Special options	Description
		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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection

Cookie name	Description	Required	Values
			between the REST client and the SPS 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 global ICA options.

For details of the `meta` object, see Message format on page 9.

```
{
    "body": {
       "audit": {
          "cleanup": {
             "enabled": false
          },
          "timestamping": {
             "selection": "local",
             "signing_interval": 30
          }
       },
       "service": {
          "enabled": true,
          "log_level": 4
       }
    },
    "key": "options",
    "meta": {
       "first": "/api/configuration/ica/channel_policies",
       "href": "/api/configuration/ica/options",
       "last": "/api/configuration/ica/settings_policies",
       "next": "/api/configuration/ica/settings_policies",
       "parent": "/api/configuration/ica",
       "previous": "/api/configuration/ica/channel_policies",
       "transaction": "/api/transaction"
    }
 }
```

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.
audit	Top level item	Contains settings for timestamping and cleanup.
service	Top level item	Global setting to enable ICA connections, and specify the logging detail.
enabled	boolean	Set to true to enable ICA connections.
log_level	int	Defines the logging detail of ICA connections.

Elements of audit	Type	Description
cleanup	Top level item	Global retention settings for ICA connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.
channel_database_cleanup_days	int	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 this element to true.
timestamping	Top level item	Global timestamping settings for ICA connections.
selection	string	Configures local or remote timestamping. • Set local to use SPS for timestamping. • Set remote to configure a remote timestamping server.
server_url	string	Required for remote timestamping. The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.

Elements of `audit`		Type	Description
	`oid`	Top level item	The Object Identifier of the policy used for timestamping.
	`enabled`	boolean	Required for `remote` timestamping. Set to `true` to configure the Object Identifier of the timestamping policy on the timestamping remote server.
	`policy_oid`	string	Required if the `oid` is `enabled`. The Object Identifier of the timestamping policy on the remote timestamping server.
	`signing_interval`	int	Time interval for timestamping open connections, in seconds.

Examples:

Set SPS as the timestamping server:

```
{
    "audit": {
       "cleanup": {
          "enabled": false
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Enable cleanup, and set it to occur every 10 days:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
      "selection": "local",
      "signing_interval": 30
```

```
    }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
             "oid": {
                "enabled": false
             },
             "selection": "remote",
             "server_url": "<url-of-timestamping-server>",
             "signing_interval": 30
          }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Change timestamping to a remote server, and specify the `1.2.3` timestamping policy:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
        "oid": {
          "enabled": true,
          "policy_oid": "1.2.3"
        },
        "selection": "remote",
        "server_url": "<url-of-timestamping-server>",
        "signing_interval": 30
      }
  },
```

```
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Modify global ICA settings

To modify global ICA settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

2. **Modify the JSON object of the global ICA settings endpoint.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/ica/options` endpoint. You can find a detailed description of the available parameters listed in Element . The elements of the `audit` item are described in Elements of audit.

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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",
       "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 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. Note that the SPS web UI displays the same value in milliseconds.

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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify ICA settings policies

To modify a settings policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

Code	Description	Notes
201	Created	The new resource was successfully created.
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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

9

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available settings for configuring for RDP connections.

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

Response

The following is a sample response received when listing the configuration settings. For details of the `meta` object, see Message format on page 9.


```
{
    "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists RDP connection policies.

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

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/rdp/connections/<connection-key>
```

Response

The following is a sample response received when listing RDP connection policies.

For details of the `meta` object, see Message format on page 9.

```
{
    "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"
        },
        "server_certificate_check": {
            "enabled": false
        },
        "source_address": {
            "selection": "box_address"
        },
        "transport_security": {
            "certificate": {
                "selection": "self_signed"
            },
            "legacy_fallback": false,
            "selection": "tls"
        },
        "web_gateway_authentication": {
            "enabled": false
        }
    },
    "key": "12932832285a830b4d2f5d7",
    "meta": {
        "first": "/api/configuration/rdp/connections/12932832285a830b4d2f5d7",
        "href": "/api/configuration/rdp/connections/12932832285a830b4d2f5d7",
        "last": "/api/configuration/rdp/connections/12932832285a830b4d2f5d7",
        "next": null,
        "parent": "/api/configuration/rdp/connections",
        "previous": null,
        "remaining_seconds": 600,
        "transaction": "/api/transaction"
    }
}
```

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).
		priority	int	Specifies the indexing priority for the connection. Possible values are:

Element			Type	Description
				• 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, integers	List of target ports.
		targets	list, string	List of target IP addresses.
	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	string	References the identifier of the analytics

Element		Type	Description
			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 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

Element		Type	Description
			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/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). 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

Element		Type	Description
			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": {`
			`"enabled": false`
			`},`

server_address		Top level item	Defines the address where the clients connect to.

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": {`
			`"enabled": true,`
			`"trusted_ca":`
			`"9106862955a844051d7bf6"`
			`},`

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

Element		Type	Description
			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:** 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:**

Element		Type	Description

			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 213.
			`"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 320.
			`"certificate": {`
			`"selection": "generate",`
			`"signing_ca": "1904188625a843f11d30a5"`
			`},`

	`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!**

Element		Type	Description

	`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, 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` and `legacy_fallback` options.
			`"transport_security": {`
			`"certificate": {`
			`"selection": "self_signed"`
			`},`
			`"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.

Element		Type	Description
	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 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.
host_certification_method	JSON object	To act as a Remote Desktop Gateway, SPS needs to display a certificate to the clients. • To display always the same certificate, set "selection": "single", and reference a X.509 certificate and the matching private key. For example: `"host_certification_method": { "selection": "signing", "value": { "signing_ca": "53449998258a4ceba80fdc" }, "common_name": "examplecn" }` For details on uploading certificates, see Certificates stored on SPS on page 213.

Elements of `remote_desktop_gateway`			Type	Description
				• To automatically create new certificates on SPS for every client, set `"selection": "signing"`, and reference the Certificate Authority (CA) to sign the generated certificates. For example: `"host_certification_method": { "selection": "single", "value": "1904188625a843f11d30a5" },` For details on creating a signing CA, see Signing CA policies on page 320.
	selection		single | signing	Determines if SPS displays the same certificate to every client (`single`), or generates a separate certificate (`signing`) 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 `signing`, this is a JSON object. • If `selection` is set to `single`, 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 `signing`. 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`

Elements of `remote_desktop_gateway`			Type	Description
		signing_ca	string	Available only if `selection` is set to `signing`. Contains the reference key of the signing CA used to sign the certificates that SPS shows to the clients. For example: `"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": { "selection": "local_user_database", "value": { "domain": "example", "local_user_database": "15646962145a843f758501d" } }`
				• 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`.

Elements of `remote_desktop_gateway`		Type	Description
			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.

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 36.

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.

Code	Description	Notes
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify an RDP connection policy

To modify an RDP connection policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

RDP channels

The available RDP channel types and their functionalities are described below.

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: • `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	Special options	Description
		Channel-specific actions: • `log_transfer_to_db` (true|false): Make the list of file operations available in the **Search > File operations** column of the SPS web interface • `log_transfer_to_syslog` (true|false): Send the file operations into the system log Channel-specific access control rules: • `devices` (list): To permit only specific redirections, list the unique name of the redirection in this field. Leave it empty to permit access to every redirection available.
`rdpsnd`	None	**Sound**: Enable access to the sound device of the server.
`customs`	Yes	**Custom**: Applications can open custom channels to the clients connecting remotely to the server. Enabling the **Custom** channel allows the clients to access all of these custom channels. To permit only specific channels, list the unique names of the channels into the `customs` field. For example, to monitor RemoteApp connections, you need to configure custom channels. For more information, see "Configuring RemoteApps" in the Administration Guide. Channel-specific access control rules: • `customs` (list): To permit only specific custom channels, list the unique name of the channels in this field. Leave it empty to permit access to every custom channel available.
`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-`	Yes	**Serial redirect**: Enables access to serial-port redirections. To

Channel	Special options	Description
`serial`		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 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

Channel	Special options	Description
		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

To 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. If you cannot or do not want to join SPS to the domain, see "Network Level Authentication without domain membership" in the Administration Guide.

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	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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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.

Element	Type	Description
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
    "body": {
       "audit": {
          "cleanup": {
             "enabled": false
          },
          "timestamping": {
             "selection": "local",
             "signing_interval": 30
          }
       },
       "service": {
          "enabled": true,
          "log_level": 4
       }
    },
    "key": "options",
    "meta": {
       "first": "/api/configuration/rdp/channel_policies",
       "href": "/api/configuration/rdp/options",
       "last": "/api/configuration/rdp/settings_policies",
       "next": "/api/configuration/rdp/settings_policies",
       "parent": "/api/configuration/rdp",
       "previous": "/api/configuration/rdp/domain_membership",
       "transaction": "/api/transaction"
    }
 }
```


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.
audit	Top level	Contains settings for timestamping and cleanup.

