One Identity Safeguard for Privileged Sessions 5.7

RSA Multi-Factor Authentication - Tutorial

Legend

WARNING: A WARNING icon indicates a potential for property damage, personal injury, or death.

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

IMPORTANT, NOTE, TIP, MOBILE, or VIDEO: An information icon indicates supporting information.

SPS RSA Multi-Factor Authentication - Tutorial
Updated - July 2018
Version - 5.7

Contents

3

Introduction

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

One Identity Safeguard for Privileged Sessions:

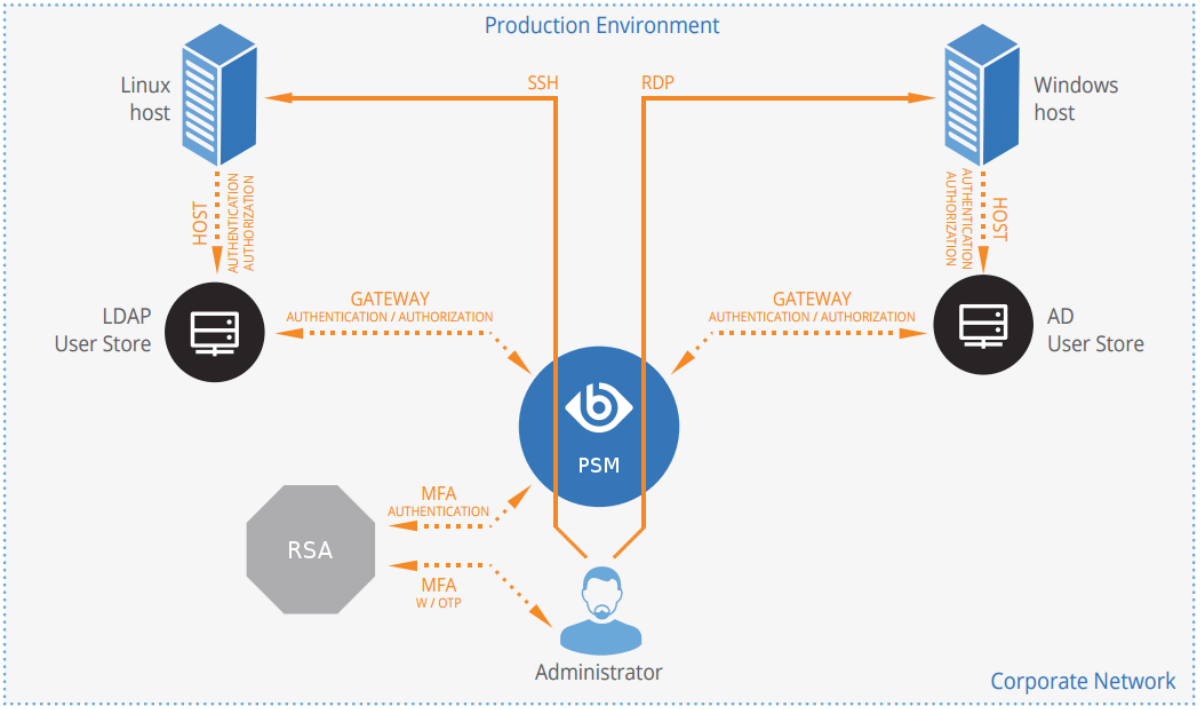
One Identity Safeguard for Privileged Sessions (Safeguard for Privileged Sessions) controls privileged access to remote IT systems, records activities in searchable, movie-like audit trails, and prevents malicious actions. Safeguard for Privileged Sessions is a quickly deployable enterprise device, completely 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.

