



One Identity Safeguard for Privileged Sessions 6.0

DEPRECATED Okta Multi-Factor Authentication - Tutorial

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.

Contents

Introduction	4
How SPS and Okta MFA work together	7
Technical requirements	9
Supported factors and scenarios	11
How SPS and Okta work together in detail	12
Mapping SPS usernames to Okta identities	14
Bypassing Okta authentication	15
Configure your Okta account for SPS	16
Configure SPS to use Okta multi-factor authentication	17
SPS Okta plugin parameter reference	19
[okta]	20
[plugin]	23
[auth]	24
[cache]	25
[ldap]	27
[username_transform]	28
[question_1]	29
Store sensitive plugin data securely	31
Perform multi-factor authentication with the SPS Okta plugin in terminal connections	32
Perform multi-factor authentication with the SPS Okta plugin in Remote Desktop connections	33
About us	35
Contacting us	35
Technical support resources	35

Introduction

This document describes how you can use the services of multi-factor authentication provider [Okta](#) to authenticate the sessions of your privileged users with One Identity Safeguard for Privileged Sessions (SPS).

One Identity Safeguard for Privileged Sessions:

One Identity Safeguard for Privileged Sessions (SPS) controls privileged access to remote IT systems, records activities in searchable, movie-like audit trails, and prevents malicious actions. SPS is a quickly deployable enterprise device, completely independent from clients and servers — integrating seamlessly into existing networks. It captures the activity data necessary for user profiling and enables full user session drill down for forensic investigations.

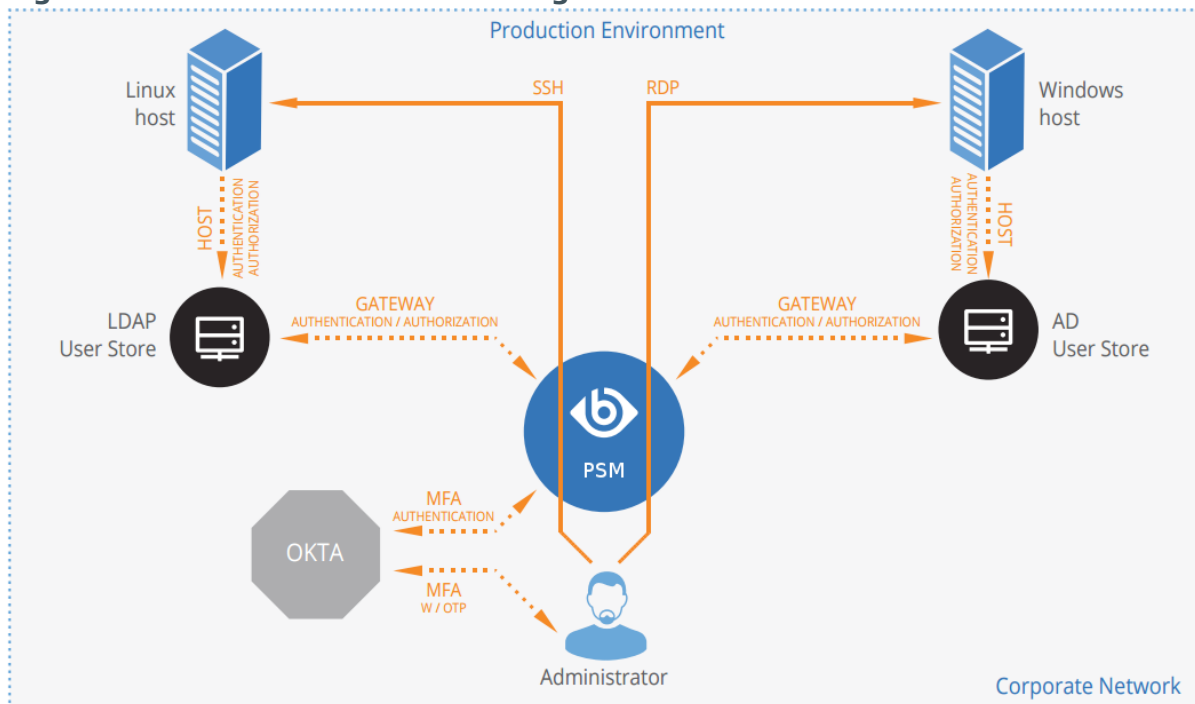
SPS acts as a central authentication gateway, enforcing strong authentication before users access sensitive IT assets. SPS can integrate with remote user directories to resolve the group memberships of users who access nonpublic information. Credentials for accessing information systems can be retrieved transparently from SPS's local credential store or a third-party password management system. This method protects the confidentiality of passwords as users can never access them. When used together with Okta (or another multi-factor authentication provider), SPS directs all connections to the authentication tool, and upon successful authentication, it permits the user to access the information system.

Okta Adaptive Multi-factor Authentication:

To support multi-factor authentication, SPS integrates with the identity management service Okta. This enables you to leverage an additional out-of-band factor (typically through the user's registered smartphone) when authenticating the user. The additional factor is processed in-line with the connection, so users do not have to switch to an external application to process the additional factor. This results in an efficient user experience that is readily accepted by the users.

