Tenable and Microsoft Azure Key Vault Integration Guide

Last Revised: August 25, 2026

Table of Contents

Welcome to Tenable for Microsoft Azure Key Vault

This document provides information and steps for integrating Tenable One Vulnerability Management, or Tenable Nessus with Microsoft Azure Key Vault. For more information, refer to the following product documentation:

- Tenable One Vulnerability Management
- Tenable Nessus

Overview

The Tenable integration with Microsoft Azure Key Vault delivers a credentialed-scanning solution that lets security teams store privileged secrets in Azure Key Vault and retrieve them automatically at scan time. Sensitive passwords and SSH private keys remain centrally managed in the vault — they are never stored in the Tenable scan policy and are rotated without any manual policy update.

You can integrate Microsoft Azure Key Vault with Tenable One Vulnerability Management, or Tenable Nessus to perform credentialed scanning of SSH, Windows (SMB), and Database targets, or use them in conjunction with the VMware vCenter API, VMware ESXi SOAP API, and Nutanix Prism Central credential types.

The benefits of integrating Tenable with Microsoft Azure Key Vault include:

- Centralized management of privileged credentials for Windows, SSH, Database, VMware, and Nutanix targets.
- OAuth2 (client-credentials) authentication using a Microsoft Entra ID service principal, so no long-lived target passwords or SSH keys need to be stored in Tenable.
- Support for both password- and SSH private-key-based target authentication via a single Key Vault secret.

What information does the Microsoft Azure Key Vault integration collect?

The Azure Key Vault Privileged Access Management (PAM) integration collects only the credential material required to authenticate to a scan target. The value of the Key Vault secret is parsed as a JSON object; the following fields are read from the secret when present:

- username — the target account username.
- password — the target account password.
- ssh_key — a PEM-encoded SSH private key used to authenticate to SSH targets.
- ssh_keyphrase — the passphrase protecting the SSH private key (mapped internally to passphrase).
- domain — the Windows/Active Directory domain associated with the account (used only when Fetch Domain is enabled).

> **Note**: password vs. ssh_key — If ssh_key is present in the secret, it is used as the SSH private key. If ssh_key is absent but the password value contains PEM markers (the literal text -----BEGIN or PRIVATE KEY), the integration treats the password field as an SSH private key. In every other case the password field is used as a plain password. The ssh_keyphrase field is only applied when one of those two SSH-key paths is taken.

If the Key Vault secret value is plain text rather than JSON, the entire string is treated as the password.

API requests per target

For each scan, the integration first performs one OAuth2 token request against https://login.microsoftonline.com/<tenant_id>/oauth2/v2.0/token to obtain a bearer access token. The access token is cached in-memory for the duration of the plugin execution and is reused for all subsequent secret retrievals. One additional GET request against https://<key_vault_name>.vault.azure.net/secrets/<secret_name>?api-version=7.4 is made for each distinct Key Vault

secret referenced by the scan policy. If SSH privilege escalation is configured with an independent Escalation Credential ID, one extra GET request is issued to fetch that escalation secret.

Retrieved credentials are cached for the duration of the scan (keyed by secret name) to avoid repeated requests for the same secret. The cache is not shared between scan chunks, so each chunk will produce a minimum of one authentication request and one secret request.

Credentialed Scans

Credentialed scans allow the Tenable scanner to log in to the target system directly and perform a deeper assessment than is possible without credentials. This includes checking installed software versions, configuration settings, patch levels, and compliance status.

What the Microsoft Azure Key Vault integration does not collect

- The integration does not collect vulnerability data, software inventory, device inventory, or configuration details from target systems — it only retrieves credentials that Tenable then uses to authenticate to those targets.
- The integration does not read Azure Key Vault Keys or Certificates objects. The API returns only the public component of a private key, and the Certificates object stores its private key separately in the Secrets object using an X.509 PEM format that is not directly usable for standard OpenSSH authentication. Only the Secrets object is supported.
- Managed identities are not used as an authentication method — the integration authenticates as a Microsoft Entra ID service principal using the OAuth2 client-credentials grant.

> **Note**: For compliance audits, add audit files to your scan policy. The Azure Key Vault integration provides credentials, while the audit files define the checks performed.

