Azure Key Vault Governance Model

**Establishing Governance Model for Azure Key Vaults**



Table of Contents

[1 Introduction 4](#_Toc122535711)

[1.1 Document scope 4](#_Toc122535712)

[1.2 Document objective 4](#_Toc122535713)

[1.3 Target Audience 4](#_Toc122535714)

[1.4 References 4](#_Toc122535715)

[1.5 Change History 5](#_Toc122535716)

[1.6 Forecast changes 5](#_Toc122535717)

[1.7 Open points 5](#_Toc122535718)

[2 Azure Key Vault Usage 6](#_Toc122535719)

[2.1 When is it required to use Key Vaults? 6](#_Toc122535720)

[2.2 Using Separate Key Vaults 6](#_Toc122535721)

[2.3 Reasoning 6](#_Toc122535722)

[3 Key Vault Authentication and Authorization (Access Controls) 7](#_Toc122535723)

[3.1 Why access controls? 7](#_Toc122535724)

[3.2 Role Separation 7](#_Toc122535725)

[3.3 Privileged Access 7](#_Toc122535726)

[3.4 Role-Based Access Control (RBAC) 8](#_Toc122535727)

[3.5 RBAC at XYZ 9](#_Toc122535728)

[3.5.1 Recommended RBAC model 10](#_Toc122535729)

[3.5.2 Minimum RBAC model 10](#_Toc122535730)

[3.6 Limitations and Considerations 10](#_Toc122535731)

[3.7 Summary 11](#_Toc122535732)

[4 Ownership of Key Vaults 12](#_Toc122535733)

[4.1 Application/resources owners 12](#_Toc122535734)

[4.2 Roles and responsibilities of key vault owners 12](#_Toc122535735)

[4.3 Compliance and remediation tasks 12](#_Toc122535736)

[5 Recovery and Protection 13](#_Toc122535737)

[5.1 Availability and Redundancy 13](#_Toc122535738)

[5.2 Preventing malicious and accidental deletion 13](#_Toc122535739)

[5.3 Soft-delete 13](#_Toc122535740)

[5.4 Purge protection 14](#_Toc122535741)

[6 Network Access Control 15](#_Toc122535742)

[6.1 Network Security 15](#_Toc122535743)

[6.2 Firewall and virtual networks 15](#_Toc122535744)

[6.3 Private endpoint connections 15](#_Toc122535746)

[7 Logging and Monitoring 16](#_Toc122535747)

[7.1 Logging 16](#_Toc122535748)

[7.2 Defender for Key vault 16](#_Toc122535749)

[7.3 Alerts 17](#_Toc122535750)

[7.4 Responsibilities 17](#_Toc122535751)

[8 Azure Policies 18](#_Toc122535752)

[8.1 Using Azure policies to enforce policies on Key Vaults 18](#_Toc122535753)

[8.2 Types of policy effects and guidance 18](#_Toc122535754)

[8.3 Policies Definitions 19](#_Toc122535755)

[8.3.1 Key vaults should have purge protection enabled (enforce) 19](#_Toc122535756)

[8.3.2 Key vaults should have soft delete enabled (enforce) 19](#_Toc122535757)

[8.3.3 Azure Key Vault should disable public network access (enforce) 19](#_Toc122535758)

[8.3.4 Azure Key Vault should have firewall enabled (enforce) 19](#_Toc122535759)

[8.3.5 Configure key vaults to enable firewall (enforce) 19](#_Toc122535760)

[8.3.6 Resource logs in Key Vault should be enabled (enforce) 19](#_Toc122535761)

[8.3.7 Certificates should use allowed key types (enforce) 19](#_Toc122535762)

[8.3.8 Certificates should have the specified lifetime action triggers (optional) 20](#_Toc122535763)

[8.3.9 Key Vault keys should have an expiration date (optional) 20](#_Toc122535764)

[8.3.10 Key Vault secrets should have an expiration date (optional) 20](#_Toc122535765)

[8.3.11 Keys should have the specified maximum validity period (optional) 20](#_Toc122535766)

[8.3.12 Secrets should have the specified maximum validity period (optional) 20](#_Toc122535767)

[8.3.13 Keys should be backed by a hardware security module (HSM) (depends on app sensitivity) 20](#_Toc122535768)

[8.3.14 Secrets should have content type set (optional) 20](#_Toc122535769)

[8.3.15 Keys using RSA cryptography should have a specified minimum key size (enforce) 20](#_Toc122535770)

[8.3.16 Keys should not be active for longer than the specified number of days (optional) 20](#_Toc122535771)

[8.3.17 Certificates using RSA cryptography should have the specified minimum key size (enforce) 21](#_Toc122535772)

[8.4 Compliance and remediation 21](#_Toc122535773)

[9 Appendix 22](#_Toc122535774)

[9.1 Glossary 22](#_Toc122535775)

[9.2 Acronym & Abbreviation 23](#_Toc122535776)

# Introduction

## Document scope

Define and describe the governance model for Azure Key Vault(s) at XYZ.