The One Identity Safeguard for Privileged Sessions can interact with your Okta account and can automatically request multi-factor authentication for your privileged users who are accessing the servers and services protected by SPS.

Figure 1: How SPS and Okta work together



Solution benefits

Matt Magleby, Sr. Computer Scientist at Adobe

One Identity Safeguard for Privileged Sessions helps us meet a portion of our compliance and internal security requirements related to access management.

Using SPS together with Okta provides the following benefits.

- Easy-to-use multi-factor authentication to secure your privileged users who access your business-critical servers. The enforcement of a second-factor authentication and the availability of session recordings make the access of high-risk users more secure.
- Out-of-band authentication to protect against privileged identity theft
- Logs and audits administrative network traffic
- Integrates with existing LDAP user directory
- Easy and fast deployment and implementation: a network-level solution that does not require installing agents on clients or servers
- Can forward user logs into Splunk for long-term storage and analysis
- Supports SSH and RDP protocols to access both Linux and Windows servers, without disrupting the daily workflow of your system administrators and other privileged users
- Supports strong authentication methods, including SSH keys and certificates

Meet compliance requirements

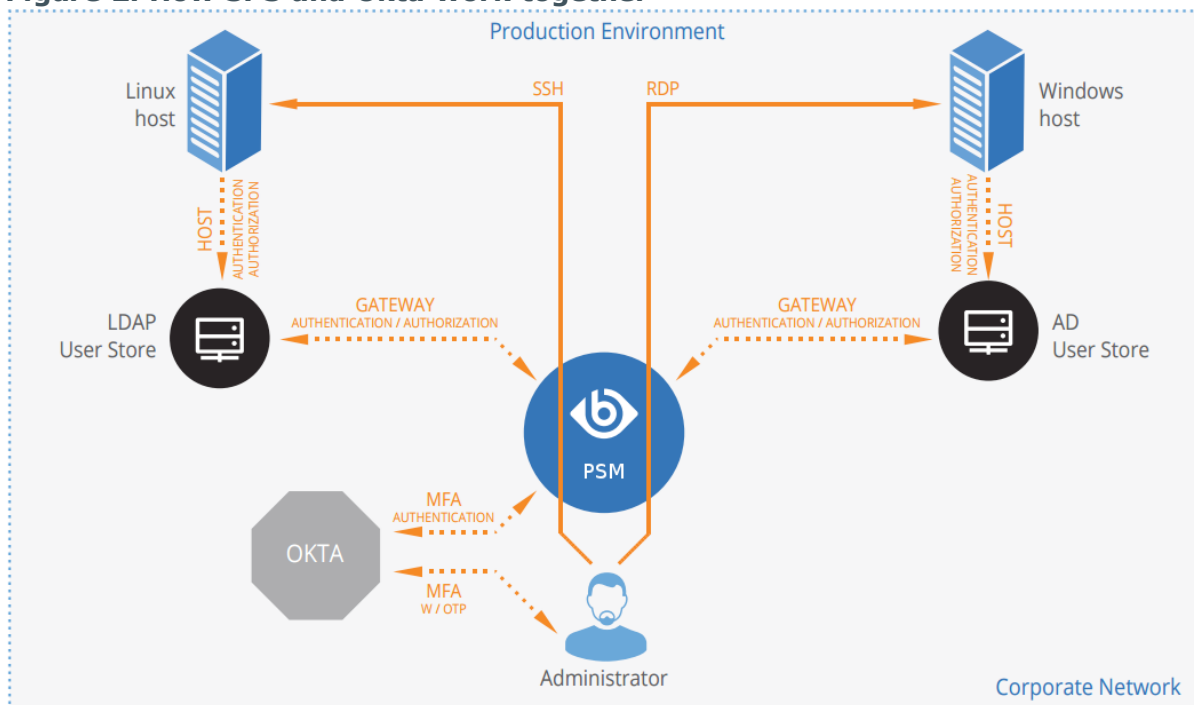
ISO 27001, ISO 27018, SOC 2, and other regulations and industry standards include authentication-related requirements, for example, multi-factor authentication (MFA) for accessing production systems, and the logging of all administrative sessions. In addition to other requirements, using SPS and Okta helps you comply with the following requirements:

- PCI DSS 8.3: Secure all individual non-console administrative access and all access to the cardholder data environment (CDE) using multi-factor authentication.
- PART 500.12 Multi-Factor Authentication: Covered entities are required to apply multi-factor authentication for:
 - Each individual accessing the covered entity's internal systems.
 - Authorized access to database servers that allow access to nonpublic information.
 - Third parties accessing nonpublic information.
- NIST 800-53 IA-2, Identification and Authentication, network access to privileged accounts: The information system implements multi-factor authentication for network access to privileged accounts

To read about the experiences of Adobe Experience Cloud business unit using SPS and Okta, [download our case study](#).

How SPS and Okta MFA work together

Figure 2: How SPS and Okta work together



1. A user attempts to log in to a protected server.

2. Gateway authentication on SPS

SPS receives the connection request and authenticates the user. SPS can authenticate the user to a number of external user directories, for example, LDAP, Microsoft Active Directory, or RADIUS. This authentication is the first factor.