Element	Type	Description
	item	
service	Top level item	Global setting to enable RDP connections, and specify the logging detail.
enabled	boolean	Set to true to enable RDP connections.
log_level	int	Defines the logging detail of RDP connections.

Elements of audit	Type	Description
cleanup	Top level item	Global retention settings for RDP connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.
channel_database_cleanup_days	int	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 this element to true.
timestamping	Top level item	Global timestamping settings for RDP connections.
selection	string	Configures local or remote timestamping. • Set local to use SPS for timestamping. • Set remote to configure a remote timestamping server.
server_url	string	Required for remote timestamping. The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.
oid	Top level item	The Object Identifier of the policy used for timestamping.
enabled	boolean	Required for remote timestamping. Set to true to configure the Object Identifier

Elements of `audit`	Type	Description
		of the timestamping policy on the timestamping remote server.
`policy_oid`	string	Required if the `oid` is `enabled`. The Object Identifier of the timestamping policy on the remote timestamping server.
`signing_interval`	int	Time interval for timestamping open connections, in seconds.

Examples:

Set SPS as the timestamping server:

```
{
    "audit":  {
       "cleanup":  {
          "enabled":  false
       },
       "timestamping":  {
          "selection":  "local",
          "signing_interval":  30
       }
    },
    "service":  {
       "enabled":  true,
       "log_level":  4
    }
 }
```

Enable cleanup, and set it to occur every 10 days:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
      "selection": "local",
      "signing_interval": 30
    }
  },
```

```
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
             "oid": {
                "enabled": false
             },
             "selection": "remote",
             "server_url": "<url-of-timestamping-server>",
             "signing_interval": 30
          }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Change timestamping to a remote server, and specify the `1.2.3` timestamping policy:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
        "oid": {
          "enabled": true,
          "policy_oid": "1.2.3"
        },
        "selection": "remote",
        "server_url": "<url-of-timestamping-server>",
        "signing_interval": 30
      }
  },
```

```
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Modify global RDP settings

To modify global RDP settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

2. **Modify the JSON object of the global RDP settings endpoint.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/rdp/options` endpoint. You can find a detailed description of the available parameters listed in Element . The elements of the `audit` item are described in Elements of audit.

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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",
    "permit_unreliable_usernames": true,
    "preconnect_channel_check": true,
    "protocol_features": {
      "nla": {
        "enabled": true,
        "require_domain_membership": true
      },
      "rdp4_auth_enabled": true,
      "rdp4_enabled": true,
      "rdp5_enabled": true
```

```
    },
    "screen": {
      "maximum_bpp": 32,
      "maximum_height": 2000,
      "maximum_width": 2000
    },
    "timeout": 600,
    "userauth_banner": "Click 'OK' to log in."
  },
  "key": "13298899495727c51f725cf",
  "meta": {
    "first": "/api/configuration/rdp/settings_policies/-301",
    "href": "/api/configuration/rdp/settings_policies/13298899495727c51f725cf",
    "last": "/api/configuration/rdp/settings_policies/13298899495727c51f725cf",
    "next": null,
    "parent": "/api/configuration/rdp/settings_policies",
    "previous": "/api/configuration/rdp/settings_policies/-303",
    "transaction": "/api/transaction"
  }
}
```


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	Settings for RDP protocol versions, and Network Layer Authentication.

Element	Type	Description
	item	
screen	Top level item	Display size and depth settings.
timeout	int	Connection timeout, in seconds. Note that the SPS web UI displays the same value in milliseconds.
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.
rdp4_auth_enabled	boolean	Set to true to enable RDP4 authentication within the RDP5 protocol. This might be needed for compatibility reasons with certain client applications.
rdp4_enabled	boolean	Set to true to enable the version 4 of the Remote Desktop Protocol.
rdp5_enabled	boolean	Set to true to enable the version 5 of the Remote Desktop Protocol. To also configure SSL-encryption for RDP5, enable the nla element, or configure a Signing CA in your connection policies.

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
       },
       "rdp4_auth_enabled": true,
       "rdp4_enabled": true,
       "rdp5_enabled": 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
       },
       "rdp4_auth_enabled": true,
       "rdp4_enabled": true,
       "rdp5_enabled": true
    },
    "screen": {
       "maximum_bpp": 24,
       "maximum_height": 2000,
       "maximum_width": 2000
    },
    "timeout": 600
  }
```

Add RDP settings policies

To add a settings policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify RDP settings policies

To modify a settings policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

Code	Description	Notes
201	Created	The new resource was successfully created.
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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

10

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
  "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_hostkey": {
      "plain_hostkey": {
        "dsa_key": null,
        "enabled": true,
        "rsa_key": {
          "key": "e5a58682-6189-4477-9415-67c1c9b20b0d",
          "meta": {
            "href": "/api/configuration/private_keys/e5a58682-6189-4477-9415-
67c1c9b20b0d"
          }
        }
      },
      "x509_hostkey": {
        "enabled": false
      }
    },
    "indexing": {
      "enabled": true,
      "policy": {
        "key": "-50000",
        "meta": {
```

```
          "href": "/api/configuration/policies/indexing/-50000"
        }
      },
      "priority": 2
    },
    "log_audit_trail_downloads": true,
    "name": "API_test_SSH",
    "network": {
      "clients": [
        "0.0.0.0/24"
      ],
      "ports": [
        22
      ],
      "targets": [
        "192.168.56.102/24"
      ]
    },
    "policies": {
      "aa_plugin": null,
      "analytics_policy": null,
      "archive_cleanup_policy": {
        "key": "1854671967571b9063c4c82",
        "meta": {
          "href": "/api/configuration/policies/archive_cleanup_
policies/1854671967571b9063c4c82"
        }
      },
      "audit_policy": {
        "key": "78101850949e47437dd91d",
        "meta": {
          "href": "/api/configuration/policies/audit_
policies/78101850949e47437dd91d"
        }
      },
      "authentication_policy": {
        "key": "1895203635707e3340262f",
        "meta": {
          "href": "/api/configuration/ssh/authentication_
policies/1895203635707e3340262f"
        }
      },
      "backup_policy": {
        "key": "512524636571b903540804",
        "meta": {
          "href": "/api/configuration/policies/backup_
policies/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"
    },
    "server_side_hostkey": {
      "plain_hostkey": {
        "enabled": true,
        "hostkey_check": "accept-first-time"
      },
      "x509_hostkey": {
        "enabled": false
```

```
      }
    },
    "source_address": {
      "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_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_`

Element		Type	Description
			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 Low priority. • 3 Normal (default) priority. • 2 High priority. • 1 Very high priority. • 0 Near real-time priority.
log_audit_		boolean	Set to true to log audit trail downloads.

Element		Type	Description
trail_downloads		n	
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.
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	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).
	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.

Element		Type	Description
			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). 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.

Element		Type	Description
	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_hostkey		Top level element	Settings for verifying the server's identity using plain hostkeys 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

Element		Type	Description
			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_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.

Elements of access_control	Type	Description
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_hostkey	Type	Description
plain_hostkey	Top level item	Configures the RSA key SPS shows to the clients.
rsa_key	string	References the identifier of the RSA key. You can add RSA keys at the /api/configuration/private_keys/ endpoint. To modify or add an RSA key, use the value of the returned key as the value of the rsa_key element, and remove any child elements (including the key).
x509_hostkey	Top level item	Configures the X.509 keys SPS shows to the clients.
enabled	boolean	Set to true to allow presenting X.509 hostkeys to clients. You must enable either plain_hostkey or x509_hostkey (or both).

Elements of `client_side_hostkey`	Type	Description
`x509`	Top level item	Parameters for X.509 hostkeys.
`selection`	string	Possible values: • `fix` Presents the same certificate for every connection. Must be used with the `x509_identity` element. • `generate` Generates a X.509 certificate for the connection policy. Must be used with the `signing_CA` element.
`signing_ca`	string	Must be used when generating the X.509 certificate. References the signing Certificate Authority (CA). You can configure signing CAs at the `/api/configuration/policies/signing_cas/` endpoint. To modify or add a signing CA, use the value of the returned `key` as the value of the `rsa_key` element, and remove any child elements (including the key).
`x509_identity`	string	Must be used when using the same X.509 host certificate across connection policies. 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).