## Document objective

At XYZ, IT services rely heavily on Azure Cloud for hosting and operating several IT components, including, but not limited to, infrastructure, applications, databases, VMs, identities and security.

Azure Key Vault is a cloud service offered by Microsoft Azure that enables Azure applications and users to securely store and access secrets. A secret is anything that we want to tightly control access to, such as API keys, passwords, certificates and cryptographic keys. Key Vault service supports two types of containers: vaults and managed hardware security module (HSM) pools. Vaults support storing software and HSM-backed keys, secrets and certificates.

The purpose of this document aims to describe and establish the controls around Azure Key Vaults and how they should be governed, secured and operated by describing the processes, procedures, roles and responsibilities. i.e., to establish a framework for Azure Key Vault usage within XYZ.

This document follows Microsoft Best Practices for using Azure Key Vault[a].

## Target Audience

Cloud Team, Enterprise Cryptography.

## References

|  |  |  |
| --- | --- | --- |
| Ref. | Doc. Title, version, location | Date |
| [a] | [Best practices for using Azure Key Vault | Microsoft Learn](https://learn.microsoft.com/en-us/azure/key-vault/general/best-practices) | 09.09.2022 |
| [b] | [Azure Key Vault security overview | Microsoft Learn](https://learn.microsoft.com/en-us/azure/key-vault/general/security-features) | 27.09.2022 |
| [c] | [Azure Resource Manager overview - Azure Resource Manager | Microsoft Learn](https://learn.microsoft.com/en-us/azure/azure-resource-manager/management/overview) | 12.10.2022 |
| [d] | [Azure Key Vault soft-delete | Microsoft Learn](https://learn.microsoft.com/en-us/azure/key-vault/general/soft-delete-overview) | 15.9.2022 |
| [e] | [Enable soft-delete on all key vault objects - Azure Key Vault | Microsoft Learn](https://learn.microsoft.com/en-us/azure/key-vault/general/soft-delete-change) | 19.7.2022 |
| [f] | [Overview of Azure Policy - Azure Policy | Microsoft Learn](https://learn.microsoft.com/en-us/azure/governance/policy/overview) | 04.08.2022 |
| [g] | [Azure Key Vault availability and redundancy - Azure Key Vault | Microsoft Learn](https://learn.microsoft.com/en-us/azure/key-vault/general/disaster-recovery-guidance) | 12.02.2021 |
|  |  |  |
|  |  |  |

## Change History

|  |  |  |
| --- | --- | --- |
| Version | Nature of changes | Date |
| 0.01 | Initial draft | xx.xx.xxxx |
| 1.00 | First release | xx.xx.xxxx |
|  |  |  |

## Forecast changes

|  |  |  |
| --- | --- | --- |
| Version | Nature of changes | Date |
| 0.01 | Initial draft | xx.xx.xxxx |
| 1.00 | First release | xx.xx.xxxx |
| 1.10 | To update the permission model and revise the limitations after the release of PCFv2 | xx.xx.xxxx |

## Open points

|  |  |  |
| --- | --- | --- |
| Ref. | Description, status, actionee | Date |
| [a] | Migration of existing Key Vaults from Access Policy to RBAC | xx.xx.xxxx |
| [b] |  |  |
| [c] |  |  |

# Azure Key Vault Usage

## When is it required to use Key Vaults?

Generally, whenever there’s a secret to store (whether it’s encryption key, certificate, connection string, password, API key, etc.), a Vault SHOULD be used to store and safeguard those secrets. Azure key vault is one of the options to store secrets and keys and is the recommended solution for Azure cloud resources.

## Using Separate Key Vaults

Services/applications that use/want to use the key vault MUST use a separate key vault per application per environment (development, pre-production and production), per region[a]. This control helps to not share secrets across environments and regions.

Storing secrets into the same vault increase the risk and impact when compromised because attacks might be able to access secrets across different areas.

## Reasoning

The main benefits of using sperate key vaults are:

* Reduce the impact when a key vault is compromised (undesired/unauthorized access)
* It allows for better manageability
* Easier for enforcing policies (chapter 8)
* In case a key vault is compromised, the impact is lesser to take one down (smaller scope) rather than taking down a key vault that contains all secrets

# Key Vault Authentication and Authorization (Access Controls)

## Why access controls?

Encryption keys and secrets like certificates, connection strings, password and API keys are very sensitive and business critical. Therefore, it’s essential to secure the access to the key vaults by allowing ONLY authorized applications and users and implement the concept of Least Privileged. Azure Key Vault Security Features[b] provides an overview of the Key Vault access model and explains authentication and authorization.

