One Identity Safeguard for Privileged Sessions 6.7.2

Okta Multi-Factor Authentication - Tutorial

SPS Okta Multi-Factor Authentication - Tutorial
Updated - 24 November 2020, 09:01
Version - 6.7.2

Contents

Introduction **5**

Technical requirements **7**

Supported factors and scenarios **9**

How SPS and Okta work together in detail **10**

Notable features **12**

Configure your Okta account for SPS **13**

Configure SPS to use Okta multi-factor authentication **14**

SPS Okta plugin parameter reference **16**

[okta] 18

[auth] 21

[connection_limit by=client_ip_gateway_user] 22

[authentication_cache] 22

[WHITELIST] 24

[whitelist source=user_list] 24

[whitelist source=ldap_server_group] 25

[USERMAPPING] 27

[usermapping source=explicit] 28

[usermapping source=ldap_server] 28

[username_transform] 29

[ldap_server] 30

[credential_store] 30

[logging] 31

[https_proxy] 32

[question_1] 32

Store sensitive plugin data securely **34**

Perform multi-factor authentication with the SPS Okta plugin in terminal connections **35**

Perform multi-factor authentication with the SPS Okta plugin in Remote **36**

Desktop (RDP) connections .....

Perform multi-factor authentication with the SPS Okta plugin in Microsoft SQL Server (MSSQL) connections **38**

About us **40**

Contacting us 40

Technical support resources 40

Introduction

This document describes how you can use the services of 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 (MFA) 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:

Okta is a cloud-based identity service offering identity, authentication, and access control functions as a service. To support multi-factor authentication, SPS integrates with the Okta identity management service. 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 a seamless and 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 strong multi-factor authentication for your privileged users who are accessing the servers and services protected by SPS. Okta supports a broad range of authentication methods, including software, hardware, and cellphone-based solutions.

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 secures privileged access to business-critical servers. The enforcement of a second-factor authentication and the availability of session recordings make access of high-risk users more secure.
- Outband 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 remote access to the cardholder data environment (CDE) using MFA.
- PART 500.12 Multi-Factor Authentication: Covered entities are required to apply MFA 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 MFA for network access to privileged accounts.

Technical requirements

In order to successfully connect SPS with RADIUS server, 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 SPS 5.11.0.
- A copy of the SPS Okta Multi-Factor Authentication 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 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 Multi-Factor Authentication plugin is available for download as-is, free of charge to every SPS customer from the Okta Multi-Factor Authentication plugin for

Safeguard for Privileged Sessions page. In case you need any customizations or additional features, contact our Support Team.

CAUTION:

Using custom plugins in SPS is recommended only if you are familiar with both Python and SPS. Product support applies only to SPS: that is, until the entry point of the Python code and passing the specified arguments to the Python code. One Identity is not responsible for the quality, resource requirements, or any bugs in the Python code, nor any crashes, service outages, or any other damage caused by the improper use of this feature, unless explicitly stated in a contract with One Identity. If you want to create a custom plugin, contact our Support Team for details and instructions.

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).

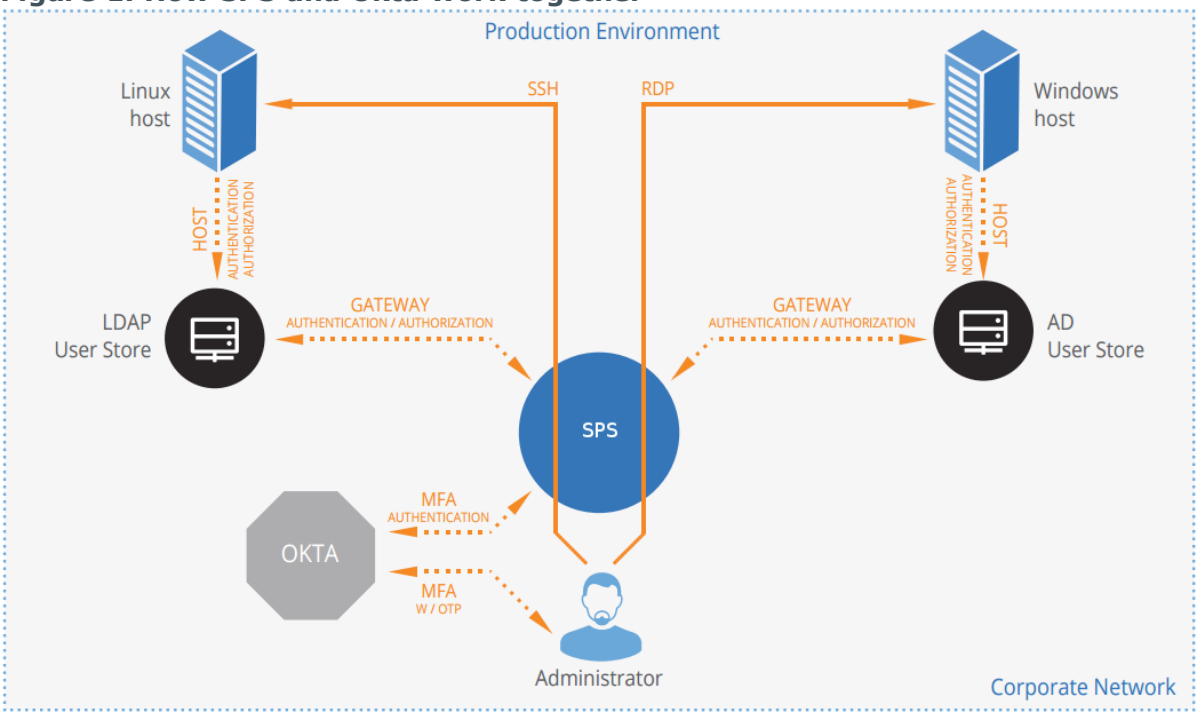
The SPS Okta plugin supports the following authentication factors:

- 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 "[WHITELIST]" in the Okta Multi-Factor Authentication - Tutorial.

How SPS and Okta work together in detail

Figure 1: 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] on page 24.

4. **Determining the external Okta identity**

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

   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 [USERMAPPING] on page 27.

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 [authentication_cache] on page 22.

Notable features

This section contains the notable features of this plugin.

- To map the gateway usernames to the external Okta identities if the gateway usernames are different from the Okta usernames, configure the [USERMAPPING] on page 27 section of the plugin.
- The [WHITELIST] on page 24 section allows configuring authentication whitelists and blacklists for example to create break-glass access for specific users to allow them to bypass Okta authentication.
- The [authentication_cache] on page 22 section contains the settings that determine how soon after performing a Okta authentication must the user repeat the authentication when opening a new session.
- The [connection_limit by=client_ip_gateway_user] on page 22 section contains the options related to limiting parallel sessions.

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 [USERMAPPING].

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 (RDP) 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.

8

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]
api_key=$
application_id=PSMOktaAAPlugin/%(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

[auth]
prompt=Press Enter for push notification or type one-time password:
disable_echo=yes

[connection_limit by=client_ip_gateway_user]
limit=0
```

```
[authentication_cache]
soft_timeout=15
hard_timeout=90
conn_limit=5

######[WHITELIST]######

[whitelist source=user_list]
name=<name-of-user-list-policy>

[whitelist source=ldap_server_group]
allow=no_user
except=<group-1>,<group-2>

######[USERMAPPING]######

[usermapping source=explicit]
<user-name-1>=<id-1>
<user-name-2>=<id-2>

[usermapping source=ldap_server]
user_attribute=description

[username_transform]
append_domain=<domain-without-@-character>

[ldap_server]
name=<name-of-LDAP-server-policy>

[credential_store]
name=<name-of-credential-store-policy-that-hosts-sensitive-data>

[logging]
log_level=info

[https_proxy]
server=<proxy-server-name-or-ip>
port=3128

[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=PSMOktaAAPlugin/%(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.

[auth]

This section contains the options related to authentication.

Declaration

```
[auth]
prompt=Press Enter for push notification or type one-time password:
disable_echo=yes
```

prompt

Type:	string
Required:	no
Default:	Press Enter for push notification or type one-time password:

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.

disable_echo

Type:	boolean (`yes`|`no`)
Required:	no
Default:	no

Description: For better security, you can hide the characters (OTP or password) that the user types after the prompt. To hide the characters (replace them with asterisks), set `disable_echo` to `yes`.

[connection_limit by=client_ip_ gateway_user]

This section contains the options related to limiting parallel sessions.

Declaration

```
[connection_limit  by=client_ip_gateway_user]
 limit=0
```

limit

Type:	integer
Required:	no
Default:	0

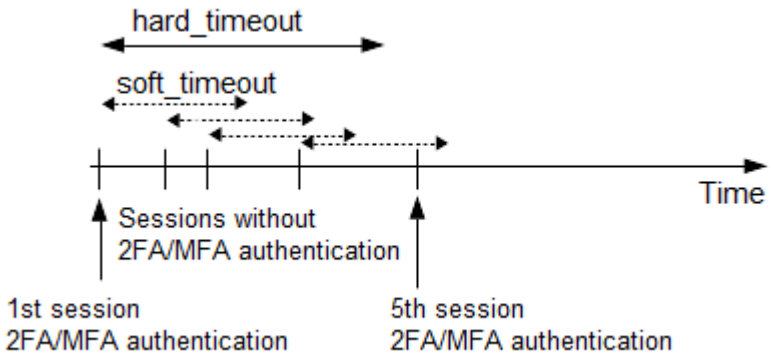
Description: To limit the number of parallel sessions the gateway user can start from a given client IP address, configure `limit`. For an unlimited number of sessions, type `0`.