RSA Adaptive Multi-factor Authentication:

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

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

Figure 1: How Safeguard for Privileged Sessions and RSA work together



Solution benefits

Using Safeguard for Privileged Sessions together with RSA provides the following benefits:

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

Meet compliance requirements

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

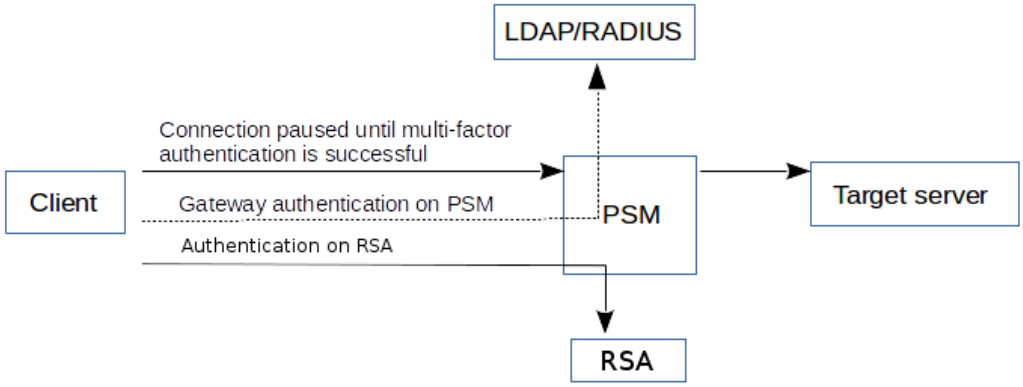
other requirements, using Safeguard for Privileged Sessions and RSA helps you comply with the following requirements:

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

Safeguard for Privileged Sessions acts as a central authentication gateway, enforcing strong authentication before users access sensitive IT assets. Safeguard for Privileged Sessions 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 Safeguard for Privileged Sessions'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 RSA (or another multi-factor authentication provider), Safeguard for Privileged Sessions directs all connections to the authentication tool, and upon successful authentication, it permits the user to access the information system.

How Safeguard for Privileged Sessions and RSA MFA work together

Figure 2: How Safeguard for Privileged Sessions and RSA work together



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

2. **Gateway authentication on Safeguard for Privileged Sessions**

   Safeguard for Privileged Sessions receives the connection request and authenticates the user. Safeguard for Privileged Sessions 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. **Authentication using RSA SecurID Access**

   If gateway authentication is successful, Safeguard for Privileged Sessions connects the RSA Authentication Manager. Then Safeguard for Privileged Sessions requests the second authentication factor from the user and sends it to the RSA server for verification.

4. If multi-factor authentication is successful, the user can start working, while Safeguard for Privileged Sessions records the user's activities. (Optionally, Safeguard for Privileged Sessions can retrieve credentials from a local or external

credential store or password vault, and perform authentication on the server with credentials that are not known to the user.)

Technical requirements

In order to successfully connect Safeguard for Privileged Sessions with RSA, 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 Safeguard for Privileged Sessions plugin.
- Your users must be enrolled in RSA Authentication Manager.
- The users must be able to perform the authentication required for the fasctor (for example, possess the required RSA SecurID Hardware Token).

In Safeguard for Privileged Sessions:

- A One Identity Safeguard for Privileged Sessions appliance (virtual or physical), at least version 5 F1.
- A copy of the Safeguard for Privileged Sessions RSA plugin. This plugin is an Authentication and Authorization (AA) plugin customized to work with the RSA multi-factor authentication service.
- Safeguard for Privileged Sessions must be able to access the RADIUS port of the RSA Authentication Manager.
- Safeguard for Privileged Sessions supports Authentication and Authorization 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 Safeguard for Privileged Sessions RSA plugin is available as-is, free of charge to every Safeguard for Privileged Sessions customer from the Plugin Page. In case you need any customizations or additional features, contact professionalservices@balabit.com.

You can use the plugin on Safeguard for Privileged Sessions 5 F5 and later. If you need to use the plugin on Safeguard for Privileged Sessions 5 LTS, contact professionalservices@balabit.com.

6

How Safeguard for Privileged Sessions and RSA work together in detail

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

2. **Gateway authentication on Safeguard for Privileged Sessions**

   Safeguard for Privileged Sessions receives the connection request and authenticates the user. Safeguard for Privileged Sessions 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 Safeguard for Privileged Sessions 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 Safeguard for Privileged Sessions records the user's activities. The procedure ends here.
   - If multi-factor authentication is required, Safeguard for Privileged Sessions continues the procedure with the next step.