Elements of `server_side_hostkey`	Type	Description
`plain_hostkey`	Top level element	Verifies the identity of the servers based on their hostkeys.

Elements of `server_side_hostkey`			Type	Description
	`enabled`		boolean	Set to `true` to enable plain hostkey checking. If enabled, the `hostkey_check` element is mandatory.
	`hostkey_check`		string	Defines the method for checking the host keys of the target server. Possible values are: • `disabled` 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 hostkeys that are already stored on SPS. You can manage hostkeys at the `/api/ssh-host-keys` endpoint.
`x509_hostkey`			Top level element	Verifies the identity of the servers based on their X.509 certificates.
	`enabled`		string	Set to true to enable X.509 hostkey verification. If enabled, the `x509_check` element is mandatory.
	`x509_check`		Top level item	Contains the configuration settings for verifying X.509 certificates.
		`selection`	string	Configures the validation of X.509 certificates. Possible values are: • `disabled` Disables X.509 certificate verification. • `accept-first-time` Records the X.509 certificate shown for the first connection, and accepts only the same certificate for any subsequent connections. • `accept-known-certificates`

Elements of `server_side_hostkey`		Type	Description
			Only accepts X.509 certificates that are already stored on SPS. You can add X.509 certificates at the `/api/ssh-host-keys` endpoint. • `accept-signed-by` Accepts all X.509 certificates that are signed by a trusted Certificate Authority. Must be used with the `trusted_ca` element.
	`trusted_ca`	string	Must be used if the `selection` element is set to `accept-signed-by`. References the identifier of the trusted CA. You can add or modify the list of trusted CAs at the `/api/configuration/policies/trusted_ca_lists/` endpoint. To modify or add a trusted CA, use the value of the returned `key` as the value of the `trusted_ca` element, and remove any child elements (including the key).

Examples

For practical purposes, the following examples show only the relevant parts of a connection policy JSON object. To modify or add a connection policy, always submit the full JSON object.

Access control list: configuring the "security" usergroup to only audit connections made by the "root_only" usergroup.

```
"access_control": [
          {
       "authorizer": "security",
       "permission": "audit",
       "require_different_ip": true,
       "require_different_username": true,
       "subject": {
          "group": "root_only",
          "selection": "only"
       }
    }
```

Target server: use the address specified by the client.

```
"server_address": {
    "selection": "original"
}
```

Target server: use a fix address.

```
"server_address": {
    "address": "<fix-IP>",
    "port": 22,
    "selection": "fix"
}
```

Target server: configure inband destination selection, where the client can specify the target address in the username. The target can be either an IP range, or a domain.

```
"server_address": {
    "dns_server": "<ip-of-dns-server>",
    "dns_suffixes": null,
    "domains": [
       {
          "domain": {
             "selection": "address",
             "value": "<IP-range>"
          },
          "port": 22
       },
       {
          "domain": {
             "selection": "domain",
             "value": "*.example"
          },
          "port": 22
       }
    ],
    "selection": "inband"
}
```

Source address: use the same fix IP when connecting to the remote server.

```
"source_address": {
    "address": "<ip-address>",
    "selection": "fix"
}
```

Web gateway authentication: require the admin usergroup to perform an additional gateway authentication on the SPS web interface. They must authenticate from the same host which initiated the connection.

```
"web_gateway_authentication": {
    "enabled": true,
    "groups": [
       "admin"
    ],
    "require_same_ip": true
}
```

Client-side hostkey: use plain hostkeys uploaded to SPS, and generate X.509 certificates for the connection.

```
"client_side_hostkey": {
    "plain_hostkey": {
       "dsa_key": "<id-of-dsa-key>",
       "enabled": true,
       "rsa_key": "<id-of-rsa-key>"
    },
    "x509_hostkey": {
       "enabled": true,
       "x509": {
          "selection": "generate",
          "signing_ca": "<key-of-signing-ca>"
       }
    }
}
```

Policies: configure only the required policies.

```
"policies": {
    "aa_plugin": null,
    "analytics_policy": null,
    "archive_cleanup_policy": null,
    "audit_policy": "<key-of-audit-policy>",
    "authentication_policy": "<key-of-auth-policy>",
    "backup_policy": null,
    "channel_policy": "<key-of-channel-policy>",
    "credential_store": null,
    "ldap_server": null,
    "settings": "<key-of-settings-policy>",
    "usermapping_policy": null
}
```

Server-side hostkey: accept the hostkey or X.509 certificate presented at the first connection, and require the same hostkey or certificate for any subsequent connections.

```
"server_side_hostkey": {
    "plain_hostkey": {
       "enabled": true,
       "hostkey_check": "accept-first-time"
    },
    "x509_hostkey": {
       "enabled": true,
       "x509_check": {
          "selection": "accept-first-time"
       }
    }
}
```

Server-side hostkey: only accept X.509 certificates that are verified by a trusted CA.

```
"server_side_hostkey": {
    "plain_hostkey": {
       "enabled": false
    },
    "x509_hostkey": {
       "enabled": true,
       "x509_check": {
          "selection": "accept-signed-by",
          "trusted_ca": "<id-of-trusted-ca>"
       }
    }
}
```

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify an SSH connection policy

To modify an SSH connection policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

SSH channels

The available SSH channel types and their functionalities are described below.

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

Channel	Special options	Description

		"local_forwards": [
		  {
		  "host_address": {
		  "selection": "address",
		  "value": "192.168.100.1"
		  },
		  "host_port": 5555,
		  "originator_address": {
		  "selection": "address",
		  "value": "192.168.1.1"
		  }
		  }
		
		Channel-specific access control rules:
		• `local_forwards` (list): To permit local forwarding only to specific addresses, list the addresses in this field. Leave it empty to enable without restriction. In this case the client may forward any traffic to the remote host.
		Enter the source of the forwarded traffic into the `originator_address` field, the target of the traffic into the `host_address` field. Specify IP addresses or networks (in IP address/Prefix format). These parameters are the end-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": {

Channel	Special options	Description

		"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 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.

Channel	Special options	Description

		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:
		`"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

Channel	Special options	Description

		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:
		• `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 460.

Element		Type	Description
	`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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists SSH authentication policies.

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

The following command retrieves the properties of a specific policy.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/ssh/authentication_policies<object-id>
```

Response

The following is a sample response received when listing SSH authentication policies.

For details of the `meta` object, see Message format on page 9.


```
{
    "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": {
    "mode": {
      "gateway_authentication": {
        "selection": "none"
      },
```

```
      "gssapi": false,
      "relayed_methods": {
        "certificate": {
          "selection": "disabled"
        },
        "keyboard_interactive": true,
        "password": true,
        "public_key": {
          "selection": "disabled"
        }
      }
    },
    "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"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the policy.
body	Top level element	Contains the elements of the policy.
mode	Top level element	Contains the configuration of the policy.
gateway_authentication	Top level item	Client-side gateway authentication settings. The value of `selection` defines which authentication method is used.
relayed_methods	Top level element	Server-side authentication settings.
gssapi	boolean	Deprecated setting.
name	string	The name of the object. This name is also displayed on the SPS web interface. It cannot contain whitespace.

Elements of `gateway_authentication`	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 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 `certificate`, `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.
`certificate`	Top level item	Configures authentication with an X.509 certificate. The `enabled` child element is required for this option. To enable it, you must also configure the `trusted_ca` child element.
`enabled`	boolean	Possible values: • `true`

Elements of gateway_authentication		Type	Description
			Enables client-side, X.509 certification-based authentication. You must also use the trusted_ca element to define a certificate authority. • false Disables client-side, X.509 certificate-based authentication.
	trusted_ca	string	References the key of the trusted CA. You can configure trusted CAs at the /api/configuration/policies/trusted_ca_lists/ endpoint. To modify or add a trusted CA, use the value of the returned key as the value of the trusted_ca element, and remove any child elements (including the key).
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.
	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,

Elements of gateway_authentication			Type	Description
				port and shared_secret elements.
	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).
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 relayed_authentication	Type	Description
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.

