One Identity Safeguard for Privileged Sessions 6.9.5

RADIUS Multi-Factor Authentication - Tutorial

Copyright 2023 One Identity LLC.

ALL RIGHTS RESERVED.

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

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

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

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

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

Patents

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

Trademarks

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

Legend

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

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

SPSRADIUS Multi-Factor Authentication - Tutorial
Updated - 08 March 2023, 10:07

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

Contents

Introduction **5**

Technical requirements **7**

How SPS and RADIUS server work together in detail **9**

Notable features **11**

Configure your RSA account for SPS **12**

Configure SPS to use RADIUS multi-factor authentication **13**

SPS RADIUS plugin parameter reference **15**

[radius] 17

[auth] 19

[connection_limit by=client_ip_gateway_user] 20

[authentication_cache] 21

[WHITELIST] 22

[whitelist source=user_list] 23

[whitelist source=ldap_server_group] 24

[USERMAPPING] 26

[usermapping source=explicit] 26

[usermapping source=ldap_server] 27

[username_transform] 28

[ldap_server] 29

[credential_store] 29

[logging] 30

[https_proxy] 30

[question_1] 31

Store sensitive plugin data securely **33**

Perform multi-factor authentication with the SPS RADIUS plugin in terminal connections **34**

Perform multi-factor authentication with the SPS RADIUS plugin in Remote Desktop (RDP) connections **35**

Perform multi-factor authentication with the SPS RADIUS plugin in Microsoft SQL Server (MSSQL) connections **36**

About us **37**

Contacting us **38**

Technical support resources **39**

Introduction

This document describes how you can use the services of RADIUS server (for example, RSA SecurID Access and FreeRADIUS) 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 RADIUS server (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.

Integrating RSA with SPS:

SPS can interact with your RSA Authentication Manager and can automatically request strong Multi-Factor Authentication for your privileged users who are accessing the servers and services protected by SPS. When used together with RSA SecurID Access, SPS prompts the user for a second factor authentication, and upon successful authentication, it permits the user to access the information system.

The integration adds an additional security layer to the gateway authentication performed on SPS. If the user has an RSA SecurID Hardware Token, the user can generate a One-Time Password (OTP) using the device. This will be used for the authentication to the One Identity platform. The one-time password is changed after 60 seconds.

Integrating a generic RADIUS server with SPS:

SPS can interact with your RADIUS server and can automatically request strong multi-factor authentication for your privileged users who are accessing the servers and services protected by SPS.

The integration adds an additional security layer to the gateway authentication performed on SPS. Your RADIUS server can use any hardware or software token as long as it provides standard RADIUS interface.

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 RADIUS server 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 RSA:

- An RSA Authentication Manager deployed.
- RADIUS access parameters, (for example, host, port, and an RSA shared secret). You will need it to configure the SPS plugin.
- Your users must be enrolled in RSA Authentication Manager.
- The users must be able to perform the authentication required for the factor (for example, possess the required RSA SecurID Hardware Token).
- Only CHAP and PAP authentication methods are supported.

In a generic RADIUS server:

- RADIUS access parameters, (for example, host, port, and a shared secret). You will need it to configure the SPS plugin.
- Your users must be enrolled in the RADIUS server or its backend.
- Only CHAP and PAP authentication methods are supported.

In SPS:

- A One Identity Safeguard for Privileged Sessions appliance (virtual or physical), at least version SPS 5.11.05.11.0.
- A copy of the SPS RADIUS (RSA) Multi-Factor Authentication plugin. This plugin is an Authentication and Authorization (AA) plugin customized to work with the RADIUS (RSA) multi-factor authentication service.
- SPS supports AA plugins in the MSSQL, 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.
- 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