Microsoft Azure Key Vault Integration limitations

- The Azure Key Vault endpoint (<key_vault_name>.vault.azure.net) and the Microsoft Entra ID token endpoint (login.microsoftonline.com) must be reachable over HTTPS (TCP 443) from every Tenable Nessus scanner used in the scan. If the Key Vault firewall is enabled, the scanner's public egress IP must be on the allow-list.
- Only the Secrets object type is supported. The Keys and Certificates objects are out of scope for this integration (see the previous section for the reason).
- Only OAuth2 client-credentials authentication is supported. The service principal must have the Key Vault Secrets User role (or a role that includes the Microsoft.KeyVault/vaults/secrets/getSecret/action data action) on the vault.
- There is no auto-discovery or dynamic-scanning variant for this integration. Each Tenable scan credential resolves a fixed Key Vault secret name; targets must be enumerated explicitly in the scan.
- Azure Key Vault imposes a service limit of approximately 4,000 secret GET operations per 10 seconds per vault / region. Scans of very large fleets that all pull from the same vault should account for this ceiling.
- Microsoft Entra ID access tokens issued for the OAuth2 client-credentials flow have a lifetime randomized between 60 and 90 minutes. The integration automatically requests a new token when the cached one expires.

Configure Microsoft Azure Key Vault integration for Tenable One Vulnerability Management

The Tenable Azure Key Vault integration exists as an authentication method within the SSH, Windows, Database, VMware ESXi SOAP API, VMware vCenter API, and Nutanix credential types in Tenable One Vulnerability Management scan policies.

Required permissions

- A Tenable One Vulnerability Management user account with a minimum role of Standard.
- A Microsoft Entra ID App registration (service principal) in the tenant that owns the Key Vault. The Application (client) ID and a Client secret from that App registration are required.
- The service principal must be granted the Key Vault Secrets User built-in role (or a custom role with the Microsoft.KeyVault/vaults/secrets/getSecret/action data action) on the target Key Vault, either through Azure RBAC or the vault's access policies. Granting the role at the individual vault scope is sufficient; subscription-wide access is not required.
- Every credential stored in the vault must be a JSON object whose keys use the field names described earlier (username, password, ssh_key, ssh_keyphrase, domain). Plain-text secrets are supported and are treated as a password only.

Note: For compliance audits, add audit files to your scan policy. The Azure Key Vault integration provides credentials only.

Scan configuration — Windows, SSH, Database, VMware, and Nutanix

Complete the following steps to configure Tenable One Vulnerability Management with the Azure Key Vault integration using the Windows, SSH, or Database credential types.

1. Log in to your Tenable One Vulnerability Management user interface.
2. In the left navigation pane, click **Scans**. The Scans page appears.
3. In the upper-right corner of the page, click the **Create a Scan** button. The Select a Scan Template page appears.
4. Select a scan template. The scan configuration page appears.
5. In the **Name** box, type a name for the scan.
6. In the **Targets** box, type an IP address, hostname, or range of IP addresses.
7. (Optional) Add a description, folder location, scanner location, and specify target groups.
8. Click the **Credentials** tab. The Credentials pane appears.
9. In the **Select a Credential** menu, select Windows, SSH, Database, VMware ESXi SOAP API, VMware vCenter API, or Nutanix from the options. The Settings pane for the selected credential appears.
10. In the **Auth Type** drop-down box, click **Azure Key Vault**. The Azure Key Vault options appear.
11. Configure each option for the selected Windows, SSH, Database, VMware, or Nutanix credential using the values in the table below.

Option	Description	Required
Username	The username to log in to the target system(s). Optional if a username field is present in the Azure Key Vault secret JSON; when supplied here it overrides the value stored in the secret.	No
Tenant ID	The Microsoft Entra ID (Azure AD) Directory (tenant) ID that owns the Key Vault. Used to build the OAuth2 token	Yes

	endpoint https://login.microsoftonline.com/<tenant_id>/oauth2/v2.0/token.	
Application ID	The Application (client) ID of the Microsoft Entra ID service principal registered under App registrations. Also referred to as the Client ID.	Yes
Client Secret	The client secret value generated for the service principal under Certificates & secrets. Used together with the Application ID and Tenant ID for the OAuth2 client_credentials grant.	Yes
Key Vault Name	The name of the Azure Key Vault (not the full URL). Tenable derives the API host from this value as <Key Vault Name>.vault.azure.net.	Yes
Secret Name	The name of the Key Vault secret that stores the target credentials. The secret value is expected to be a JSON object containing one or more of the fields: username, password, ssh_key, ssh_keyphrase, domain.	Yes
Use Kerberos KDC (Windows-only)	When enabled, Kerberos authentication is used to log in to Windows targets. Selecting Yes reveals the Key Distribution Center (KDC), KDC Port (default 88), KDC Transport (tcp / udp), and Domain fields.	No
Fetch Domain (Windows-only)	When enabled, the integration reads the api_domain field from the secret JSON and uses it as the Windows domain for the target authentication. Domain precedence: KDC Domain > manually entered Domain > domain from the secret.	No