   For details on creating exemption lists, see Safeguard for Privileged Sessions RSA plugin parameter reference.

4. **Determining the RSA username**

   If the gateway usernames are different from the RSA usernames, you must configure the Safeguard for Privileged Sessions RSA plugin to map the gateway usernames to the RSA usernames. The mapping can be as simple as appending a domain name to the gateway username, or you can query an LDAP or Microsoft Active Directory server. For details, see Mapping Safeguard for Privileged Sessions usernames to RSA identities.

5. **Authentication using RSA SecurID Access**

   If gateway authentication is successful, Safeguard for Privileged Sessions connects the RSA Authentication Manager. Then Safeguard for Privileged Sessions requests the second authentication factor from the user and sends it to the RSA server for verification.

6. If multi-factor authentication is successful, the user can start working, while Safeguard for Privileged Sessions records the user's activities. (Optionally, Safeguard for Privileged Sessions 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 Safeguard for Privileged Sessions RSA plugin parameter reference.

7

Mapping Safeguard for Privileged Sessions usernames to RSA identities

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

If the gateway usernames are different from the RSA usernames, you must configure the Safeguard for Privileged Sessions RSA plugin to map the gateway usernames to the RSA usernames. You can use the following methods:

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

Bypassing RSA authentication

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

- In Safeguard for Privileged Sessions, 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 Safeguard for Privileged Sessions RSA plugin parameter reference.
- You can configure Safeguard for Privileged Sessions using whitelists and blacklists to selectively require multi-factor authentication for your users, for example, to create break-glass access for specific users. For details on creating exemption lists, see Safeguard for Privileged Sessions RSA plugin parameter reference.

Configure your RSA account for Safeguard for Privileged Sessions

Prerequisites:

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

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

   The users you want to authenticate with Safeguard for Privileged Sessions 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 Safeguard for Privileged Sessions.

   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 Safeguard for Privileged Sessions to use RSA multi-factor authentication

Prerequisites:

- Your RSA API token.

  CAUTION:

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

- Administrator access to Safeguard for Privileged Sessions.
- Make sure that you have all the required components listed in Technical requirements.

Steps:

1. **Download the Safeguard for Privileged Sessions RSA plugin**

   Safeguard for Privileged Sessions customers can download the plugin from Plugin Page.

2. **Upload the plugin to Safeguard for Privileged Sessions**

   Upload the plugin to Safeguard for Privileged Sessions. For details, see Administration Guide.

3. **Configure the plugin on Safeguard for Privileged Sessions**

   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 Safeguard for Privileged Sessions web interface.

a. Configure the usermapping settings if needed. Safeguard for Privileged Sessions 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 Mapping Safeguard for Privileged Sessions usernames to RSA identities.

b. Configure other parameters of your plugin as needed for your environment. For details, see Safeguard for Privileged Sessions RSA plugin parameter reference.

4. ## Configure a Connection policy and test it

Configure a Connection policy on Safeguard for Privileged Sessions. In the **AA plugin** field of the Connection policy, select the Safeguard for Privileged Sessions RSA 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 Safeguard for Privileged Sessions RSA plugin in terminal connections and Perform multi-factor authentication with the Safeguard for Privileged Sessions RSA plugin in Remote Desktop connections.

CAUTION:

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

Safeguard for Privileged Sessions RSA plugin parameter reference

This section describes the available options of the Safeguard for Privileged Sessions 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 Safeguard for Privileged Sessions web interface. The following code snippet is a sample configuration file.

```
[rsa]
server=<radius.example.com>
# Do NOT use secret in production
; secret=<RADIUS-shared-secret>
port=1812
auth_type=chap
timeout=10
retries=3

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

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

```
[username_transform]
append_domain=""

[ldap]
ldap_server_config=<Safeguard for Privileged Sessions-LDAP-server-policy-name>
filter=(&(samAccountName={})(objectClass=user))
user_attribute=mail

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

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

[question_2]...
```

[rsa]

This section contains the options related to your RSA account.