3. Outband authentication on Okta

If gateway authentication is successful, SPS connects the Okta server to check which authentication factors are available for the user. Then SPS requests the second authentication factor from the user.

- For OTP-like authentication factors, SPS requests the one-time password (OTP) from the user, and sends it to the Okta server for verification.
 - For the Okta push notification factor, SPS asks the Okta server to check if the user successfully authenticated on the Okta server.
4. If multi-factor authentication is successful, the user can start working, while SPS records the user's activities. (Optionally, SPS can retrieve credentials from a local or external credential store or password vault, and perform authentication on the server with credentials that are not known to the user.)

Technical requirements

In order to successfully connect SPS with Okta, you need the following components.

In Okta:

- A valid Okta subscription that permits multi-factor authentication.
- An Okta API key. You will need it to configure the SPS plugin.
- Your users must be available in Okta.
- The users must activate their Okta accounts, and be able to perform the authentication required for the factor (for example, install the Okta mobile app, possess the required hardware token, and so on).
- A factor that the SPS plugin supports must be enabled in Okta. Note that this is a global setting, you cannot selectively enable factors. For a list of factors that SPS supports, see [Supported factors and scenarios](#).

In SPS:

- A One Identity Safeguard for Privileged Sessions appliance (virtual or physical), at least version 5 F1.
- A copy of the SPS Okta plugin. This plugin is an Authentication and Authorization (AA) plugin customized to work with the Okta multi-factor authentication service.
- SPS must be able to access the Internet (at least the services on <https://www.okta.com>). Since Okta is a cloud-based service provider, SPS must be able to access its web services to authorize the user.
- Depending on the factor you use to authenticate your users, your users might need Internet access as well, for example, to use the Okta Verify Push Notification factor. Note that the Okta app generates one-time passwords (OTPs) offline.
- SPS supports AA plugins in the RDP, SSH, and Telnet protocols.
- In RDP, using an **AA plugin** together with Network Level Authentication in a Connection Policy has the same limitations as using Network Level Authentication without domain membership. For details, see "[Network Level Authentication without domain membership](#)" in the [Administration Guide](#).

- In RDP, using an **AA plugin** requires TLS-encrypted RDP connections. For details, see ["Enabling TLS-encryption for RDP connections" in the Administration Guide](#).

Availability and support of the plugin

The SPS Okta plugin is available as-is, free of charge to every SPS customer from the [Plugin Page](#). In case you need any customizations or additional features, [contact our Professional Services Team](#).

You can use the plugin on SPS 5 F3 and later. If you need to use the plugin on SPS 5 LTS, [contact our Professional Services Team](#).

Supported factors and scenarios

The SPS Okta plugin can provide multi-factor authentication in the Remote Desktop (RDP), Secure Shell (SSH), and TELNET protocols. In RDP, using push notifications (when the user authenticates using the Okta mobil app) is the most convenient method. You can use another factor, but in this case the user must encode the OTP password into the username used in the connection, before trying to connect to the server.

Okta supports several different authentication backends ("factor types" in Okta terminology). When using one-time passwords (OTP-like factors), your users can specify which factor they use (from the ones available for them in Okta).

You can also set a default factor in the Okta configuration, it defaults to Okta Verify (One-Time Password).

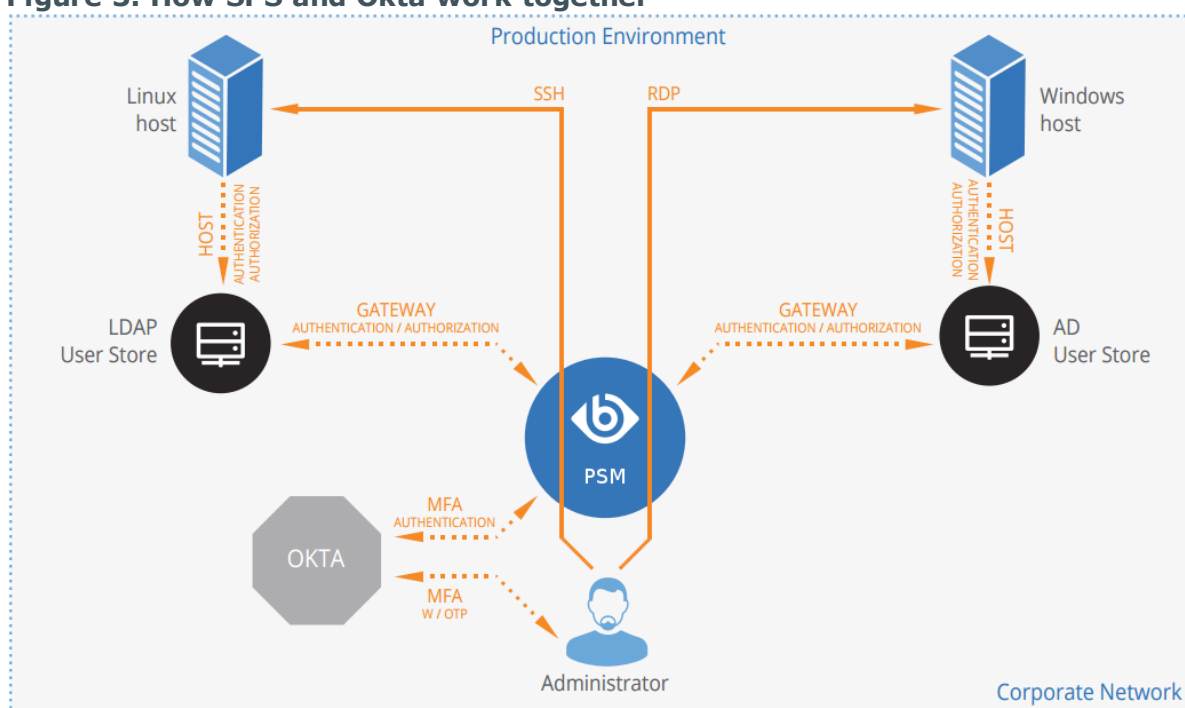
The SPS Okta plugin supports the following authentication factors:

- Google Authenticator (One-Time Password)
- Okta Push Authentication
- Okta Verify (One-Time Password)
- RSA token
- Symantec token (One-Time Password)
- YubiKey token (One-Time Password)

You can also configure SPS to require multi-factor authentication only from selected users or usergroups. You can also specify exceptions, that is, require multi-factor authentication from everyone except the specified users. This allows you to have emergency or break-glass users who can access your servers even if the Okta services are not available. For details, see ["\[auth\]" in the Okta Multi-Factor Authentication - Tutorial](#).

How SPS and Okta work together in detail

Figure 3: How SPS and Okta work together



1. A user attempts to log in to a protected server.

2. Gateway authentication on SPS

SPS receives the connection request and authenticates the user. SPS can authenticate the user to a number of external user directories, for example, LDAP, Microsoft Active Directory, or RADIUS. This authentication is the first factor.

3. Check if the user is exempt from multi-factor authentication

You can configure SPS using whitelists and blacklists to selectively require multi-factor authentication for your users, for example, to create break-glass access for specific users.

- If multi-factor authentication is not required, the user can start working, while SPS records the user's activities. The procedure ends here.
- If multi-factor authentication is required, SPS continues the procedure with the next step.

For details on creating exemption lists, see [whitelist](#).

4. Determining the Okta username

If the gateway usernames are different from the Okta usernames, you must configure the SPS Okta plugin to map the gateway usernames to the Okta usernames. The mapping can be as simple as appending a domain name to the gateway username, or you can query an LDAP or Microsoft Active Directory server. For details, see [Mapping SPS usernames to Okta identities](#).

5. Outband authentication on Okta

If gateway authentication is successful, SPS connects the Okta server to check which authentication factors are available for the user. Then SPS requests the second authentication factor from the user.

- For OTP-like authentication factors, SPS requests the OTP from the user, and sends it to the Okta server for verification.
 - For the Okta push notification factor, SPS asks the Okta server to check if the user successfully authenticated on the Okta server.
6. If multi-factor authentication is successful, the user can start working, while SPS records the user's activities. (Optionally, SPS can retrieve credentials from a local or external credential store or password vault, and perform authentication on the server with credentials that are not known to the user.)
 7. If the user opens a new session within a short period, they can do so without having to perform multi-factor authentication again. After this configurable grace period expires, the user must perform multi-factor authentication to open the next session. For details, see [\[cache\]](#).

Mapping SPS usernames to Okta identities

By default, SPS assumes that the Okta username of the user is the same as the gateway username (that is, the username the user used to authenticate on SPS during the gateway authentication). To identify the users, SPS uses the username (login) field in Okta, which is an email address.

If the gateway usernames are different from the Okta usernames, you must configure the SPS Okta plugin to map the gateway usernames to the Okta usernames. You can use the following methods:

- To simply append a string to the gateway username, configure the [append_domain parameter](#). In this case, SPS automatically appends the @ character and the value of this option to the username from the session, and uses the resulting username on the Okta server to authenticate the user. For example, if the domain is set as `append_domain: example.com` and the username is `Example.User`, the SPS plugin will look for the user `Example.User@example.com` on the Okta server.
- To look up the Okta username of the user from an LDAP/Active Directory database, configure the `[ldap]` section of the SPS Okta plugin. Typically, the SPS plugin queries the email address corresponding to the username from your LDAP or Active Directory database. For details on LDAP parameters, see [\[ldap\]](#).
- If you configure both the [append_domain parameter](#) and the [\[ldap\] section](#) of the SPS Okta plugin, SPS appends the @ character and the value of the `append_domain` parameter to the value retrieved from the LDAP database.
- If you have configured neither the `Domain` parameter nor the `[ldap]` section, SPS assumes that the Okta username of the user is the same as the gateway username.