Elevate privileges with (SSH-only)	The privilege escalation method used after initial authentication. Supported options include Nothing (default), .k5login, Cisco enable, dzdo, pbrun, su, su+sudo, sudo, and Checkpoint Gaia expert. Selecting a value reveals the escalation credential fields described in the Optional features section.	No
Escalation Credential ID (SSH-only)	The name of a separate Key Vault secret whose password field is used as the privilege escalation password. If left blank while an escalation method is selected, the primary secret's password is reused.	No
Database Port (Database-only)	The TCP port on the target database used for the credentialed check. Defaults to the standard port for the selected database type (for example 1521 for Oracle, 1433 for SQL Server, 3306 for MySQL, 5432 for PostgreSQL, 50000 for DB2, 27017 for MongoDB).	Yes
Database Name / SID (Database-only)	The database name, SID, or service name to connect to on the target. Optional for engines where a default database exists (for example postgres for PostgreSQL, admin for MongoDB).	No

Do one of the following:

- If you want to save without launching the scan, click **Save**.
- If you want to save and launch the scan immediately, click **Save & Launch**.

Note: If you scheduled the scan to run at a later time, the Save & Launch option is not available.

Configure Microsoft Azure Key Vault integration for Tenable Nessus

The Tenable Azure Key Vault integration exists as an authentication method within the SSH, Windows, Database, VMware ESXi SOAP API, VMware vCenter API, and Nutanix credential types in Tenable Nessus scan policies.

Required permissions

- A Tenable Nessus user account with permission to create and configure scans.
- The same Microsoft Entra ID service principal, client secret, and Key Vault Secrets User role assignment described in the Tenable One Vulnerability Management section above.

Scan configuration – Windows, SSH, Database, VMware, and Nutanix

Complete the following steps to configure Tenable Nessus with the Azure Key Vault integration using the Windows, SSH, or Database credential types.

1. Log in to your Tenable Nessus user interface.
2. In the left navigation pane, click **Scans**. The Scans page appears.
3. In the upper-right corner of the page, click the **Create a Scan** button. The Select a Scan Template page appears.
4. Select a scan template. The scan configuration page appears.
5. In the **Name** box, type a name for the scan.
6. In the **Targets** box, type an IP address, hostname, or range of IP addresses.

7. (Optional) Add a description, folder location, scanner location, and specify target groups.
8. Click the **Credentials** tab. The Credentials pane appears.
9. In the **Select a Credential** menu, select Windows, SSH, Database, VMware ESXi SOAP API, VMware vCenter API, or Nutanix from the options. The Settings pane for the selected credential appears.
10. In the **Auth Type** drop-down box, click **Azure Key Vault**. The Azure Key Vault options appear.
11. Configure each option for the selected Windows, SSH, Database, VMware, or Nutanix credential using the values in the table below.

Option	Description	Required
Username	The username to log in to the target system(s). Optional if a username field is present in the Azure Key Vault secret JSON; when supplied here it overrides the value stored in the secret.	No
Tenant ID	The Microsoft Entra ID (Azure AD) Directory (tenant) ID that owns the Key Vault. Used to build the OAuth2 token endpoint https://login.microsoftonline.com/<tenant_id>/oauth2/v2.0/token.	Yes
Application ID	The Application (client) ID of the Microsoft Entra ID service principal registered under App registrations. Also referred to as the Client ID.	Yes
Client Secret	The client secret value generated for the service principal under Certificates & secrets. Used together with the Application ID and Tenant ID for the OAuth2 client_credentials grant.	Yes