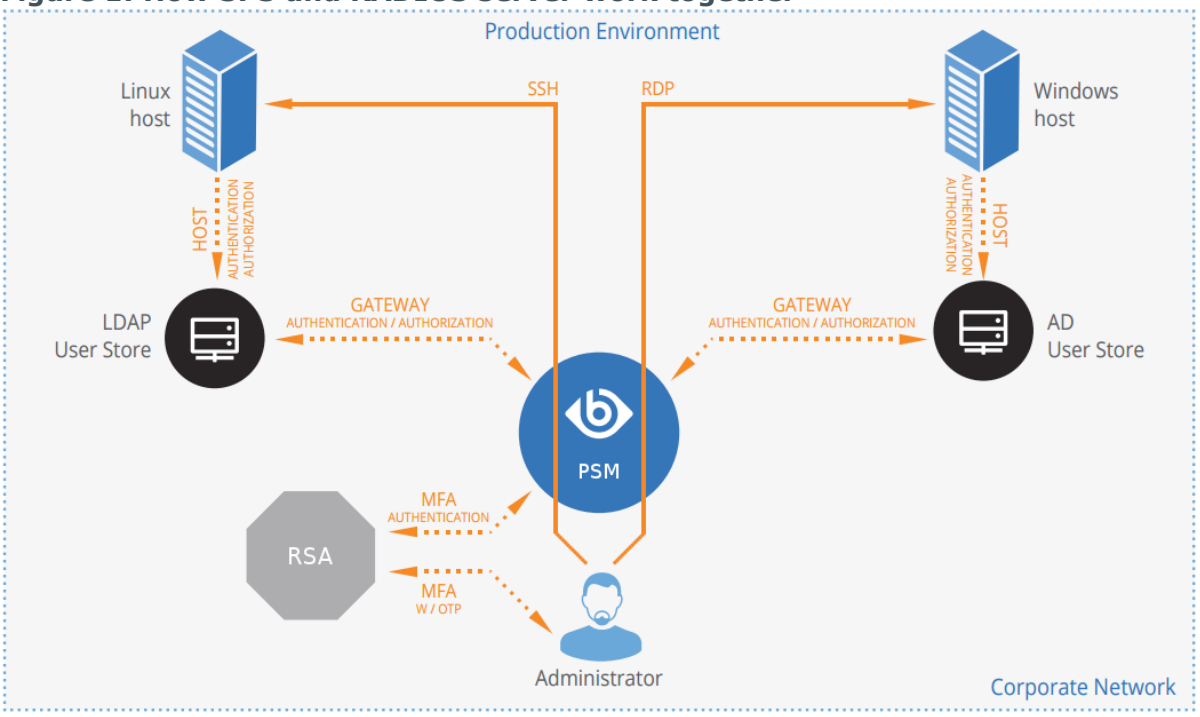
The SPS RADIUS (RSA) Multi-Factor Authentication plugin is available for download as-is, free of charge to every SPS customer from the RADIUS Multi-Factor Authentication plugin for Safeguard for Privileged Sessions page. In case you need any customizations or additional features, contact our Support 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.

How SPS and RADIUS server work together in detail

Figure 1: How SPS and RADIUS server 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 22.

4. **Determining the external RADIUS server identity**

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

5. **Authentication using a RADIUS server**

   If gateway authentication is successful, SPS connects to the RADIUS server. Then SPS requests the second authentication factor from the user and sends it to the RADIUS server for verification.

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.)

   Alternatively, the RADIUS server can request Access-Challenge response. The challenge is displayed to the user and they have to respond to the challenge. After a successful response, the user can start working. In RSA SecurID, this process is used for next token mode. This means that if the password is entered incorrectly several times, two subsequent RSA SecurID tokens have to be entered for a successful authentication.

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 21.

4

Notable features

This section contains the notable features of this plugin.

- To map the gateway usernames to the external RADIUS server identities if the gateway usernames are different from the RADIUS server usernames, configure the [USERMAPPING] on page 26 section of the plugin.
- The [WHITELIST] on page 22 section allows configuring authentication whitelists and blacklists for example to create break-glass access for specific users to allow them to bypass RADIUS server authentication.
- The [authentication_cache] on page 21 section contains the settings that determine how soon after performing a RADIUS server authentication must the user repeat the authentication when opening a new session.
- The [connection_limit by=client_ip_gateway_user] on page 20 section contains the options related to limiting parallel sessions.
- This plugin supports the RADIUS Access-Challenge response and therefore the RSA SecurID next token mode. For details, see How SPS and RADIUS server work together in detail on page 9.

