Privileged Account Manager
Use Case Guide
January 2023
Legal Notice

For information about Micro Focus legal notices, see https://www.microfocus.com/about/legal.

© Copyright 2023. Micro Focus or one of its affiliates.
<table>
<thead>
<tr>
<th></th>
<th>Contents</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Securing a Web-Based RDP Session</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>Examining the Elevated SSH Session without the Thick Client</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Provisioning Just-In-Time Access to AWS Services Using IAM Roles</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>Configuring Time Bound Just-In-Time Privileges</td>
<td>19</td>
</tr>
</tbody>
</table>
A use case describes how a person who actually uses a process or system will accomplish a goal. A use case also helps you understand the errors that could occur in the process and design features to resolve those errors.

You can use Privileged Account Manager to impose control over elevated access and permissions for users, accounts, processes, and systems throughout an IT environment by using the sample scenarios provided in this guide.

For more information about this software, see the Privileged Account Manager documentation.
Securing a Web-Based RDP Session

Secure and monitor privilege RDP access, without the need of agent or client application on the target machine.

Let us assume, an organization Intech Data, Inc. wants to secure and monitor employees’ action during the privileged access to Windows Server that is located in the network lab. This Windows Server contains confidential data.

Employees can access this server without using the remote desktop client. A few employees can access it remotely only during the business hours (9 AM to 6 PM) on business days (Monday to Friday).

This example refers to the following user profiles:

- Markus: An database administrator of Intech Data, Inc.
- Albert Jones: An employee of Intech Data, Inc.

Markus wants to enforce **Windows Agentless Web RDP** to secure the Windows server. With this implementation, designated employees can access the Windows server through PAM User console for privileged access, and each action is recorded and monitored for auditing purposes. If employees perform any action that poses potential risk, such as deleting, copying data, and so on, the actions are recorded. Markus can suspend the just-in-time session if he is monitoring the active session.

The following diagram depicts the requirement of this use case:

![Diagram of RDP session security](image)

This section explains the prerequisites, flow of actions, and step-by-step configuration details to achieve this requirement.
Securing a Web-Based RDP Session

Flow Diagram

The following diagram illustrates the sequential flow of configuration required for implementing the Windows Agentless Web RDP:

1. **Add Active Directory Services as the LDAP Server**
   - Administrator (Markus)

2. **Create resource for Windows Server in Credential Vault**

3. **Define a User Role for the LDAP users or group**

4. **Create a Resource pool with Windows vault for Agentless Windows Servers**

5. **Create an Assignment for RDP Web and grant required permission**

6. **Administration Console > Reports**

7. **Windows Server**

8. **Privileged Access via User Console**

   - End-user (Albert Jones)

   - Recorded actions and video capture are stored in Reports

Prerequisites

- Privileged Account Manager is installed with the required Agentless module.

**NOTE:** The agentless component of Privileged Account Manager (agentless) is supported on SLES 12 (64-bit), SLES 15 (64-bit), Oracle Linux 8 (64-bit), and RHEL 8 (64-bit).

- Enable the Access Control Engine option in Server Settings > Policy Engine.
- Import the CA certificate on the system where Privileged Account Manager is installed to enable communication over Secure Sockets Layer (SSL).
Follow the below mentioned procedure:

1. Configure the LDAP Repository
2. Add a Resource
3. Define a User Role
4. Create a Resource Pool
5. Create an Assignment and Grant Permission

After Markus implements **Windows Agentless Web RDP**, the following are possible scenarios:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albert logs in to the User Console and accesses the Windows Server. Later, he installs software and configures it.</td>
<td>Each action is recorded, the live session video is captured, and stored in Reports. No malicious activity is detected.</td>
</tr>
<tr>
<td>Albert wants to access Windows Server on Thursday at 8 PM.</td>
<td>The Windows Server resource is not available on the User Console for Albert. As Windows Server is accessible from 9 AM to 6 PM.</td>
</tr>
<tr>
<td>Albert installs a malware on Windows Server unintentionally.</td>
<td>The install procedure is recorded. When Markus checks the report and notices malware installation, he notifies Albert to uninstall the identified malware.</td>
</tr>
<tr>
<td>Markus is monitoring the live session and Albert attempts to delete the server configuration.</td>
<td>Markus terminates the session. Later, he drafts a mail with the reason and sends it to Albert.</td>
</tr>
</tbody>
</table>
3

Examining the Elevated SSH Session without the Thick Client

Live Monitoring and Mitigation

Let us assume an organization, Intech Data, Inc. wants to assess the user actions during the elevated session to the Linux system that functions as the Web Server of the organization. The Linux system is accessed without the thick clients, such as PuTTY, MobaXterm, and so on. In the elevated session, an employee can access Linux system with the root permissions.

This example refers to the following user profiles:

- Markus: An administrator of Intech Data, Inc.
- Silvia Pereria: An employee of Intech Data, Inc.

Markus wants to implement **Linux Agentless Web SSH** to secure the elevated session from internal misuse. With this implementation, authorized employees can access Linux system through the User Console for elevated access and each Keystore is captured and recorded for auditing purposes. Also, Markus has the authority to inspect the live session and perform the following upon noticing any suspicious activity, such as changing the system configuration, installing an unauthorized application, and so on:

- Terminate the active session and send a notice to the user
- Block access to the elevated session

The following diagram depicts the requirement of this use case:

This section explains the prerequisites, flow of actions, and step-by-step configuration details to achieve this requirement.
Flow Diagram

The following diagram illustrates the sequential flow of configuration required for implementing the Linux Agentless Web SSH:

To achieve this requirement, Markus performs the following tasks:

1. Configure the LDAP Repository
2. Add a Resource
3. Defining a User Role
4. Create a Resource Pool
5. Create an Assignment and Grant Permission
Prerequisites

- The Privileged Account Manager is installed with the required Agentless module.