Bypassing Okta authentication

Having to perform multi-factor authentication to a remote server every time the user opens a session can be tedious and inconvenient for the users, and can impact their productivity. SPS offers the following methods to solve this problem:

- In SPS, the Connection policy determines the type of authentication required to access a server. If you do not need multi-factor authentication for accessing specific servers, configure your Connection policies accordingly.
- If the user opens a new session within a short period, they can do so without having to perform multi-factor authentication. After this configurable grace period expires, the user must perform multi-factor authentication to open the next session. For details, see [\[cache\]](#).
- You can configure SPS using whitelists and blacklists to selectively require multi-factor authentication for your users, for example, to create break-glass access for specific users. For details on creating exemption lists, see [whitelist](#).

Configure your Okta account for SPS

Prerequisites:

- Administrator access to your Okta account.
- Make sure that you have all the required components listed in [Technical requirements](#).

1. Add users to your Okta account.

The users you want to authenticate with SPS must have an activated account in Okta. Navigate to *Directory > People*, and add or import your users. For details, see [A Quick Look at Adding People](#) in the Okta documentation.

2. Enable Multifactor Authentication (MFA) for your organization.

Optionally, you can create a Multifactor Policy in Okta to enable MFA only for the group of users who you want to authenticate with SPS.

When selecting the accepted factor types for your users, make sure to [select at least one factor that SPS supports](#).

For details, see [Multifactor Authentication](#) in the Okta documentation.

3. Create an API token.

Navigate to *Admin > API > Tokens*, click *Create Token*, and save it.

Configure SPS to use Okta multi-factor authentication

Prerequisites:

- Your Okta API token.

⚠ CAUTION:

According to the current Okta policies, your API token expires if it is not used for 30 days. Make sure that you use it regularly, because SPS will reject your sessions if the API token is expired.

- Administrator access to SPS.
- Make sure that you have all the required components listed in [Technical requirements](#).

To configure SPS to use Okta multi-factor authentication

1. Download the SPS Okta plugin

SPS customers can [download the official plugin from GitHub](#).

2. Upload the plugin to SPS

Upload the plugin to SPS. For details, see "[Using a custom Authentication and Authorization plugin to authenticate on the target hosts](#)" in the [Administration Guide](#).

3. Configure the plugin on SPS

The plugin includes a default configuration file, which is an ini-style configuration file with sections and name=value pairs. You can edit it on the **Policies > AA Plugin Configurations** page of the SPS web interface.

- a. Copy your Okta API token and the name of your Okta site in the [OKTA] section of the configuration file, for example:

```
[OKTA]
APIKey=YOUR-OKTA-API-KEY
SiteName=yoursite.okta.com
```

- b. Configure the usermapping settings if needed. SPS must find out which Okta user belongs to the username of the authenticated connection. For that, it can query your LDAP/Microsoft Active Directory server. For details, see [Mapping SPS usernames to Okta identities](#).
- c. Configure other parameters of your plugin as needed for your environment. For details, see [SPS Okta plugin parameter reference](#).

4. Configure a Connection policy and test it

Configure a Connection policy on SPS. In the **AA plugin** field of the Connection policy, select the SPS Okta plugin you configured in the previous step, then start a session to test it. For details on how a user can perform multi-factor authentication, see [Perform multi-factor authentication with the SPS Okta plugin in terminal connections](#) and [Perform multi-factor authentication with the SPS Okta plugin in Remote Desktop connections](#).

 **CAUTION:**

According to the current Okta policies, your API token expires if it is not used for 30 days. Make sure that you use it regularly, because SPS will reject your sessions if the API token is expired.

SPS Okta plugin parameter reference

This section describes the available options of the SPS Okta plugin.

The plugin uses an ini-style configuration file with sections and name=value pairs. This format consists of sections, led by a [section] header and followed by name=value entries. Note that the leading whitespace is removed from values. The values can contain format strings, which refer to other values in the same section. For example, the following section would resolve the %(dir)s value to the value of the dir entry (/var in this case).

```
[section name]
dirname=%(dir)s/mydirectory
dir=/var
```

All reference expansions are done on demand. Lines beginning with # or ; are ignored and may be used to provide comments.

You can edit the configuration file from the SPS web interface. The following code snippet is a sample configuration file.

```
[okta]
# Do NOT use api_key in production
; api_key=YOUR-OKTA-API-KEY
application_id=PSMoktaAAPPlugin/%(VERSION)s
api_url=https://example.okta.com/api/v1/
default_prefix=o
timeout=60
http_socket_timeout=10
rest_poll_interval=1
ignore_conn_err=no

[plugin]
config_version=1
log_level=info
cred_store=<name-of-credstore-storing-sensitive-data>
```

```

[auth]
prompt=Hit Enter to send Okta Verify push notification or provide the OTP:
whitelist=name-of-a-userlist

[username_transform]
append_domain=""

[ldap]
ldap_server_config=<SPS-LDAP-server-policy-name>
filter=(&(cn={})(objectClass=inetOrgPerson))
user_attribute=cn

[cache]
soft_timeout=15
hard_timeout=90
conn_limit=5

[question_1]
key=<name-of-name-value-pair>
prompt=<the-question-itself-in-text>
disable_echo=No

[question_2]...