Configure your RSA account for SPS

Prerequisites:

- Administrator access to your RSA account.
- Make sure that you have all the required components listed in Technical requirements on page 7.

1. **Add users to your RSA account.**

   The users you want to authenticate with SPS must have an activated account in RSA. For details on adding or importing your users, see *Integrating LDAP Directories* in RSA Authentication Manager Administrator's Guide in the RSA documentation.

2. **Enable Multi-factor Authentication (MFA) for your organization.**

   Optionally, you can create a Multi-factor Policy in RSA to enable MFA only for the group of users who you want to authenticate with SPS.

   For details, see *Policy Enforcement* in RSA Authentication Manager Administrator's Guide in the RSA documentation.

3. **Retrieve the RADIUS access parameters.**

   RADIUS access parameters, (for example, host, port, and an RSA shared secret).

Configure SPS to use RADIUS multi-factor authentication

Prerequisites:

- Your RADIUS secret and other RADIUS server parameters that are required for configuration.
- Administrator access to SPS.
- Make sure that you have all the required components listed in Technical requirements on page 7.

To configure SPS to use RADIUS multi-factor authentication

1. **Download the SPS RADIUS (RSA) Multi-Factor Authentication plugin**

   SPS customers can download the official plugin from GitHub.

2. **Upload the plugin to SPS**

   Upload the plugin to SPS. For details, see the "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 this configuration file on the **Policies > AA Plugin Configurations** page of the SPS web interface.

   a. Configure the usermapping settings if needed. SPS must find out which RSA user belongs to the username of the authenticated connection. For that, it can query your LDAP/Microsoft Active Directory server. For details, see [USERMAPPING] on page 26.

   b. Configure other parameters of your plugin as needed for your environment. For details, see SPS RADIUS plugin parameter reference on page 15.

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 RADIUS 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 RADIUS plugin in terminal connections and Perform multi-factor authentication with the SPS RADIUS plugin in Remote Desktop (RDP) connections.

SPS RADIUS plugin parameter reference

This section describes the available options of the SPS RSA 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.

Sample configuration file

```
[radius]
server=<radius-server-ip-or-hostname>
port=1812
secret=$
auth_type=pap
conn_retries=3
conn_timeout=5

[auth]
prompt=Press Enter for push notification or type one-time password:
disable_echo=no

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

```
[authentication_cache]
hard_timeout=90
soft_timeout=15
reuse_limit=0

######[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]
prompt=<prompt-to-show-to-the-user>
key=<target-key-for-the-answer>
disable_echo=yes
```

[radius]

This section contains the options related to your RADIUS (RSA) connectivity.

Declaration