Elements of `relayed_authentication`	Type	Description
		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.
`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

Elements of relayed_authentication	Type	Description
		server. Therefore, you must configure your clients to enable agent forwarding, otherwise authentication will fail. For details on enabling agent forwarding in your SSH application, see the documentation of the application.
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:

```
{
    "mode": {
       "gateway_authentication": {
          "certificate": {
             "enabled": false
          },
          "password": true,
          "public_key": {
             "enabled": false
          },
          "selection": "ldap"
       },
       "gssapi": false,
       "relayed_methods": {
          "certificate": {
             "selection": "disabled"
          },
          "keyboard_interactive": false,
          "password": true,
          "public_key": {
             "selection": "disabled"
          }
       }
    },
    "name": "Passwords"
}
```

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.

```
{
    "mode": {
       "gateway_authentication": {
          "certificate": {
             "enabled": false
          },
          "password": true,
          "public_key": {
             "enabled": false
          },
          "selection": "local",
          "user_database": <key-of-the-local-usr-db>
       },
       "gssapi": false,
       "relayed_methods": {
          "certificate": {
             "selection": "disabled"
          },
          "keyboard_interactive": false,
          "password": true,
          "public_key": {
             "selection": "disabled"
          }
       }
    },
    "name": "Local_usr_db"
 }
```

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.

```
{
  "mode": {
    "gateway_authentication": {
      "selection": "radius",
      "servers": [
        {
          "address": {
            "selection": "ip",
            "value": "<radius-server-ip>"
          },
          "port": 1812,
          "shared_secret": <key-of-shared-secret>
        }
```

```
      ]
    },
    "gssapi": false,
    "relayed_methods": {
      "certificate": {
        "selection": "disabled"
      },
      "keyboard_interactive": false,
      "password": true,
      "public_key": {
        "selection": "disabled"
      }
    }
  },
  "name": "RADIUS"
}
```

Using X.509 certificates against an LDAP server on the client-side, and forwarding it for authentication on the server-side. The key of the trusted Certificate Authority (CA) is available at the `/api/configuration/policies/trusted_ca_lists` endpoint.

```
{
    "mode": {
       "gateway_authentication": {
          "certificate": {
             "enabled": true,
             "trusted_ca": <key-of-trusted-ca>
          },
          "password": false,
          "public_key": {
             "enabled": false
          },
          "selection": "ldap"
       },
       "gssapi": false,
       "relayed_methods": {
          "certificate": {
             "selection": "agent"
          },
          "keyboard_interactive": false,
          "password": false,
          "public_key": {
             "selection": "disabled"
          }
       }
    },
    "name": "X509_forwarding"
}
```

Using X.509 certificates against an LDAP server on the client-side, and generating X.509 certificate and key on the fly for authentication on the server-side. The generated keys are uploaded to the LDAP server, so that SPS can authenticate the user on the remote server. The key of the trusted Certificate Authority (CA) is available at the `/api/configuration/policies/trusted_ca_lists/` endpoint. The key of the signing Certificate Authority (CA) is available at the `/api/configuration/policies/signing_cas/` endpoint.

```
{
    "mode": {
       "gateway_authentication": {
          "certificate": {
             "enabled": true,
             "trusted_ca": <key-of-trusted-ca>
          },
          "password": false,
          "public_key": {
             "enabled": false
          },
          "selection": "ldap"
       },
       "gssapi": false,
       "relayed_methods": {
          "certificate": {
             "selection": "publish_to_ldap",
             "signing_ca": <key-of-signing-ca>
          },
          "keyboard_interactive": false,
          "password": false,
          "public_key": {
             "selection": "disabled"
          }
       }
    },
    "name": "X509"
 }
```

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 36.

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 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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 Element . The elements of `gateway_authentication` are listed in Elements of gateway_authentication. The elements of `relayed_authentication` are listed in Elements of relayed_authentication.

   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 details, see Commit a transaction on page 30.

Modify an SSH authentication policy

To modify an SSH authentication policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 Element . The elements of `gateway_authentication` are listed in Elements of gateway_authentication. The elements of `relayed_authentication` are listed in Elements of relayed_authentication.

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists global SSH options.

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

Response

The following is a sample response received when listing global SSH options.

For details of the `meta` object, see Message format on page 9.


```
{
    "body": {
       "audit": {
          "cleanup": {
             "channel_database_cleanup_days": 600,
             "enabled": true
          },
          "timestamping": {
             "selection": "local",
             "signing_interval": 30
          }
       },
       "gssapi": {
          "enabled": false
       },
       "service": {
          "enabled": true,
          "log_level": 4
       }
    },
    "key": "options",
    "meta": {
       "first": "/api/configuration/ssh/authentication_policies",
       "href": "/api/configuration/ssh/options",
       "last": "/api/configuration/ssh/settings_policies",
       "next": "/api/configuration/ssh/settings_policies",
       "parent": "/api/configuration/ssh",
       "previous": "/api/configuration/ssh/connections",
       "transaction": "/api/transaction"
    }
 }
```


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.
audit	Top level	Contains settings for timestamping and cleanup.

Element	Type	Description
	item	
service	Top level item	Global setting to enable SSH connections, and specify the logging detail.
enabled	boolean	Set to true to enable SSH connections.
log_level	int	Defines the logging detail of SSH connections.
gssapi	Top level item	Deprecated setting.

Elements of audit		Type	Description
cleanup		Top level item	Global retention settings for SSH connection metadata. To configure retention time for a specific connection policy, use the archive_cleanup_policy element at the endpoint of the policy instead.
	channel_database_cleanup_days	int	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 this element to true.
timestamping		Top level item	Global timestamping settings for SSH connections.
	selection	string	Configures local or remote timestamping. • Set local to use SPS for timestamping. • Set remote to configure a remote timestamping server.
	server_url	string	Required for remote timestamping. The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.
	oid	Top level item	The Object Identifier of the policy used for timestamping.

Elements of `audit`		Type	Description
	`enabled`	boolean	Required for `remote` timestamping. Set to `true` to configure the Object Identifier of the timestamping policy on the timestamping remote server.
	`policy_oid`	string	Required if the `oid` is `enabled`. The Object Identifier of the timestamping policy on the remote timestamping server.
`signing_interval`		int	Time interval for timestamping open connections, in seconds.

Examples:

Set SPS as the timestamping server:

```
{
    "audit": {
       "cleanup": {
          "enabled": false
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "gssapi": {
       "enabled": false
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Enable cleanup, and set it to occur every 10 days:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
      "selection": "local",
      "signing_interval": 30
    }
```

```
  },
  "gssapi": {
    "enabled": false
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, without specifying a timestamping policy:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
             "oid": {
                "enabled": false
             },
             "selection": "remote",
             "server_url": "<url-of-timestamping-server>",
             "signing_interval": 30
          }
    },
    "gssapi": {
       "enabled": false
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Change timestamping to a remote server, and specify the `1.2.3` timestamping policy:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
        "oid": {
          "enabled": true,
          "policy_oid": "1.2.3"
        },
```

```
        "selection": "remote",
        "server_url": "<url-of-timestamping-server>",
        "signing_interval": 30
      }
  },
  "gssapi": {
    "enabled": false
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Status and error codes

The following table lists the typical status and error codes for this request. For a complete list of error codes, see Application level error codes on page 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

2. **Modify the JSON object of the global SSH settings endpoint.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/ssh/options` endpoint. You can find a detailed description of

the available parameters listed in Element . The elements of the `audit` item are described in Elements of audit.

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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": {
    "client_side_algorithms": {
      "cipher": [
        "aes128-ctr",
        "aes192-ctr",
        "aes256-ctr"
      ],
      "compression": [
        "none"
      ],
```

```
      "kex": [
        "diffie-hellman-group14-sha1"
      ],
      "mac": [
        "hmac-sha2-256",
        "hmac-sha2-512"
      ]
    },
    "greeting": "Welcome!",
    "name": "API_SSH_Setting",
    "preconnect_channel_check": true,
    "server_side_algorithms": {
      "cipher": [
        "aes128-ctr",
        "aes192-ctr",
        "aes256-ctr"
      ],
      "compression": [
        "none"
      ],
      "kex": [
        "diffie-hellman-group14-sha1"
      ],
      "mac": [
        "hmac-sha2-256",
        "hmac-sha2-512"
      ]
    },
    "software_version": "SSH",
    "strict_mode": true,
    "timeout": 600,
    "userauth_banner": "This is a monitored connection."
  },
  "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

Element			Type	Description
				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.
		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.

Element	Type	Description
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 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. Note that the SPS web UI displays the same value in milli-seconds.
userauth_banner	string	You can display a banner message to the clients before authentication (as specified in RFC 4252 â The Secure Shell (SSH) Authentic-ation 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 36.

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 author-ization 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.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify SSH settings policies

To modify a settings policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the SSH host keys and certificates of the servers that the users can connect to using SSH.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/ssh-host-keys/
```

The following command retrieves the properties of a specific key.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/ssh-host-keys/<object-id>
```

Response

The following is a sample response received when listing SSH host keys and certificates from the `https:<IP-address-of-SPS>/api/ssh-host-keys/` endpoint.

For details of the `meta` object, see Message format on page 9.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 44. 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 42. 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.

11

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "items": [
       {
          "key": "channel_policies",
          "meta": {
             "href": "/api/configuration/telnet/channel_policies"
          }
       },
       {
          "key": "options",
          "meta": {
             "href": "/api/configuration/telnet/options"
          }
       }
    ],
    "meta": {
       "first": "/api/configuration/aaa",
       "href": "/api/configuration/telnet",
       "last": "/api/configuration/x509",
       "next": "/api/configuration/troubleshooting",
       "parent": "/api/configuration",
       "previous": "/api/configuration/ssh",
       "transaction": "/api/transaction"
    }
 }
```

Item	Description
`channel_policies`	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 36.