```

[okta]

This section contains the options related to your Okta account.

```

[okta]
# Do NOT use api_key in production
; api_key=YOUR-OKTA-API-KEY
application_id=PSMOktaAAPugin/%(VERSION)s
site_name=example.okta.com
api_url=https://%(site_name)s/api/v1/
default_prefix=o
http_socket_timeout=10
ignore_conn_err=Yes
rest_poll_interval=1
timeout=55

```

api_key

Type: string

Required:	yes
Default:	N/A

⚠ CAUTION:

This parameter contains sensitive data. Make sure to store this data in your local Credential Store. Type the \$ value for this parameter in production.

For details, see "Store sensitive plugin data securely".

Only enter a value different than \$ for this parameter in the configuration for testing purposes in a secure, non-production environment.

Description: Your Okta API key. SPS uses this to communicate with the Okta server. For details on using a local Credential Store to host this data, read [Store sensitive plugin data securely](#).

⚠ CAUTION:

According to the current Okta policies, your API token expires if it is not used for 30 days. Make sure that you use it regularly, because SPS will reject your sessions if the API token is expired.

application_id

Type:	string
Required:	no
Default:	PSMOktaMFA/1.0

Description: The application ID used in the communication with the Okta server. This ID is visible in the Okta logs.

api_url

Type:	string
Required:	yes
Default:	N/A

Description: The URL where the Okta server can be accessed. Usually you can use the default value:

```
api_url=https://example.okta.com/api/v1/
```

To override the access URL for the Okta API, change the value.

default_prefix

Type:	string
Required:	no
Default:	o

Description: If the user uses an OTP-like factor, and does not specify the type of factor in the OTP string, the SPS plugin assumes that the OTP is for the default factor. The possible values are as follows:

- Google Authenticator: g
- inWebo Authenticator: o
- Symantec token: s
- YubiKey: y
- RSA token: r

If you do not set this option and the user does not specify an OTP type, the plugin assumes that the OTP received from the user is an Okta OTP.

timeout

Type:	integer [seconds]
Required:	no
Default:	60

Description: How long the authentication process can take during the communication with the Okta server (potentially consisting of multiple HTTP requests).

http_socket_timeout

Type:	integer [seconds]
Required:	no
Default:	10

Description: How long the plugin waits for an approval when using the Okta push notification factor. This option sets the timeframe (measured from the user initiating the connection to SPS) within which SPS must receive the approval from the Okta server. SPS [periodically asks the Okta server](#) to check if the user successfully authenticated on the Okta server.

rest_poll_interval

Type:	integer [seconds]
Required:	no
Default:	1

Description: How often the plugin checks the Okta server to see if the push notification was successful. Note that SPS rejects the connection of the user if it does not receive an approval for the push notification within the period set in [http_socket_timeout](#).

ignore_conn_err

Type:	yes no
Required:	no
Default:	no

Description: Determines how to handle the sessions if the Okta service is not available. If set to *yes*, the plugin assumes that the user successfully authenticated even if the plugin cannot access Okta to verify this.

CAUTION:

Enabling this option allows the users to bypass multi-factor authentication if SPS cannot access the Okta service for any reason, for example, a network configuration error in your environment.

[plugin]

This section contains general plugin-related settings.

```
[plugin]
config_version=1
log_level=20
cred_store=<name-of-credstore-hosting-sensitive-data>
```

config_version

Type:	integer
Required:	yes
Default:	1

Description: The version number of the configuration format. This is used to enable potentially incompatible changes in the future. If provided, the configuration will not be upgraded automatically. If not provided, the configuration will be upgraded automatically.

cred_store

Type:	string
Required:	no
Default:	N/A

Description: The name of a local credential store policy configured on SPS (**AAA > Credential Stores > <name-of-credential-store>**). You can use this credential store to store sensitive information of the plugin in a secure way, for example, the Okta API key. For details, see [Store sensitive plugin data securely](#).

log_level

Type:	integer or string
Required:	no
Default:	info

Description: The logging verbosity of the plugin. The plugin sends the generated log messages to the SPS syslog system. You can check the log messages in the **Basic settings > Troubleshooting > View log files** section of the SPS web interface. Filter on the plugin: string to show only the messages generated by the plugins.

The possible values are:

- debug or 10
- info or 20
- warning or 30
- error or 40
- critical or 50

For details, see Python logging API's log levels: [Logging Levels](#).

[auth]

This section contains the options related to authentication.


```
[auth]
prompt=Hit Enter to send Okta Verify push notification or provide the OTP:
whitelist=name-of-a-userlist
```

prompt

Type:	string
Required:	no
Default:	Hit Enter to send push notification or provide the OTP:

Description: SPS displays this text to the user in a terminal connection to request an OTP interactively. The text is displayed only if the user uses an OTP-like factor, and does not send the OTP in the connection request.

```
prompt=Hit Enter to send Okta Verify push notification or provide the OTP:
```

whitelist

Type:	string
Required:	no
Default:	N/A

Description: The name of a user list containing gateway users configured on SPS (**Policies > User Lists**). You can use this option to selectively require multi-factor authentication for your users, for example, to create break-glass access for specific users.