## Role Separation

Role separation is an essential step in designing and implementing Key/Secrets Management (in this case, Key Vault). In essence, it’s the best way to protect ourselves from malicious/accidental actions that could lead to a key/secret compromise. Also, since an account can be hijacked, role separation prevents a single person form doing everything on the key vault.

Microsoft Azure Key Vault Access Model controls the access to the key vault via two interfaces: the **management plane** and the **data plane**.

The management plane is where the key vault itself is managed. Operations in this plane include creating and deleting key vaults, retrieving/reading Key Vault properties, as well as updating access policies.

The data plane is where the users/consumers work with the data stored in the key vault. They can add, read, delete and modify keys, secrets, passwords and certificates.

Both planes use Azure Active Directory for authentication. However, for authorization, the management plane only uses Azure role-based access control (Azure RBAC) and the data plane uses both Key Vault access policy and Azure RBAC for Key Vault data plane operations.

To access a key vault in either plane, all callers (users or applications) must have proper authentication and authorization. **Authentication** establishes the identity of the caller. **Authorization** determines which operation the caller can execute.

Authentication with Key Vault works in conjunction with Azure Active Directory (Azure AD), which is responsible for authenticating the identity of any given security principal.

A security principal is an object that represents a user, group, service, or application that's requesting access to Azure resources. Azure assigns a unique object ID to every security principal.

## Privileged Access

Authorization determines which operations the caller can perform. Authorization in Key Vault uses Azure role-based access control (Azure RBAC) on management plane and either Azure RBAC or Azure Key Vault access policies on data plane.

As mentioned earlier, access to the key vault can take place via two interfaces or planes, **management plane** and the **data plane**.

Applications/callers access the planes through endpoints. The access control for the two planes works independently. To grant an application access to use objects in the key vault, a data plane access is granted using Azure RBAC or Key Vault Access Policy.

To grant a user read access to Key Vault properties and tags, but not access to data (keys, secrets, certificates, API keys, passwords, etc.), a management plane access is granted with Azure RBAC.

This table shows the endpoints for the management and data planes.

|  |  |  |  |
| --- | --- | --- | --- |
| Access Plane | Access Endpoints | Operations | Access control mechanism |
| Management Plane | **Global**:  management.azure.com:443  **Azure Germany**:  management.microsoftazure.de:443 | Create, read, update, and delete key vaults  Set Key Vault access policies  Set Key Vault tags | Azure RBAC |
| Data Plane | **Global:**  <vault-name>.vault.azure.net:443  **Azure Germany**:  <vault-name>.vault.microsoftazure.de:443 | **Keys**: encrypt, decrypt, wrapKey, unwrapKey, sign, verify, get, list, create, update, import, delete, recover, backup, restore, purge, rotate (preview), get rotation policy (preview), set rotation policy (preview), release(preview)  **Certificates**: manage contacts, get issuers, list issuers, set issuers, delete issuers, manage issuers, get, list, create, import, update, delete, recover, backup, restore, purge  **Secrets**: get, list, set, delete, recover, backup, restore, purge | Key Vault access policy or Azure RBAC |

## Role-Based Access Control (RBAC)

Azure RBAC is an authorization system built on Azure Resource Manager[c] that provides fine-grained access management to Azure resources.

The Azure RBAC model provides the ability to set permissions on different scope levels: management group, subscription, resource group, or individual resources. Azure RBAC for key vault also provides the ability to have separate permissions on individual keys, secrets, and certificates.

Using RBAC, we can limit access to individual objects (keys, secrets, certificates, etc.) to a particular security principal (user/application/service) to consume the secret (read/get and perform cyrpto operations) and remove all other permissions from that account.

**For example**, a key vault includes secret-a and secret-b. App-1 uses *secret-a* and app-2 uses *secret-b*. with RBAC, we can limit App-1 to only **READ** *secret-a* and no other permissions whatsoever on any other security object in the vault, or vault management settings.

Also, roles can be divided between administrator/officer/contributor, with different rights/permission for each. That helps to achieve the role separation and segregation of duty.

This table show the main roles and their permissions on Azure Key Vault

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Azure Key Vault Role | Permissions | | | | |
| List Objects | Manage KV Settings | Manage Objects | Read Secrets | Manage Access |
| Key Vault Reader | X | - | - | - | - |
| Key Vault Administrator | X | - | X | X | - |
| Key Vault Contributor | - | X | - | - | - |
| User Access Administrator | - | - | - | - | X |
| Key Vault Secret/Crypto Officer | X | - | X | X | - |
| Key Vault Secret/Crypto User | X | - | - | X | - |

## RBAC at XYZ

Every key vault SHOULD ideally have the below roles defined and assigned with the responsibilities/activities described.