Key Vault Name	The name of the Azure Key Vault (not the full URL). Tenable derives the API host from this value as <Key Vault Name>.vault.azure.net.	Yes
Secret Name	The name of the Key Vault secret that stores the target credentials. The secret value is expected to be a JSON object containing one or more of the fields: username, password, ssh_key, ssh_keyphrase, domain.	Yes
Use Kerberos KDC (Windows-only)	When enabled, Kerberos authentication is used to log in to Windows targets. Selecting Yes reveals the Key Distribution Center (KDC), KDC Port (default 88), KDC Transport (tcp / udp), and Domain fields.	No
Fetch Domain (Windows-only)	When enabled, the integration reads the api_domain field from the secret JSON and uses it as the Windows domain for the target authentication. Domain precedence: KDC Domain > manually entered Domain > domain from the secret.	No
Elevate privileges with (SSH-only)	The privilege escalation method used after initial authentication. Supported options include Nothing (default), .k5login, Cisco enable, dzdo, pbrun, su, su+sudo, sudo, and Checkpoint Gaia expert. Selecting a value reveals the escalation credential fields described in the Optional features section.	No
Escalation Credential ID (SSH-only)	The name of a separate Key Vault secret whose password field is used as the privilege escalation password. If left blank while an escalation method is selected, the primary secret's password is reused.	No

Database Port (Database-only)	The TCP port on the target database used for the credentialed check. Defaults to the standard port for the selected database type (for example 1521 for Oracle, 1433 for SQL Server, 3306 for MySQL, 5432 for PostgreSQL, 50000 for DB2, 27017 for MongoDB).	Yes
Database Name / SID (Database-only)	The database name, SID, or service name to connect to on the target. Optional for engines where a default database exists (for example postgres for PostgreSQL, admin for MongoDB).	No

Do one of the following:

- If you want to save without launching the scan, click **Save**.
- If you want to save and launch the scan immediately, click **Save & Launch**.

> **Note**: If you scheduled the scan to run at a later time, the Save & Launch option is not available.

Scan Results Review

This section helps users interpret the results of their scans.

Plugin families and plugins

The Microsoft Azure Key Vault integration is used to gather credentials for target authentication during credentialed scans via SMB (Windows), SSH, Database, VMware vCenter, VMware ESXi, or Nutanix Prism Central. The following Tenable plugins are relevant when reviewing scan results:

Misc. plugin family:

- Plugin #204872: Integration Status: When using one of Tenable's integrations with any PAM integration, the Integration Status plugin confirms whether credential retrieval was a success or failure.
- Plugin #160185: Nutanix Data Collection: Reports data collected from Nutanix Prism Central via its REST APIs after the Azure Key Vault integration retrieves the credentials used to authenticate to the target.

Settings plugin family:

- Plugin #14273: SSH settings: Reads the SSH PAM credential configuration and calls the Azure Key Vault integration to retrieve credentials for SSH targets. This plugin is not indicative of authentication issues by itself; authentication failures are reported by the credential-status plugins below.
- Plugin #10870: Login configurations: Reads Windows (SMB) credential configuration and calls the Azure Key Vault integration to retrieve credentials for Windows targets.
- Plugin #33815: Database settings: Reads database credential configuration and calls the Azure Key Vault integration to retrieve credentials for database targets.

- Plugin #19506: Nessus Scan Information: Reports scan metadata including whether credentialed checks succeeded. Check this plugin first when investigating authentication issues.
- Plugin #91822: Database Authentication Failure(s) for Provided Credentials: Indicates that the database credentials retrieved from Azure Key Vault could not authenticate to the target database. Check that the Secret Name resolves to a JSON object containing a valid username and password for the target database.
- Plugin #141118: Target Credential Status by Authentication Protocol – Valid Credentials Provided: Confirms credential validity by successfully authenticating to the remote target. Look for "Proto: SMB" or "Proto: SSH" in the output.
- Plugin #110095: Target Credential Issues by Authentication Protocol – No Issues Found: Indicates that credentials were provided and authentication succeeded for all targeted protocols.
- Plugin #104410: Target Credential Status by Authentication Protocol – Failure for Provided Credentials: Indicates that credentials were provided but authentication to the target failed — the values retrieved from Azure Key Vault may be incorrect or lack sufficient permission on the target.
- Plugin #110723: Target Credential Status by Authentication Protocol – No Credentials Provided: Indicates that no credentials were available for the targeted protocol. If this appears when an Azure Key Vault credential is configured, the integration likely failed to retrieve the secret.
- Plugin #117885: Target Credential Issues by Authentication Protocol – Intermittent Authentication Failure: Indicates that authentication succeeded for some targets but not others — may point to per-target credential differences or intermittent Key Vault connectivity.