- If you set the **Default Policy** of the user list to Reject, then the list is a whitelist, so the plugin will not request Okta authentication from the users on the list.
- If you set the **Default Policy** of the user list to Accept, then the list is a blacklist, so the plugin will request Okta authentication only from the users on the list.

For details on creating user lists, see ["Creating and editing user lists" in the Administration Guide](#).

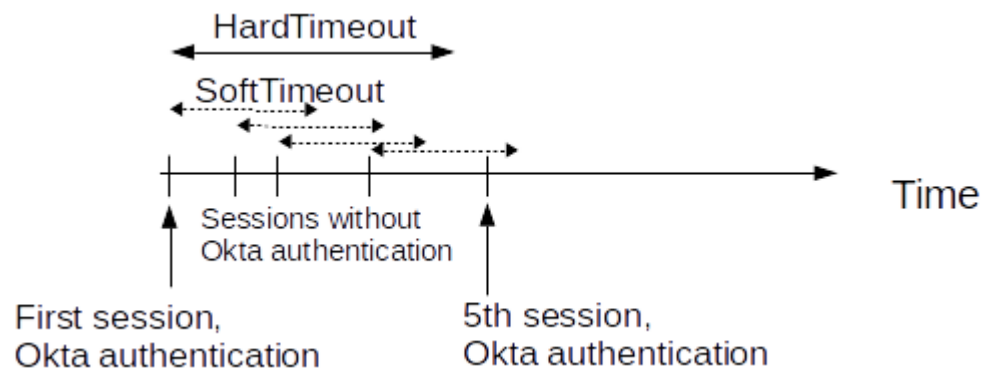
[cache]

This section contains the settings that determine how soon after performing an Okta authentication must the user repeat the authentication when opening a new session.

After the first Okta authentication of the user, SPS will not request a new Okta authentication from the user as long as the new authentications would happen within soft_

timeout seconds from each other. After the `hard_timeout` expires (measured from the first Okta login of the user), SPS will request a new Okta authentication.

In other words, after opening the first session and authenticating on Okta, the user can keep opening other sessions without having to authenticate again on Okta as long as the time between opening any two sessions is less than `soft_timeout`, but must authenticate on Okta if `hard_timeout` expires.



```
[cache]
soft_timeout=15
hard_timeout=90
```

soft_timeout

Type:	integer [seconds]
Required:	yes, if you want caching
Default:	N/A

Description: The time in seconds after which the SPS plugin requires a new Okta authentication for the next new session of the user, unless the user successfully authenticates another session within this period. Recommended value: 900.

If you want caching, both values are required, and the value of `soft_timeout` has to be smaller than the value of `hard_timeout`.

hard_timeout

Type:	integer [seconds]
Required:	yes, if you want caching
Default:	N/A

Description: The time in seconds after which the SPS plugin requires a new Okta authentication for the next new session of the user. The time is measured from the last Okta authentication of the user. Recommended value: 3600.

If you want caching, both values are required, and the value of soft timeout has to be smaller than the value of hard timeout.

[ldap]

This section contains the settings you configure when you need to use an LDAP query to map the usernames from your audited sessions to the usernames in Okta.

To look up the Okta username of the user from an LDAP/Active Directory database, configure the [ldap] section of the SPS Okta plugin. Typically, the SPS plugin queries the email address corresponding to the username from your LDAP or Active Directory database. For details on LDAP parameters, see [ldap].

If you configure both the [append_domain parameter](#) and the [ldap] section of the SPS Okta plugin, SPS appends the @ character and the value of the append_domain parameter to the value retrieved from the LDAP database.

For other methods of mapping gateway usernames to Okta usernames, see [Mapping SPS usernames to Okta identities](#).

```
[ldap]
ldap_server_config=<SPS-LDAP-server-policy-name>
filter=(&(cn={})(objectClass=inetOrgPerson))
user_attribute=CN
```

ldap_server_config

Type:	string
Required:	no
Default:	N/A

Description: The name of a configured LDAP server policy in SPS. For details on configuring LDAP policies, see "[Authenticating users to an LDAP server](#)" in the [Administration Guide](#).

filter

Type:	string
Required:	no
Default:	(&(cn={})(objectClass=inetOrgPerson))

Description: The LDAP filter query that locates the user based on the gateway username. The plugin automatically replaces the {} characters with the gateway username from the session.

```
filter=(&(cn={})(objectClass=inetOrgPerson))
```

user_attribute

Type:	string
Required:	no
Default:	cn

Description: The name of the LDAP attribute that contains the Okta username.

[username_transform]

This section contains username transformation-related settings.

```
[username_transform]  
append_domain=""
```

append_domain

Type:	string (nonrequired, no default)
Required:	no
Default:	N/A

Description: If the gateway usernames are different from the Okta usernames, you must configure the SPS Okta plugin to map the gateway usernames to the Okta usernames.

To simply append a string to the gateway username, configure the [append_domain parameter](#). In this case, SPS automatically appends the @ character and the value of this option to the username from the session, and uses the resulting username on the Okta server to authenticate the user. For example, if the domain is set as `append_domain: example.com` and the username is `Example.User`, the SPS plugin will look for the user `Example.User@example.com` on the Okta server.