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

Code	Description	Notes
		retrieved.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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.

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/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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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>
```

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 global Telnet options.

For details of the `meta` object, see Message format on page 9.

```
{
    "body": {
       "audit": {
          "cleanup": {
             "enabled": false
          },
          "timestamping": {
             "selection": "local",
             "signing_interval": 30
          }
       },
       "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"
    }
 }
```

Element	Type	Description
key	Top level item	Contains the ID of the endpoint.
body	Top level item	Contains the elements of the global Telnet options.
audit	Top level item	Contains settings for timestamping and cleanup.
service	Top level item	Global setting to enable Telnet connections, and specify the logging detail.
enabled	boolean	Set to `true` to enable Telnet connections.
log_level	int	Defines the logging detail of Telnet connections.

Elements of `audit`			Type	Description
`cleanup`			Top level item	Global retention settings for Telnet connection metadata. To configure retention time for a specific connection policy, use the `archive_cleanup_policy` element at the endpoint of the policy instead.
	`channel_database_cleanup_days`		int	Global retention time for the metadata of Telnet connections, in days. Must exceed the retention time of the archiving policy (or policies) used for Telnet connections, and the connection-specific database cleanup times (if configured).
	`enabled`		boolean	To enable the global cleanup of Telnet connection metadata, set this element to `true`.
`timestamping`			Top level item	Global timestamping settings for Telnet connections.
	`selection`		string	Configures local or remote timestamping. • Set `local` to use SPS for timestamping. • Set `remote` to configure a remote timestamping server.
	`server_url`		string	Required for `remote` timestamping. The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.
	`oid`		Top level item	The Object Identifier of the policy used for timestamping.
		`enabled`	boolean	Required for `remote` timestamping. Set to `true` to configure the Object Identifier of the timestamping policy on the timestamping remote server.
		`policy_oid`	string	Required if the `oid` is `enabled`. The Object Identifier of the timestamping policy on the remote timestamping server.
	`signing_interval`		int	Time interval for timestamping open connections, in seconds.

Examples:

Set SPS as the timestamping server:

```
{
    "audit": {
       "cleanup": {
          "enabled": false
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Enable cleanup, and set it to occur every 10 days:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Change timestamping to a remote server, without specifying a timestamping policy:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
        "oid": {
```

```
          "enabled": false
        },
        "selection": "remote",
        "server_url": "<url-of-timestamping-server>",
        "signing_interval": 30
      }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, and specify the `1.2.3` timestamping policy:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
             "oid": {
                "enabled": true,
                "policy_oid": "1.2.3"
             },
             "selection": "remote",
             "server_url": "<url-of-timestamping-server>",
             "signing_interval": 30
          }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Modify global Telnet settings

To modify global Telnet settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

2. **Modify the JSON object of the global Telnet settings endpoint.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/telnet/options` endpoint. You can find a detailed description of the available parameters listed in Element . The elements of the `audit` item are described in Elements of audit.

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

12

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available settings for configuring for VNC connections.

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

Response

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

For details of the `meta` object, see Message format on page 9.

```
{
    "items": [
       {
          "key": "channel_policies",
          "meta": {
             "href": "/api/configuration/vnc/channel_policies"
          }
       },
       {
          "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.
`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 36.

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

Code	Description	Notes
		retrieved.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "body": {
       "audit": {
          "cleanup": {
             "enabled": false
          },
          "timestamping": {
             "selection": "local",
             "signing_interval": 30
          }
       },
       "service": {
          "enabled": true,
          "log_level": 4
       }
    },
    "key": "options",
    "meta": {
       "first": "/api/configuration/vnc/channel_policies",
       "href": "/api/configuration/vnc/options",
       "last": "/api/configuration/vnc/options",
       "next": null,
       "parent": "/api/configuration/vnc",
       "previous": "/api/configuration/vnc/channel_policies",
       "transaction": "/api/transaction"
    }
 }
```

Element	Type	Description
key	Top level item	Contains the ID of the endpoint.
body	Top level item	Contains the elements of the global VNC options.
audit	Top level item	Contains settings for timestamping and cleanup.
service	Top level item	Global setting to enable VNC connections, and specify the logging detail.
enabled	boolean	Set to `true` to enable VNC connections.
log_level	int	Defines the logging detail of VNC connections.

Elements of `audit`			Type	Description
`cleanup`			Top level item	Global retention settings for VNC connection metadata. To configure retention time for a specific connection policy, use the `archive_cleanup_policy` element at the endpoint of the policy instead.
	`channel_database_cleanup_days`		int	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`.
`timestamping`			Top level item	Global timestamping settings for VNC connections.
	`selection`		string	Configures local or remote timestamping. • Set `local` to use SPS for timestamping. • Set `remote` to configure a remote timestamping server.
	`server_url`		string	Required for `remote` timestamping. The URL of the timestamping server. Note that HTTPS and password-protected connections are not supported.
	`oid`		Top level item	The Object Identifier of the policy used for timestamping.
		`enabled`	boolean	Required for `remote` timestamping. Set to `true` to configure the Object Identifier of the timestamping policy on the timestamping remote server.
		`policy_oid`	string	Required if the `oid` is `enabled`. The Object Identifier of the timestamping policy on the remote timestamping server.
	`signing_interval`		int	Time interval for timestamping open connections, in seconds.

Examples:

Set SPS as the timestamping server:

```
{
    "audit": {
       "cleanup": {
          "enabled": false
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Enable cleanup, and set it to occur every 10 days:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
          "selection": "local",
          "signing_interval": 30
       }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Change timestamping to a remote server, without specifying a timestamping policy:

```
{
  "audit": {
    "cleanup": {
      "channel_database_cleanup_days": 10,
      "enabled": true
    },
    "timestamping": {
        "oid": {
          "enabled": false
        },
        "selection": "remote",
        "server_url": "<url-of-timestamping-server>",
```

```
        "signing_interval": 30
      }
  },
  "service": {
    "enabled": true,
    "log_level": 4
  }
}
```

Change timestamping to a remote server, and specify the `1.2.3` timestamping policy:

```
{
    "audit": {
       "cleanup": {
          "channel_database_cleanup_days": 10,
          "enabled": true
       },
       "timestamping": {
             "oid": {
                "enabled": true,
                "policy_oid": "1.2.3"
             },
             "selection": "remote",
             "server_url": "<url-of-timestamping-server>",
             "signing_interval": 30
          }
    },
    "service": {
       "enabled": true,
       "log_level": 4
    }
 }
```

Modify global VNC settings

To modify global VNC settings, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

2. **Modify the JSON object of the global VNC settings endpoint.**

   PUT the modified JSON object to the `https://<IP-address-of-SPS>/api/configuration/vnc/options` endpoint. You can find a detailed description of the available parameters listed in Element . The elements of the `audit` item are described in Elements of audit.

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

13

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the connections.

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

The following command retrieves the properties of a specific connection.

```
curl --cookie cookies https://<IP-address-of-SPS>/api/audit/sessions/<session-id>
```

Response

The following is a sample response received when listing connections.

For details of the `meta` object, see Message format on page 9.


```
{
    "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 item	Contains a link to the details of the alerts. For details, see Session alerts on page 565. An event is listed as alert only if the **Actions > Store**

Element	Type	Description

		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 569.
		`"events": {`
		`"href": "/api/audit/sessions/7930f4308efe8aecd710202d815b76ff/events"`
		`},`

hidden	boolea-	True if this is a session that has not been displayed on

Element		Type	Description
		n	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 579.
			`"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 532.
	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 the /api/configuration/<protocol>/channel_policies/ endpoint.
	command_	boolea-	If commands have been extracted from this terminal

Element		Type	Description
	extracted	n	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. • 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

Element		Type	Description
			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` The connection attempt failed. • `gw-auth-fail` Gateway authentication failure. • `key-error` The connection attempt failed due to a hostkey

Element		Type	Description
			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.
	`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.
	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": {`

Audit trail elements	Type	Description
		`"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/sessions/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.
`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

Channel elements	Type	Description
		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` 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.