VMWare ESX Local Security Checks family:

- Plugin #57400 (Scan Results Review): VMware vSphere installed VIBs: Reports the installed VIBs collected on a VMware ESXi or vCenter host after the Azure Key Vault integration retrieves the credentials used to authenticate to the target.

Debug log reporting

Logs generated by other plugins are reported by Plugin #84239: Debugging Log Report. Debug logs for the Azure Key Vault integration are written by the Tenable Nessus scanner during the scan. Debugging log report output is controlled by the debug logging settings in Nessus.

The integration writes to log files named after the credential plugin that invoked it, with "Azure Key Vault" appended so the PAM-integration log lines can be distinguished from the target-authentication log lines:

- SSH: ssh_settings.nasl~Azure Key Vault
- Windows (SMB): logins.nasl~Azure Key Vault
- Database: database_settings.nasl~Azure Key Vault
- VMware vCenter: vmware_vcenter_settings.nbin~Azure Key Vault
- VMware ESXi: vmware_soap_settings.nbin~Azure Key Vault
- Nutanix Prism Central: nutanix_settings.nasl~Azure Key Vault

What the logs contain:

- Configuration settings loaded from the scan policy (with sensitive values — secret values, access tokens, passwords, and SSH keys — automatically scrubbed).
- The OAuth2 token request against login.microsoftonline.com and whether authentication to Microsoft Entra ID succeeded.
- Cache hit/miss information for both the access token and the retrieved secret.
- The Key Vault secret GET request URL (with the Key Vault name and secret name) and the HTTP response status.

- Whether the secret value was successfully parsed as JSON, and which fields were extracted (with sensitive values masked).
- Any errors returned by Microsoft Entra ID or Azure Key Vault, including 401 Unauthorized (bad tenant/client credentials), 403 Forbidden / ForbiddenByRbac (missing Key Vault Secrets User role), and 404 Not Found (secret does not exist or vault name is wrong).

Common reasons for credentialed checks showing "no" status:

- The service principal does not have the Key Vault Secrets User role on the target vault, or the role assignment has not yet propagated (RBAC changes can take several minutes).
- The Client Secret was rotated in Azure and the new value was not updated in the Tenable scan credential.
- The Tenant ID, Application ID, or Key Vault Name in the credential does not match the values registered in Azure.
- The Key Vault firewall is enabled and the scanner's public egress IP is not on the allow-list.
- The Key Vault secret value is not valid JSON (or is JSON that is missing the password / ssh_ key fields), causing the parsed credential to be empty.
- The scanner host resolves login.microsoftonline.com or *.vault.azure.net over IPv6 only. If the scanner network does not have working IPv6 egress, prefer IPv4 for these hostnames on the scanner.

Optional features

Privilege escalation

Privilege escalation can be configured to work with the Azure Key Vault integration within the SSH credential type.

You can add privilege escalation while configuring an SSH credentialed scan with the Azure Key Vault integration using the Elevate Privileges with field option, which lets you select the privilege account type used to gain elevated access to the target machine.

The available privilege account types are:

- Nothing (the default)
- .k5login
- Cisco 'enable'
- Dzdo
- Pbrun
- Su
- su+sudo
- sudo
- Checkpoint Gaia 'expert'

Each escalation type can be configured with the additional fields below. When Escalation Credential ID is supplied, the integration performs one extra Key Vault GET request to fetch that secret and uses its password field as the sudo/su password. If Escalation Credential ID is left blank, the password from the primary Key Vault secret is reused as the escalation password.

For the sudo escalation type specifically, the standard sudo prompt expects the invoking user's own password rather than a separate root-account password. In that case, leave Escalation Credential ID blank so the integration reuses the primary Key Vault secret's password (which corresponds to the account identified by Username) as the sudo password — no additional Key Vault lookup is performed.

Option	Description	Required
Escalation Credential ID	The name of the Azure Key Vault secret that contains the escalation account credentials. The password field of that secret becomes the sudo/su password. Configurable within: .k5login, Cisco enable, dzdo, pbrun, su, su+sudo, sudo, Checkpoint Gaia expert.	Yes
Escalation Account Name	The username for the account with elevated privileges. When left blank, the username retrieved from the primary Key Vault secret is used. Configurable within: .k5login, Cisco enable, dzdo, pbrun, su, sudo, su+sudo, Checkpoint Gaia expert.	No
Location of [account type] (directory)	The directory path for the escalation binary (for example /usr/bin for sudo). Configurable within: dzdo, pbrun, su, su+sudo, sudo.	No