If you configure both the [append_domain parameter](#) and the [\[ldap\] section](#) of the SPS Okta plugin, SPS appends the @ character and the value of the `append_domain` parameter to the value retrieved from the LDAP database.

For other methods of mapping gateway usernames to Okta usernames, see [Mapping SPS usernames to Okta identities](#).

[question_1]

Type: integer [seconds]

Description: Used for communication between plugins. This is an interactive request/response right after authentication in order to supply data to credential store plugins. The question is transferred to the session cookie and all hooks of all plugins receive it.

For example, if you have an external authenticator app, you do not have to wait for the question to be prompted but can authenticate with a one-time password:

```
ssh otp=123456@root@scb
```

Name subsequent questions with the appropriate number, for example, [question_1], [question_2], and so on.

For details, see ["Performing authentication with AA plugin in terminal connections" in the Administration Guide](#) and ["Performing authentication with AA plugin in Remote Desktop connections" in the Administration Guide](#).

key

Type:	string
Required:	yes
Default:	N/A

Description: The name of the name-value pair.

prompt

Type:	string
Required:	yes
Default:	N/A

Description: The question itself in text format.

disable_echo

Type:	boolean yes no
Required:	no
Default:	no

Description: Whether the answer to the question is visible (yes), or replaced with asterisks (no).

Store sensitive plugin data securely

By default, the configuration of the plugin is stored on SPS in the configuration of SPS. Make sure that you store the sensitive parameters (for example, `api_key`) of the plugin in an encrypted way.

To store sensitive plugin data securely

The following describes how to

1. Log in to SPS and create a local Credential Store. For details, see "[Configuring local Credential Stores](#)" in the [Administration Guide](#).
Instead of usernames and passwords, you will store the configuration parameters of the plugin in this Credential Store.
2. Add the plugin parameters you want to store in an encrypted way to the Credential Store. You can store any configuration parameter of the plugin in the Credential Store, but note that if an option appears in the Credential Store, the plugin will use it. If the same parameter appears in the configuration of the plugin, it will be ignored.
 - Enter the name of the configuration section without the brackets in the **HOST** field (for example, `okta`).
 - Enter the name of the plugin parameter in the **USERNAME** field (for example, `api_key`).
 - Enter the value of the plugin parameter in the **PASSWORD** field.
3. Commit your changes, and navigate to the configuration of the plugin on the **Policies > AA Plugin Configurations** page.
4. In the plugin configuration file, enter the name of the local Credential Store under the `[plugin]` section, in the `cred_store` parameter.

Perform multi-factor authentication with the SPS Okta plugin in terminal connections

The following describes how to establish a terminal connection (SSH, TELNET, or TN3270) to a server.

To establish a terminal connection (SSH, TELNET, or TN3270) to a server

1. Connect to the server.
 - If you can authenticate using an OTP or token, encode the OTP as part of the username. You can use the @ as a field separator. For example:

```
ssh otp=YOUR-ONE-TIME-PASSWORD@user@server
```

Replace YOUR-ONE-TIME-PASSWORD with your actual OTP. If needed, you can specify the type of OTP as a prefix to the OTP. For example, to specify the OTP of a YubiKey token:

```
ssh otp=y_ YOUR-ONE-TIME-PASSWORD@user@server
```

- Google Authenticator: g
 - inWebo Authenticator: o
 - Symantec token: s
 - YubiKey: y
 - RSA token: r
- If you need to authenticate using the Okta Verify push notification, approve the connection in your mobile app.
2. If SPS prompts you for further information, enter the requested information. If you need to authenticate with an OTP, but you have not supplied the OTP in your username, you will be prompted to enter the OTP.
 3. Authenticate on the server.
 4. If authentication is successful, you can access the server.

Perform multi-factor authentication with the SPS Okta plugin in Remote Desktop connections

The following describes how to establish a Remote Desktop (RDP) connection to a server when the **AA plugin** is configured.

To establish a RDP connection to a server when the AA plugin is configured

1. Open your Remote Desktop client application.
2. If you have to provide additional information to authenticate on the server, you must enter this information in your Remote Desktop client application in the *User name* field, before the regular content (for example, your username) of the field.

If you can authenticate using an OTP or token, encode the OTP as part of the username. To encode additional data, you can use the following special characters:

- % as a field separator
- ~ as the equal sign
- ^ as a colon (for example, to specify the port number or an IPv6 IP address)

For example, use the following format:

```
domain\otp~YOUR-ONE-TIME-PASSWORD%Administrator
```

Replace YOUR-ONE-TIME-PASSWORD with your actual OTP. If needed, you can specify the type of OTP as a prefix to the OTP. For example, to specify the OTP of a YubiKey token: `domain\otp~y_YOUR-ONE-TIME-PASSWORD%Administrator`

- Google Authenticator: g
- inWebo Authenticator: o
- Symantec token: s
- YubiKey: y
- RSA token: r

3. Connect to the server.

If you need to authenticate using the Okta Verify push notification, approve the connection in your mobile app.

4. Authenticate on the server.
5. If authentication is successful, you can access the server.

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