**NOTE:** The agentless component of Privileged Account Manager (agentless) is supported only on SLES 12 (64-bit), SLES 15 (64-bit), Oracle Linux 8 (64-bit), and RHEL 8(64-bit).

- Ensure to enable the Access Control Engine option in Server Settings > Policy Engine.
- Import the CA certificate on the system where you have installed Privileged Account Manager to enable communication over Secure Sockets Layer (SSL).

Follow the below mentioned procedure:

1. Add LDAP Repository
2. Add Linux System as a Resource
3. Define a User Role
4. Create a Resource Pool
5. Create an Assignment and Grant Permission

After Markus implements **Linux Agentless Web SSH**, the following are possible scenarios:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silvia logs in to the User Console and accesses the elevated session on Linux. Later, she executes some commands.</td>
<td>Each command is recorded and the live session video is captured. No malicious activity is detected.</td>
</tr>
<tr>
<td>Silvia installs malware on Linux SSH unintentionally.</td>
<td>The install procedure is recorded. When Markus goes through the recorded session and notices malware installation, he notifies Silvia to uninstall the identified malware.</td>
</tr>
<tr>
<td>Markus is monitoring the live session and Silvia attempts to modify the security controls.</td>
<td>Markus terminates the session and blocks access to Linux SSH. Later, he drafts a mail with the reason and sends it to Silvia.</td>
</tr>
<tr>
<td>Silvia deletes a configuration file.</td>
<td>The action is recorded and saved in the <strong>Reports</strong>. Markus can take relevant actions while auditing the report.</td>
</tr>
</tbody>
</table>
Provisioning Just-In-Time Access to AWS Services Using IAM Roles

Controlling granular privileged action on AWS through IAM roles by providing ephemeral account.

This is a sample scenario featuring Acme Corporation. This firm are in the process of migrating their applications and workload to SaaS platform name AWS.

This use case highlights the use of Privileged Account Manager in the following circumstances:

- Delegated resource administration in vault
- Secret level permissions in vault and ability to directly read secrets from Vault
- Just in time (JIT) access with zero standing privileges
- Support for AWS privilege access using ephemeral account
- Support for PKI, where Privileged Account Manager acts as a Certificate issuer

This example refers to the following user profiles:

- Fiona: An application administrator of Acme Corporation. Fiona needs occasional privileged access on AWS (managed by Steve) to perform application maintenance and configuration.
- Adam: An administrator of Privileged Account Manager at Acme Corporation.
- Aravanti: An application owner who manages and provides secure, on-demand access for application using secure access methods such as, PKI.

This section explains the prerequisites, flow of actions, and step-by-step configuration details to achieve this requirement.
Flow Diagram

The following diagram illustrates the sequential flow of configuration required for implementing the Just in time access framework:

1. Adam configures the Geo Resources with different Resource Administrators
2. Adam configures Steve and Aravanti as delegated administrators for respective resource paths
3. Steve and Aravanti create and manage resources in respective paths
4. Steve and Aravanti configure and create different JIT templates for various privileged access levels
5. Steve and Aravanti can configure new users who require on-demand and short-lived privileged access

Just-in-Time access using read secret

Privileged Account Manager Console

End-user

End-user
To achieve this requirement, Steve performs the following tasks:

After implementation of **Just-in-Time access**, the following are possible scenarios:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator can log in and review. If required, can disconnect a session or revoke AWS secret for risky logins and audit logs.</td>
<td>Each action is recorded, the live session video is captured, and stored in <strong>Reports</strong>. No malicious activity can go undetected.</td>
</tr>
</tbody>
</table>
SSH certificate based authentication comes with the following benefits:

- There is a specific validity period.
- The new certificates can be provided with ease without the overhead of modifying the authorized keys in the target server.
- You can customize the certificate to ensure the certificates are used only for a specific purpose and with least privileges to avoid misuse.

Let us consider a scenario where John, a DevOps engineer wants to access the target production server. The Certificate Authority in Privileged Account Manager, could either be imported or generated by Privileged Account Manager and will be used to sign the end user certificate. This Certificate Authority is protected by Privileged Account Manager's vault encryption capability.

For a new ephemeral certificate request, John will be evaluated by the Credential Vault's policy engine which will ensure the right access is provided to the resource. If he is validated by the permissions provided, he can access the certificate signed my CA. The Public key configured int he target server and will be used for validation.

This use case can also be used for devices - zero trust privileged config for network devices.

This example refers to the following user profiles:

- John: A DevOps engineer of Pristine Inc.

John wants to access the production server for collection performance metrics. He leverages SSH Certificates for Just-in-Time access to the server.

The following diagram depicts the requirement of this use case:
This section explains the prerequisites, flow of actions, and step-by-step configuration details to achieve this requirement.

To achieve this requirement, John performs the following tasks:

1. Create a Certificate Authority either by importing from or create using Privileged Account Manager
2. You receive the public key and configure the same in the target server
3. Restart SSHD
4. Configure Certificate that is required for the temporary certificate
5. Read the credential
6. Log in as John into Privileged Account Manager into the production server and using the new certificate to obtain the required performance metrics
7. Connection is closed by the target server as the action is complete
8. Monitor the logs for further details
After John implements **SSH Certificate based Authentication**, the following are possible scenarios:

<table>
<thead>
<tr>
<th>Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSHD, No revocation possible, SSH Certificate options can be more scenarios (through host and command restrictions)</td>
</tr>
<tr>
<td>Through host and command restrictions</td>
</tr>
</tbody>
</table>