Channel elements	Type	Description
		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`, 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.
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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 532. 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 "Replay audit files in follow mode" in the Safeguard Desktop Player User Guide.

Searching in the session database

You can list, search, and filter the SPS session database at the `/api/audit/sessions` endpoint. You can use the following actions:

- `?start`

  Display sessions that started after the specified date. Use the ISO 8601 format for the date, for example, `2017-01-25T10:00`.

- `?q`

  Filter the list using one or more property (element) of the sessions.

- `?content`

  Search in the content of indexed sessions.

- `?end`

  Display sessions that ended before the specified date. Use the ISO 8601 format for the date, for example, `2017-01-25T10:00`.

- `?fields`

  Display the selected properties (elements and values) of the listed sessions.

- `?limit`

  Configure the pagination of the displayed results using the `?offset` and `?limit` parameters.

  The `?limit` parameter allows you to configure the maximum number of results to display on a page at once.

  The default value of `?limit` is `500`.

  > NOTE:
  >
  > The default value of `500` is the maximum permitted value you can set for `?limit`. If you set the `?limit` parameter to a value bigger than `500`, only the first 500 results will be displayed.

- `?offset`

  Configure the pagination of the displayed results using the `?offset` and `?limit` parameters.

  The `?offset` parameter allows you to configure the offset from the first result that is displayed. This can be useful if the number of items returned exceeds the number of items displayed on the first page, and you want to navigate to any of the subsequent items displayed on other pages.

  The default value of `?offset` is `null`.

  > NOTE:
  >
  > The maximum number of search results in One Identity Safeguard for Privileged Sessions is `10000`. As a result, any `?offset` values set to larger than `10000` will be ignored and the results exceeding the value of `10000` will not be displayed.

- `?sort`

  Sort the results based on the values of the fields.

- `?format`

Configure the format of the displayed results.

The default value of `?format` is `json`. If you do not configure the `?format` parameter, the results will be displayed in JSON format.

To display search results in a CSV format, enter `csv` as a value.

To combine multiple expressions, use the & (ampersand) character, for example:

Display the target server and port of each active session:

```
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.
`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" },` For details about the returned fields, see Element .
`key`	string	A globally unique string that identifies the session. This session ID has the following format: *svc/<unique-random-hash>/<name-of-*

Element	Type	Description
		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 object	Top level element, a list containing meta information 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" ],` For details about the returned fields, see Element .
`limit`	integer	The maximum number of sessions returned in a the response (by default, 500).

Element	Type	Description
`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 hostkey mismatch.
  - `user-mapping-fail`

    The connection attempt failed due to a user mapping failure.

Content search in indexed audit trails

You can use the `?content` option to search for keywords that appear in the content of the audit trails. Such content is any text that appeared on the screen in terminal or graphical sessions, or commands that the user typed in terminal sessions. Note that content search works only if:

- Indexing was enabled in the connection policy related to the audit trail during the session, and
- the audit trail has already been indexed.

```
?content="my-search-expression"
```

You can use the Apache Lucene query syntax 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`

```
?content=example
```

Search for the words `example`, `examples`, and so on:

```
?content=example%3F
```

Search for the words `example`, `examine`, and so on:

```
?content=exam%2A
```

Search in metadata and session content at the same time:

```
curl --cookie cookies "https://<IP-address-of-
SPS>/api/audit/sessions?q=protocol:ssh&content=sudo"
```

For further details and examples, see "Using the content search" 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 audit trail file where the content element first appeared.

Element		Type	Description
	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.
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 9.

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_	Contains the	Required	The value of the session ID cookie received from

Cookie name	Description	Required	Values
id	authentication token of the user		the REST server in the authentication response, for example, `a1f71d030e657634730b9e887cb59a5e56162860`. For details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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`.
- `?content`: Statistics are returned for indexed sessions that contain the type of content specified.

> NOTE:
>
> When performing a content query, the maximum number of results returned is 3000. When this limit is exceeded, the scope of statistics is limited to the first 3000 sessions (even if there are more than 3000 sessions that match your criteria).

Response

The following snippet is a sample response received when retrieving statistics about the `protocol` field.

For details of the `meta` object, see Message format on page 9.

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 list is returned		Top-level element (string)	Contains the properties that are in the scope of the requested statistics.
	`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 36.

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 details on authentication, see Authenticate to the SPS REST API on page 18.

Cookie name	Description	Required	Values
			Note that this session ID refers to the connection between the REST client and the SPS 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 parameters to fine-tune your request for statistics:

- `?q`: Narrow down the scope of the histogram using one or more properties (elements) of the sessions.
- `?field`: Create a histogram for the selected properties (elements and values) of sessions (for example, protocol).

  Using this parameter is mandatory.
- `?bin-size`: Determines the size of the unit for the histogram, for example, `hour`. SPS splits the queried period to intervals of this unit, and returns the number of sessions to each interval. For example, if you query an histogram from 2018-02-12:14:40 to 2018-02-16:14:40, and you set the `bin-size` to `day`, then SPS will return five datasets (one for each day). If you set the `bin-size` to `week`, then SPS will return only one dataset.
- `?start`: Create a histogram from the sessions that started after the specified date. Use the ISO 8601 format for the date, for example, `2017-01-25T10:00`. By default, this is the one month before the date of the request.
- `?end`: Create a histogram from the sessions that ended before the specified date. Use the ISO 8601 format for the date, for example, `2017-01-25T11:00`. By default, this is the date of the request.
- `?size`: Limit the range of values displayed in the histogram for a given field. The histogram will only be created for the top `size` number of most frequently occurring values (that is, values with the highest number of counts).

Response

The following snippet is a sample response received when retrieving a histogram about the audited sessions.

For details of the `meta` object, see Message format on page 9.

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.

Element	Type	Description
`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.
`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 36.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the alerts of a session.

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-
id>/alerts"
```

Response

The following is a sample response received when listing the alerts of a session.

For details of the `meta` object, see Message format on page 9.

```
{
    "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.

Element	Type	Description
matched_action	integer	A reference to the ID of the channel in the session where the event occurred.
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 579.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the events of a session.

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-id>/events"
```

Response

The following is a sample response received when listing the events of a session.

For details of the `meta` object, see Message format on page 9.


```
{
    "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 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=content:sudo
```

You can escape special characters using the backslash character.

```
?q=content:\"Copying Files\"
```

To add multiple elements to the filter, you can use the `AND`, `AND NOT`, and `OR` operators.

```
content:ls AND content:cp AND NOT content:mv
```

You can create groups using `()` (parentheses).

```
?q=(content:rm OR content:mv) AND channels_id:5
```

You can also use `()` (parentheses) to add multiple possible values for a property.

```
?q=content:(sudo rm)
```

You can use the * (asterisk) and ? (question mark) wildcards for string-type values.

```
?q=content:?dmi*
```

You can define ranges using [] (brackets) or {} (braces) and the TO operator. This only works for numeric (int) values.

- [means equal or higher than the following value
-] means equal or lower than the preceding value
- { means higher than the following value
- }means lower than the preceding value

For example, the following range resolves to 2:

```
?q=channels_id:{1 TO 3}
```

You can also use the * (asterisk) wildcard in the range.

```
?q=channels_id:[* TO 5]
```

Note that not all connection data can be used for filtering. The available elements are:

- `channels_id`

  Integer, the channel in the session where the event occurred.
- `content`

  Text, the event that occurred in the session.
- `record_id`

  Integer, the identifier of the event in the session.
- `time`

  String, the timestamp when the event occurred.
- `type`

  String, the type of the event:
  - `command`: A command entered in SSH or Telnet.
  - `screen.content`: Screen content.
  - `screen.creditcard`: Credit card numbers detected. Displayed only as an alert, not visible in the events.
  - `screen.windowtitle`: The title of the window in graphic protocols.

Indexing sessions

The `api/audit/sessions/<session-id>/indexing` endpoint lists the indexing-related details in this session (if any). For details on configuring indexing, see Local services: configuring the indexer on page 579.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the indexing-related details of a session.

```
curl --cookie cookies "https://<IP-address-of-SPS>/api/audit/sessions/<session-
id>/indexing"
```

Response

The following is a sample response received when listing the indexing-related details of a session.