```
[radius]
server=<radius-server-ip-or-hostname>
port=1812
secret=<$-or-shared-secret-with-radius-server>
auth_type=pap
conn_retries=3
conn_timeout=5
```

server

Type:	string
Required:	yes
Default:	N/A

Description: The name of your server where the RADIUS interface is available. Enter either the IP address or the hostname.

secret

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 RADIUS shared secret. SPS uses this to communicate with the RADIUS server. For details on using a local Credential Store to host this data, read Store sensitive plugin data securely.

port

Type:	integer
Required:	no
Default:	1812

Description: The port where the RADIUS server is listening for access requests.

auth_type

Type:	string (`chap` | `pap`)
Required:	no
Default:	`pap`

Description: RADIUS authentication type.

- `chap`: CHAP (Challenge-Handshake Authentication Protocol) is a more secure authentication scheme than PAP. In a CHAP scheme, the following process establishes a user identity:
  1. After the link between the user machine and the authenticating server is established, the server sends a challenge message to the connection requester. The requester responds with a value obtained by using a one-way hash function.
  2. The server checks the response by comparing it against its own calculation of the expected hash value.
  3. If the values match, the authentication is acknowledged, otherwise the connection is terminated.

At any time, the server can request the connected party to send a new challenge message. CHAP identifiers are changed frequently and the server can make an authentication request at any time.

- `pap`: The Password Authentication Protocol (PAP) provides a simple method for a user to authenticate using a two-way handshake. PAP only executes this process when establishing the initial link to the authenticating server. A user machine repeatedly sends an ID/Password pair to the authenticating server until authentication is acknowledged or the connection is terminated.

  Use PAP authentication where a plain text password must be available to simulate a login at a remote host. This method provides a similar level of security to the usual user login at the remote host.

conn_timeout

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

Description: Number of seconds to wait for an answer at each retry.

conn_retries

Type:	integer
Required:	no
Default:	3

Description: Number of times to retry sending a RADIUS request if the communication fails.

[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

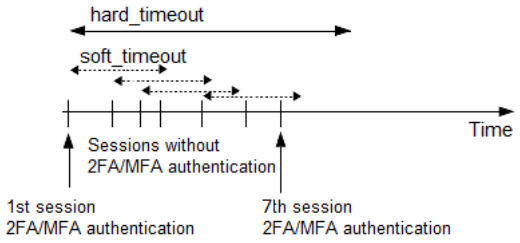
Description: To limit the number of parallel sessions the gateway user can start from a given client IP address, configure `limit`. For an unlimited number of sessions, type `0`.

[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 RADIUS server authentication of the user, SPS will not request a new RADIUS server 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 RADIUS server login of the user), SPS will request a new RADIUS server authentication.

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



Declaration

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

soft_timeout

Type:	integer [in seconds]
Required:	yes, if you want caching
Default:	N/A
Min value:	`0`
Max value:	2147483647

Description: The time in seconds after which the SPS plugin requires a new RADIUS server 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
Min value:	`0`
Max value:	`2147483647`

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

reuse_limit

Type:	integer [number of]
Required:	Optional
Default:	`0`
Min value:	`0`
Max value:	`2147483647`

Description: The number of times that you can reuse the authentication cache before the SPS plugin requires from you a new authentication for the next session. The default is `0`, which means that the authentication cache is not unlimited, but it is turned off.

In the example, if `reuse_limit` is set to `5`, and you successfully authenticated with multi-factor authentication, the next `5` authentications are bypassed in the next `90` seconds (`hard_timeout`), if there is no gap bigger than `15` seconds (`soft_timeout`) between the authentications.

If any of the `hard_timeout`, `soft_timeout`, or `reuse_limit` parameters, which operate independently from one another, exceed the configured limit, the SPS plugin requires you to authenticate for the new session.

[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 RADIUS server 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 RADIUS server 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 RADIUS server 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 RADIUS server 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 RADIUS server 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 RADIUS server 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 RADIUS server 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 RADIUS server identities, you must configure the SPS RADIUS plugin to map the gateway usernames to the external RADIUS server identities.

You can use the following methods:

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

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

  If the RADIUS server service requires the use of domain name in the external RADIUS server 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 RADIUS 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 RADIUS server server.

  If you configure both the `append_domain` parameter in the `[username_transform]` section and the `[usermapping source=ldap_server]` section of the SPS RADIUS 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 RADIUS server identity of the user is the same as the gateway username.

[usermapping source=explicit]

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

- Type the gateway user name instead of `<example-user-1>`.
- Type the external RADIUS server 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 RADIUS server identity of the user from an LDAP/Active Directory database, configure the `[usermapping source=ldap_server]` section of the SPS RADIUS 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 RADIUS 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 RADIUS server service requires the use of domain name in the external RADIUS server 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 RADIUS 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 RADIUS server server.

If you configure both the `append_domain` parameter in the `[username_transform]` section and the `[usermapping source=ldap_server]` section of the SPS RADIUS 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 27 and [whitelist source=ldap_server_group] on page 24.

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 `secret` value in the `[radius]` 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, `secret`) 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, `radius`).
   - Enter the name of the plugin parameter in the **Username** field (for example, `secret`).
   - 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 RADIUS 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.

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 RADIUS 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.

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 RADIUS 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.

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