[authentication_cache]

This section contains the settings that determine how soon after performing a 2FA/MFA authentication the user must 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 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.



Declaration

```
[authentication_cache]
soft_timeout=15
hard_timeout=90
conn_limit=5
```

soft_timeout

Type:	integer [in 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.

hard_timeout

Type:	integer [in 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.

conn_limit

Type:	integer [number of]

Description: The cache can be used `conn_limit` times without multi-factor authentication. If the number of logins exceeds this number, the plugin will request multi-factor authentication again. If this parameter is not set, the number of logins from cache are unlimited.

[WHITELIST]

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 `[authentication_cache]`.
- The `[whitelist source=user_list]` and `[whitelist source=ldap_server_group]` sections allow configuring authentication whitelists and blacklists based on a **User List** policy or an **LDAP Server** policy. These two sections are independent, therefore any of the two can be configured and, for example, can create break-glass access for specific users to allow them to bypass Okta authentication.

[whitelist source=user_list]

The `[whitelist source=user_list]` section allows whitelisting users based on a **User List** policy configured in SPS (**Policies > User Lists**). To enable this whitelist, configure one of the use cases below.

> NOTE:
>
> The user names are compared to the **User List** in a case-sensitive manner.

Declaration

```
[whitelist source=user_list]
name=<name-of-user-list-policy>
```

For details on creating user lists, see "Creating and editing user lists" in the Administration Guide.

name

Type:	string
Required:	no
Default:	N/A

Description: The name of a **User List** policy 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).

Use case #1: Allow no user except certain users

To allow specific users to connect without providing Okta credentials, the **User List** policy should have the following settings:

- Set **Allow** to **No user** and list the users in the **Except** list.
- Then type the name of this **User List** policy as the value of the `name` parameter.

Use case #2: Allow all users except certain users

To enforce Okta authentication for selected users, the **User List** policy should have the following settings:

- Set **Allow** to **All users** and list the users in the **Except** list.
- Then type the name of this **User List** policy as the value of the `name` parameter.

[whitelist source=ldap_server_group]

The `[whitelist source=ldap_server_group]` section allows whitelisting users based on **LDAP Server** group membership. To enable this whitelist, configure one of the use cases below.

NOTE:

The user names and groups are compared in LDAP in a case-insensitive manner.

Declaration

```
[whitelist source=ldap_server_group]
allow=<no_user-or-all_users>
except=<group-1>,<group-2>
```

allow

Type:	string (`all_users` | `no_users`)
Required:	no
Default:	N/A

Description: This parameter defines whether to allow all users or no user to connect without providing Okta credentials. Used together with the `except` parameter, you can define specific LDAP/AD group(s) that are exempt from this rule.

except

Type:	string
Required:	no
Default:	N/A

Description: This parameter defines those specific LDAP/AD group(s) that are exempt from the rule defined by the `allow` parameter.

Use case #1: Allow no user except members of specific group(s)

To allow members of specific LDAP/AD group(s) to connect without providing Okta credentials, type the names of these LDAP/AD groups as values of the `except` parameter and set the `allow` parameter to `no_user`:

```
[whitelist source=ldap_server_group]
allow=<no_user>
except=<group-1>,<group-2>
```

You must configure the name of the LDAP Server policy in the `[ldap_server]` section.

Use case #2: Allow all users except members of specific group(s)

To enforce Okta authentication only on members of specific LDAP/AD group(s), type the names of these LDAP/AD groups as values of the `except` parameter and set the `allow` parameter to `all_users`:

```
[whitelist source=ldap_server_group]
allow=<all_users>
except=<group-1>,<group-2>
```

You must configure the name of the LDAP Server policy in the `[ldap_server]` section.

[USERMAPPING]

By default, SPS assumes that the external Okta identity 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). If there was no gateway authentication, then the server username is used for authentication.

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

You can use the following methods:

- Explicit mapping: `[usermapping source=explicit]`
- LDAP server mapping: `[usermapping source=ldap]`

  To look up the external Okta identity of the user from an LDAP/Active Directory database, configure the `[usermapping source=ldap_server]` section of the SPS Okta plugin.

  If the Okta service requires the use of domain name in the external Okta identity, configure the `append_domain` parameter in the `[username_transform]` section. 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 to `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 in the `[username_transform]` section and the `[usermapping source=ldap_server]` 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.

The Explicit method has priority over the LDAP server method.

If you have configured neither the `append_domain` parameter nor any of the `[USERMAPPING]` sections, SPS assumes that the external Okta identity of the user is the same as the gateway username.