```
[rsa]
server=<radius.example.com>
# Do NOT use secret in production
; secret=<RADIUS-shared-secret>
port=1812
auth_type=chap
timeout=10
retries=3
```

server

Type:	string
Required:	yes
Default:	N/A

Description: The name of your RSA SecurID server, where the RADIUS interface is available.

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. Never use it in production.

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

Only use this parameter in the configuration for testing purposes in a secure, non-production environment.

Description: Your RADIUS shared secret. Safeguard for Privileged Sessions 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:	chap

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.

timeout

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

Description: How long the RADIUS server waits to respond.

retries

Type:	integer
Required:	no
Default:	3

Description: The number of times authentication is retried.

[plugin]

This section contains general plugin-related settings.

```
[plugin]
config_version=1
```

```
log_level=20
cred_store=<name-of-credstore-hosting-sensitive-data>
```

config_version

Type:	integer
Required:	yes
Default:	1

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

cred_store

Type:	string
Required:	no
Default:	N/A

Description: The name of a local credential store policy configured on Safeguard for Privileged Sessions. You can use this credential store to store sensitive information of the plugin in a secure way, for example, the `ikey`/`skey` values in the `[rsa]` section. For details, see Store sensitive plugin data securely.

log_level

Type:	integer or string
Required:	no
Default:	info

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

The possible values are:

- `debug` or `10`
- `info` or `20`
- `warning` or `30`

- `error` or `40`
- `critical` or `50`

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

[auth]

This section contains the options related to authentication.

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

prompt

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

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