For details of the `meta` object, see Message format on page 9.

```
{
    "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 filters" 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 579.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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/addresses/
1"
                        }
                    },
                    "port": 12345
                }
            ],
            "ssl_config": {
                "ca": {
                    "key": "52735ce4-4a43-458d-8803-c23c715640a5",
                    "meta": {
                        "href": "/api/configuration/x509/52735ce4-4a43-458d-8803-
c23c715640a5"
                    }
```

```
                },
                "service": {
                    "key": "60eacdba-d889-4cb4-bdb0-cbbd4054f01c",
                    "meta": {
                        "href": "/api/configuration/x509/60eacdba-d889-4cb4-bdb0-
cbbd4054f01c"
                    }
                },
                "worker": {
                    "key": "93198544-1e82-4661-90b7-e01b0b1e2ed9",
                    "meta": {
                        "href": "/api/configuration/x509/93198544-1e82-4661-90b7-
e01b0b1e2ed9"
                    }
                }
            }
        },
        "selection": "integrated"
    },
    "key": "indexer",
    "meta": {
        "first": "/api/configuration/local_services/admin_web",
        "href": "/api/configuration/local_services/indexer",
        "last": "/api/configuration/local_services/user_web",
        "next": "/api/configuration/local_services/postgresql",
        "parent": "/api/configuration/local_services",
        "previous": "/api/configuration/local_services/admin_web",
        "remaining_seconds": 599,
        "transaction": "/api/transaction"
    }
}
```


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 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 210.

Element	Type	Description
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 210.
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`.
enabled	boolean	Set it to `true` to restrict access to the specified client addresses.
enabled	boolean	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.

Element		Type	Description
			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.
	`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,
```

Element		Type	Description

			"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.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"
			}
			},

	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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It

Cookie name	Description	Required	Values
			is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available indexer policies.

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

The following command displays a specific indexer policy.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/policies/indexing/<id-of-the-policy>
```

Response

The following is a sample response received when querying the `/api/configuration/policies/indexing/` endpoint.

For details of the `meta` object, see Message format on page 9.


```
{
    "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	Contains the configuration options of the indexer

Element		Type	Description
		level element (string)	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.
	`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 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`,

Custom languages elements	Type	Description
		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`, *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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.

Code	Description	Notes
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 28.

Add an indexing policy

To add an indexing policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

14

Reporting

Reporting

List of endpoints for configuring reporting, and accessing the generated reports.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
    "meta": {
       "first": "/api/configuration/aaa",
       "href": "/api/configuration/reporting",
       "last": "/api/configuration/x509",
       "next": "/api/configuration/ssh",
       "parent": "/api/configuration",
       "previous": "/api/configuration/rdp",
       "transaction": "/api/transaction"
    },
    "items": [
       {
          "key": "content_subchapters",
          "meta": {
             "href": "/api/configuration/reporting/content_subchapters"
          }
       },
       {
          "key": "custom_subchapters",
          "meta": {
             "href": "/api/configuration/reporting/custom_subchapters"
          }
       },
       {
          "key": "predefined_reports",
          "meta": {
             "href": "/api/configuration/reporting/predefined_reports"
          }
       },
       {
          "key": "reports",
          "meta": {
             "href": "/api/configuration/reporting/reports"
          }
       },
       {
          "key": "statistics_subchapters",
          "meta": {
             "href": "/api/configuration/reporting/statistics_subchapters"
          }
       }
    ]
}
```

Endpoint	Description
content_subchapters	List of the reporting subchapters created from audit trail content (statistics of search keywords, and screenshots).
custom_subchapters	List of the reporting subchapters created from custom queries to the SPS connection database.
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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
  "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.
		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.

Element		Type	Description
	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 605. 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.

Chapters elements	Type	Description
	`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 36.

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 authen-

Code	Description	Notes
		ticated 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.
404	NotFound	The requested object does not exist.

Add a report

To add a report, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a report

To modify a report, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "meta": {
       "first": "/api/configuration/reporting/content_subchapters",
       "href": "/api/configuration/reporting/predefined_reports",
       "last": "/api/configuration/reporting/statistics_subchapters",
       "next": "/api/configuration/reporting/reports",
       "parent": "/api/configuration/reporting",
       "previous": "/api/configuration/reporting/custom_subchapters",
       "transaction": "/api/transaction"
    },
    "items": [
       {
          "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.
	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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.


```
{
    "meta": {
       "first": "/api/configuration/reporting/content_subchapters",
       "href": "/api/configuration/reporting/content_subchapters",
       "last": "/api/configuration/reporting/statistics_subchapters",
       "next": "/api/configuration/reporting/custom_subchapters",
       "parent": "/api/configuration/reporting",
       "previous": null,
       "transaction": "/api/transaction"
    },
    "items": [
       {
          "key": "13869311625707e0a3e0892",
          "meta": {
             "href": "/api/configuration/reporting/content_
subchapters/13869311625707e0a3e0892"
          }
       }
    ]
 }
```


When retrieving the endpoint of a specific content subchapter, the response is the following.


```
{
  "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_	string	References the key of the channel policy. You can

Element		Type	Description
	policy		configure channel policies at the "/api/configuration/<protocol>/channel_policies/<policy-ID>" endpoint. Note that the path is different for each protocol. 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 36.

Code	Description	Notes
201	Created	The new resource was successfully created.
400	InvalidQuery	The requested filter or its value is invalid.
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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a content subchapter

To modify a content subchapter, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Custom subchapters

List of the reporting subchapters created from custom queries to the SPS connection database. The list of tables and fields you can query are described in "Database tables available for custom queries" in the Administration Guide.

URL

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

Cookies

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the available custom subchapters.

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

The following command retrieves the properties of a specific subchapter.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/reporting/custom_subchapters/<object-id>
```

Response

The following is a sample response received when listing custom subchapters.

For details of the `meta` object, see Message format on page 9.

```
{
    "meta": {
       "first": "/api/configuration/reporting/content_subchapters",
       "href": "/api/configuration/reporting/custom_subchapters",
       "last": "/api/configuration/reporting/statistics_subchapters",
       "next": "/api/configuration/reporting/predefined_reports",
       "parent": "/api/configuration/reporting",
       "previous": "/api/configuration/reporting/content_subchapters",
       "transaction": "/api/transaction"
    },
    "items": [
       {
          "key": "5983143445707eb740fee3",
          "meta": {
             "href": "/api/configuration/reporting/custom_
subchapters/5983143445707eb740fee3"
          }
       }
    ]
 }
```

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

```
{
  "body": {
    "access": [
      "search"
    ],
    "chart": {
      "column_titles": [
        "col1",
        "col2"
      ],
      "type": "list"
    },
    "name": "API_test_adv_stats",
    "query": "select\n  to_timestamp(audit_trail_downloads.download_time),\n  audit_
trail_downloads.username,\n  channels.channel_type,\n  channels.connection,\n
audit_trail_downloads.ip\nfrom audit_trail_downloads,\n     channels\nwhere
channels._connection_channel_id = audit_trail_downloads.id\nand audit_trail_
```

```
downloads.download_time <= :range_start\nand audit_trail_downloads.download_time >
:range_end\nand audit_trail_downloads.username != 'admin'\norder by audit_trail_
downloads.download_time;"
  },
  "key": "5983143445707eb740fee3",
  "meta": {
    "first": "/api/configuration/reporting/custom_
subchapters/5983143445707eb740fee3",
    "href": "/api/configuration/reporting/custom_
subchapters/5983143445707eb740fee3",
    "last": "/api/configuration/reporting/custom_
subchapters/5983143445707eb740fee3",
    "next": null,
    "parent": "/api/configuration/reporting/custom_subchapters",
    "previous": null,
    "transaction": "/api/transaction"
  }
}
```


Element	Type	Description
key	string	Top level element, contains the ID of the custom subchapter.
body	Top level element (string)	The elements of the custom 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. You have to provide the `y_axis_title` element for bar charts (its can be `null`). • Use `pie` to generate pie a chart. • Use `list` to generate a list. You have to provide the `column_titles` element for lists (it can be `null`).
y_axis_title	string	Required if the `type` element is set to `bar`. The name of the y axis for the generated bar

Element	Type	Description
		chart.
`column_titles`	list	Required if the `type` element is set to `list`. The column titles for the generated list.
`name`	string	The name of the subchapter.
`query`	string	The SQL database query for creating the subchapter. **CAUTION:** **Generating a report that includes an Advanced statistics chapter that returns several thousands of entries requires significant CPU and memory resources from One Identity Safeguard for Privileged Sessions (SPS). While generating such a partial report, the web interface of SPS can become slow or unresponsive.**

Examples:

Create a bar chart with a custom title for the y-axis:

```
"chart": {
    "type": "bar",
    "y_axis_title": "Y_axis"
}
```

Create a pie chart:

```
"chart": {
    "type": "pie"
}
```

Create a list with custom column names:

```
"chart": {
    "column_titles": [
        "col1",
        "col2"
    ],
    "type": "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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

Add a custom subchapter

To add a custom subchapter, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

2. **Create the JSON object for the new subchapter.**

   POST the JSON object to the `https://<IP-address-of-SPS>/api/configuration/reporting/custom_subchapters` endpoint. You can find a detailed description of the available parameters listed in Element .

   If the POST request is successful, the response includes the key of the new subchapter. For example:

```
{
    "key":  "9a8f7f19-edbf-4327-9d3a-9f527e7331ee",
    "meta":  {
       "href":  "/api/configuration/reporting/custom_subchapters/9a8f7f19-edbf-
4327-9d3a-9f527e7331ee",
       "parent":  "/api/configuration/reporting/custom_subchapters",
       "transaction":  "/api/transaction"
    }
 }
```

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

Modify a custom subchapter

To modify a subchapter, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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

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

3. **Commit your changes.**

   For details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It

Cookie name	Description	Required	Values
			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 details of the `meta` object, see Message format on page 9.