[usermapping source=explicit]

To map the gateway user name to an external Okta identity, configure the following name-value pairs.

Declaration

```
[usermapping source=explicit]
<example-user-1>=<ID-1>
<example-user-2>=<ID-2>
```

<exampleuser>

Type:	string
Required:	no
Default:	N/A

Description: To map the gateway user name to an external Okta identity, configure the name-value pairs in the following way:

- Type the gateway user name instead of `<example-user-1>`.
- Type the external Okta ID instead of `<ID-1>`.

NOTE:

Use this option only if there are not only a few users, or for testing purposes. If there are too many users, it can cause performance issues.

[usermapping source=ldap_server]

To look up the external Okta identity of the user from an LDAP/Active Directory database, configure the `[usermapping source=ldap_server]` section of the SPS Okta plugin.

Declaration

```
[usermapping source=ldap_server]
user_attribute=description
```

You must configure the name of the LDAP Server policy in the `[ldap_server]` section.

If you configure both the `append_domain` parameter in the `[username_transform]` section and the `[usermapping source=ldap_server]` 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.

user_attribute

Type:	string
Required:	no
Default:	N/A

Description: The `user_attribute` must be an LDAP/AD user attribute (with a non-empty UTF8 attribute string) that contains the external identity. For example, `description`, `cn`, `mail`. For a complete list see the User class section of the Active Directory Schema document.

[username_transform]

This section contains username transformation-related settings.

Declaration

```
[username_transform]
append_domain=<domain-without-@-character>
```

If you have configured `[USERMAPPING]`, the `[username_transform]` process will run after the `[USERMAPPING]` process.

append_domain

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

Description:

If the Okta service requires the use of domain name in the external Okta identity, configure the `append_domain` parameter in the `[username_transform]` section. 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 to `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 in the `[username_transform]` section and the `[usermapping source=ldap_server]` 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.

[ldap_server]

The LDAP Server policy that you want to use in an LDAP server usermapping source or an LDAP server group whitelist source. Required if you have configured [usermapping source=ldap_server] on page 28 and [whitelist source=ldap_server_group] on page 25.

Declaration

```
[ldap_server]
name=<name-of-LDAP-server-policy>
```

name

Type:	string
Required:	conditional
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.

[credential_store]

This section contains settings related to storing sensitive information of the plugin.

Declaration

```
[credential_store]
name=<name-of-credential-store-policy-that-hosts-sensitive-data>
```

name

Type:	string
Required:	no
Default:	N/A

Description: The name of a local Credential Store policy configured on SPS. You can use this Credential Store to store sensitive information of the plugin in a secure way (for example, the `api_key` value in the `[okta]` section).

For details, see Store sensitive plugin data securely.

[logging]

This section contains logging-related settings.

Declaration

```
[logging]
log_level=info
```

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. To show only the messages generated by the plugins, filter on the `plugin:` string.

The possible values are:

- `debug`
- `info`
- `warning`
- `error`
- `critical`

For details, see Python logging API's log levels: Logging Levels.

[https_proxy]

This section contains HTTPS proxy-related settings.

Declaration

```
[https_proxy]
 server=<proxy-server-name-or-ip>
 port=3128
```

server

Type:	string
Required:	no
Default:	N/A

Description: The name or IP address of the HTTPS proxy server.

name

Type:	integer
Required:	no
Default:	3128

Description: The port number of the HTTPS proxy server.

[question_1]

> NOTE:
>
> To configure this optional section, contact our Support Team.
>
> To request additional information from the user (for example, ticket number), define one or more `[question_]` section (for example, `[question_1]`, `[question_2]`). The user input will be stored under the value of `key` in the `questions` section of the session cookie.

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.

prompt

Type:	string
Required:	yes
Default:	N/A

Description: The question itself in text format.

key

Type:	string
Required:	yes
Default:	N/A

Description: The name of the name-value pair.

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

1. Log in to SPS, navigate to **Policies > Credential Stores** 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 **Passwords** field.
   - Click 

.
3. 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 `[credential_store]` section as the value of the `name` parameter.
5. Enter `$` as the value of the parameter storing sensitive data.

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.

   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`

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 (RDP) connections

The following section 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)

   Example:

   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 a push notification, approve the connection in your mobile app.

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

Perform multi-factor authentication with the SPS Okta plugin in Microsoft SQL Server (MSSQL) connections

The following section describes how to establish a Microsoft SQL Server (MSSQL) connection to a server when the **AA plugin** is configured.

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

1. Open your SQL client application.
2. If you have to provide additional information to authenticate on the server, you must enter this information in your SQL 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)

   Example:

   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 a push notification, approve the connection in your mobile app.

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

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