```
prompt="Hit Enter to send RSA push notification or provide the OTP:"
```

whitelist

Type:	string
Required:	no
Default:	N/A

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

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

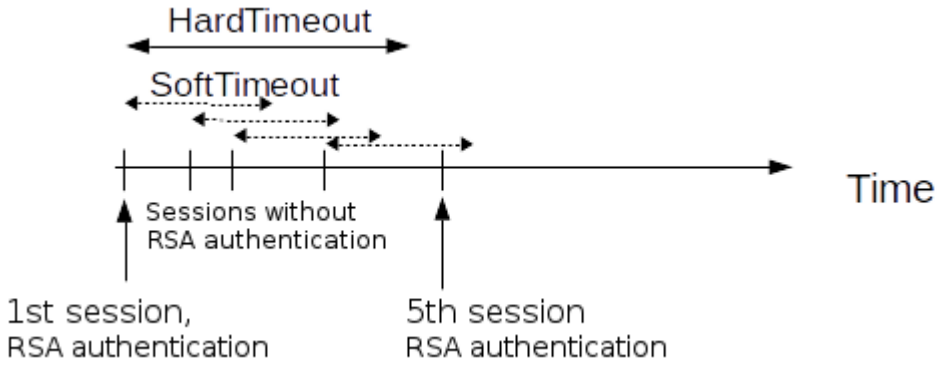
For details on creating user lists, see "Creating and editing user lists" in the Administration Guide.

[cache]

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

After the first RSA authentication of the user, Safeguard for Privileged Sessions will not request a new RSA authentication from the user as long as the new authentications would happen within `soft_timeout` seconds from each other. After the `hard_timeout` expires (measured from the first RSA login of the user), Safeguard for Privileged Sessions will request a new RSA authentication.

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




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

soft_timeout

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

Description: The time in seconds after which the Safeguard for Privileged Sessions plugin requires a new RSA authentication for the next new session of the user, unless the user successfully authenticates another session within this period.

hard_timeout

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

Description: The time in seconds after which the Safeguard for Privileged Sessions plugin requires a new RSA authentication for the next new session of the user. The time is measured from the last RSA 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.

[ldap]

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

To look up the RSA username of the user from an LDAP/Active Directory database, configure the `[ldap]` section of the Safeguard for Privileged Sessions RSA plugin. Typically, the Safeguard for Privileged Sessions plugin queries the email address corresponding to the username from your LDAP or Active Directory database. For details on LDAP parameters, see Safeguard for Privileged Sessions RSA plugin parameter reference.

If you configure both the `append_domain` parameter and the `[ldap]` section of the Safeguard for Privileged Sessions RSA plugin, Safeguard for Privileged Sessions appends the `@` character and the value of the `append_domain` parameter to the value retrieved from the LDAP database.

For other methods of mapping gateway usernames to RSA usernames, see Mapping Safeguard for Privileged Sessions usernames to RSA identities.

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

ldap_server_config

Type:	string
Required:	no
Default:	N/A

Description: The name of a configured LDAP server policy in Safeguard for Privileged Sessions. For details on configuring LDAP policies, see "Authenticating users to an LDAP server" in the Administration Guide.

filter

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

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

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

user_attribute

Type:	string
Required:	no
Default:	cn

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

[username_transform]

This section contains username transformation-related settings.

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

append_domain

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

Description: If the gateway usernames are different from the RSA usernames, you must configure the Safeguard for Privileged Sessions RSA plugin to map the gateway usernames to the RSA usernames.

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

If you configure both the `append_domain` parameter and the `[ldap]` section of the Safeguard for Privileged Sessions RSA plugin, Safeguard for Privileged Sessions appends the `@` character and the value of the `append_domain` parameter to the value retrieved from the LDAP database.

For other methods of mapping gateway usernames to RSA usernames, see Mapping Safeguard for Privileged Sessions usernames to RSA identities.

[question_1]

Type:	integer [seconds]

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

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

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

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

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

key

Type:	string
Required:	yes
Default:	N/A

Description: The name of the name-value pair.

prompt

Type:	string
Required:	yes
Default:	N/A

Description: The question itself in text format.

disable_echo

Type:	boolean yes|no
Required:	no
Default:	no

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

Store sensitive plugin data securely

Purpose:

By default, the configuration of the plugin is stored on Safeguard for Privileged Sessions in the configuration of Safeguard for Privileged Sessions. Make sure that you store the sensitive parameters (for example, secret) of the plugin in an encrypted way. To do this, complete the following procedure.

Steps:

1. Log in to Safeguard for Privileged Sessions and create a local Credential Store. For details, see "Configuring password-protected 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, `rsa`).
   - Enter the name of the plugin parameter in the**USERNAME** field field (for example, `secret`).
   - Enter the value of the plugin parameter in the**PASSWORD** field.
3. Commit your changes, and navigate to the configuration of the plugin on the **Policies > AA Plugin Configurations** page.
4. In the plugin configuration file, enter the name of the local Credential Store under the `[plugin]` section, in the cred_store parameter.

Perform multi-factor authentication with the Safeguard for Privileged Sessions RSA plugin in terminal connections

Purpose:

To establish a terminal connection (SSH, TELNET, or TN3270) to a server, complete the following steps.

Steps:

1. Connect to the server.

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

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

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

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

   - Google Authenticator: `g`
   - inWebo Authenticator: `o`
   - Symantec token: `s`
   - YubiKey: `y`
   - RSA token: `r`
2. If Safeguard for Privileged Sessions 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 Safeguard for Privileged Sessions RSA plugin in Remote Desktop connections

Purpose:

To establish a Remote Desktop (RDP) connection to a server when the **AA plugin** is configured, complete the following steps.

Steps:

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

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

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

   For example, use the following format:

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

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

   - Google Authenticator: `g`
   - inWebo Authenticator: `o`
   - Symantec token: `s`

- YubiKey: `y`
- RSA token: `r`

3. Connect to the server.

   If you need to authenticate using the RSA Authenticator push notification, approve the connection in your mobile app.
4. Authenticate on the server.
5. If authentication is successful, you can access the server.

15

Learn more

To find out more about Safeguard for Privileged Sessions, visit the One Identity page.

If you need help in connecting your RSA account with One Identity Safeguard for Privileged Sessions, contact our Sales Team or contact professionalservices@balabit.com.

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 or other inquiries, visit https://www.oneidentity.com/company/contact-us.aspx or call +1-800-306-9329.

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