```
{
    "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. • 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 532.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a connection statistics subchapter

To modify a subchapter, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Health and maintenance

Monitor appliance health status

To monitor the health status of an appliance, complete the following steps.

1. **Query the `/api/health-status` endpoint.**

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

The following is a sample response received.

For details of the `meta` object, see Message format on page 9.

For details of the other objects, see tables Cluster status details and "issues" object details.

```
{
      "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
      }
}
```

Elements	Type	Description
`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	Hard disk use

Elements	Type	Description
	(percent)	
swap	floating point integer (percent)	Swap use
load1	floating point integer	The average system load during the last one minute. The values mean the following: • < 1: low system load • 1-5: high system load • > 5: extremely high system load
load5	floating point integer	The average system load during the last five-minute period. The values mean the following: • < 1: low system load • 1-5: high system load • > 5: extremely high system load
load15	floating point integer	The average system load during the last fifteen-minute period. The values mean the following: • < 1: low system load • 1-5: high system load • > 5: extremely high system load

sessions	string	The protocol type and the number of ongoing sessions. For example:
		`"sessions": {`
		`  "ssh": 3,`
		`  "rdp": 4`
		`},`

total_sessions	integer (number of)	The total number of ongoing sessions.

15

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the existing usermapping policies.

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

The following command retrieves the properties of a specific usermapping policy.

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

Response

The following is a sample response received when listing usermapping policies.

For details of the `meta` object, see Message format on page 9.


```
{
    "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 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_`	list	The usergroups allowed to log in as the `remote_`

Element		Type	Description
	groups		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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has

Code	Description	Notes
		failed.
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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify a usermapping policy

To modify a usermapping policy, you have to:

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists endpoints for configuring plugins.

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

Response

The following is a sample response received when listing endpoints for configuring plugins.

For details of the `meta` object, see Message format on page 9.

```
{
    "items": [
       {
          "key": "aa",
          "meta": {
             "href": "/api/configuration/plugins/aa"
          }
       },
       {
          "key": "credentialstore",
          "meta": {
             "href": "/api/configuration/plugins/credentialstore"
          }
       }
    ],
    "meta": {
       "first": "/api/configuration/aaa",
       "href": "/api/configuration/plugins",
       "last": "/api/configuration/x509",
       "next": "/api/configuration/policies",
       "parent": "/api/configuration",
       "previous": "/api/configuration/passwords",
       "transaction": "/api/transaction"
    }
 }
```

Element	Description
aa	Endpoint for configuring authentication and authorization plugins.
credentialstore	Endpoint for configuring credential store plugins.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

Upload a plugin

To upload or update a plugin, complete the following steps. Note that currently you cannot delete a plugin, only update it by uploading a new version.

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Authentication and authorization plugins

The authentication and authorization (AAA) 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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS 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 AAA plugins used on SPS.

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

The following command retrieves the properties of a specific plugin.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/configuration/plugins/aa/<plugin-id>
```

Response

The following is a sample response received when querying the list of AAA plugins used on SPS.

For details of the `meta` object, see Message format on page 9.

```
{
  "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",
       "path": "/opt/scb/var/plugins/aa/AAPluginExample",
       "version": "1.0"
    },
    "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.
`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.
`name`	string	The name of the plugin. This name is also displayed on the SPS web interface. It cannot contain whitespace.

Element	Type	Description
`path`	string	The path where the plugin is stored on SPS.
`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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

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 details on authentication, see Authenticate to the SPS REST API on page 18.

Cookie name	Description	Required	Values
			Note that this session ID refers to the connection between the REST client and the SPS 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 details of the `meta` object, see Message format on page 9.

```
{
    "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": "0.0"
    },
    "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.
`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.
`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 36.

Code	Description	Notes
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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the credential stores.

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

The following command retrieves the properties of a specific credential store.

```
curl --cookie cookies https://<IP-address-of-
SPS>/api/policies/credentialstores/<policy-id>
```

Response

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

For details of the `meta` object, see Message format on page 9.


```
{
    "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 item	The IP addresses of the DNS servers to use 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	Configures the account SPS uses to login to the ERPM server.

Element		Type	Description
		level item	
	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.

Elements of `domain_mappings`	Type	Description
`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 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.

Elements of `login_mode`	Type	Description
		• `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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
404	NotFound	The requested object does not exist.

16

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. Replace the line-breaks in the license file with `\n` characters, for example:
		`"license": "Product: Shell Control Box\nEdition: Single\n[...]",`
		Note that you can complete the Welcome Wizard without uploading a license. In this case, SPS will start in demo mode. To skip uploading the license, use the `null` value:
		`"license": null,`
		To upload a license file, see Upload a new license.

`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 203.

Element	Type	Description
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: `"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

Element	Type	Description
		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.**
`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. One Identity recommends using 2048-bit RSA keys (or stronger). For example: `"certificates": { "ca": { "certificate": "-----BEGIN CERTIFICATE----- \nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMAkGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n" }, "webserver": { "certificate": "-----BEGIN CERTIFICATE----- \nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMAkGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n", "private_key": "-----BEGIN RSA PRIVATE KEY----- \nMIIEogIBAAKCAQEA/JERC+o1UksvUfbzS5Yp77CNlS6RkkdZLPjl2i9+ACzv/lOy\n...\n-----END RSA PRIVATE KEY-----\n" }, "tsa": { "certificate": "-----BEGIN CERTIFICATE----- \nMIIEWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQ0FADCBzDELMAkGA1UEBhMCUk8x\n...\n-----END CERTIFICATE-----\n", "private_key": "-----BEGIN RSA PRIVATE KEY----- \nMIIEogIBAAKCAQEA/JERC+o1UksvUfbzS5Yp77CNlS6RkkdZLPjl2i9+ACzv/lOy\n...\n-----END RSA PRIVATE KEY-----\n"` (machine-data) (machine-data) (machine-data)

Element	Type	Description
		`}` `},`
`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 36.

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

Code	Description	Notes
		"parent": "/api", "remaining_seconds": 0 } }

401	ConfigurationAlreadyInitialized	The Welcome Wizard was already successfully completed on this SPS.
		{
		"error": {
		"details": {
		"path": "/api/setup"
		},
		"message": "The configuration of the system is already initialized.",
		"type": "ConfigurationAlreadyInitialized"
		},
		"meta": {
		"href": "/api/setup",
		"parent": "/api",
		"remaining_seconds": 0
		}
		}

17

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 details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

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 36.

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.
401	AuthenticationFailure	Authenticating the user with the given credentials has failed.
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 details on authentication, see Authenticate to the SPS REST API on page 18. Note that this session ID refers to the connection between the REST client and the SPS REST API. It is not related to the sessions that SPS records (and which also have a session ID, but in a different format).

Sample request

The following command lists the analytics policies configured.

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

The following command retrieves the properties of a specific policy.

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

Response

The following is a sample response received when listing analytics policies.

For details of the `meta` object, see Message format on page 9.


```
{
    "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"
            "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.

Elements of scoring	Type	Description
fis	string	
hostlogin	string	
keystroke	string	
logintime	string	
windowtitle	string	

Add an analytics policy

To add an analytics policy, complete the following steps.

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Modify an analytics policy

To modify an analytics policy, complete the following steps.

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 details, see Commit a transaction on page 30.

Delete an analytics policy

To delete an analytics policy, complete the following steps.

1. **Open a transaction.**

   For details, see Open a transaction on page 28.

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 42.

   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 30.

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 36.

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 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.
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