But when it’s not possible to use all the below roles, at least the first and last roles **MUST** be defined and enforced.

|  |  |  |
| --- | --- | --- |
| Custom Role | Description | Responsibilities/Activities |
| XYZ-KV-Owner | The administrator for the entire key vault. | In change of the entire key vault and the objects in it  Responsible for settings and complying with the security baseline  Responsible for setting up permission and adding role assignment for officers/users/apps (consumers)  Ensures that apps have individual secrets access |
| XYZ-KV-Officer | The officer in charge for the objects inside the key vault | Creates new secrets  Maintains backup  Manages secret objects (validity, enable/disable) |
| XYZ-KV-App | A user of the key vault who can read/view secrets and/or the end app/service that consumes the secret | Can copy secret values (in case those secrets are meant to be used by humans and not apps/services)  Can only fetch/get a secret from the key vault has to be defined per secret object |

### Recommended RBAC model

The below table shows an example of a key vault with multiple roles and their permissions. This is the ideal situation and SHOULD be implemented whenever possible

|  |  |  |  |
| --- | --- | --- | --- |
| Custom Role | Azure built-in roles | Notes | Scope |
| XYZ-KV-Owner | Key Vault Administrator  Key Vault Contributor  User Access Administrator | This role has Full-Control on the entire key vault, its objects and the access control | Entire Key Vault |
| XYZ-KV-Officer | Key Vault Secrets Officer  Key Vault Crypto Officer  Key Vault Certificates Officer | This role can perform any action on keys/certificates/secrets, except manage permissions  This can be thought of as admin of the contents inside the key vault. This role can manage the objects (disable/enable) change settings on them | Entire Key Vault |
| XYZ-KV-App | Key Vault Secrets User  Key Vault Crypto User | Read secrets content  Perform crypto ops using keys  This role can read the content of the secrets and use the keys for cryptographic operations  This role is for the actual consumer of the object inside the key vault and should be assigned per object. | Per object (secret, key, certificate) |

### Minimum RBAC model

The below table shows an example of a key vault with only two roles and their permissions. This is the absolute minimum for how roles to be setup on a key vault.

|  |  |  |  |
| --- | --- | --- | --- |
| Custom Role | Azure built-in roles | Notes | Scope |
| XYZ-KV-Owner | Key Vault Administrator  Key Vault Contributor  User Access Administrator | This role has Full-Control on the entire key vault, its objects and the access control | Entire Key Vault |
| XYZ-KV-App | Key Vault Secrets User  Key Vault Crypto User | This is a non-human account (e.g., App or service account).  The role here is only read the content of a secret or perform cryptographic operations on a key.  This role is for the actual consumer of the object inside the key vault and should be assigned per object. | Per object (secret, key, certificate) |

## Custom Roles

The custom roles mentioned in the above tables do not exist right now and will need to be created first. Each custom role contains multiple Azure built-in roles that are needed for an owner to manage their key vault or for an application to use their secrets/keys in the vault.

The reason for reason custom roles is to group several built-in azure roles (permissions) that are required for certain functionalities (owner/consumer) with clear names that match those functionalities.

## Limitations and Considerations

Due to the current setup of Azure at XYZ based on the Public Cloud Framework version 1 (PCFv1), the below limitations and constraints have been identified

* RBAC permission model allows to assign access to individual objects in Key Vault to user or application, but any administrative operations like network access control, monitoring, and objects management require vault level permissions which will then expose secure information to operators across application teams.
* As per \*\*\* IAM standard, Application team will be granted with permission on the scope of their application team's resource group.
* The user who creates the key vault will get all the permissions on Key, Secrets and Certificates when permission model is selected as "Vault access Policy".
* In case application team member is assigned with CMC-PaaS-Contributor role, he/she will have permissions to create new key vaults.
* Role assignments disappeared when Key Vault was deleted (soft-delete) and recovered - it's currently a limitation of soft-delete feature across all Azure services. It's required to recreate all role assignments after recovery.

## Summary

Without RBAC, any user/application with access to the key vault can have the same permission on all objects inside the key vault. That means, we cannot limit which app can access which objects and there’s no fine-grained permission set.

RBAC MUST be used for both data plane and management plane, in order to give more granularity and access control over the kay vault.

Section 3.5 above outlines the RBAC model that SHOULD be used at XYZ. All Key Vault owners are required to maintain the permissions on their key vault using RBAC as defined above.

**Important notes:**

* *As per the current Public Cloud Framework at XYZ (PCFv1),* ***User Access Administrator*** *permission CANNOT be provided to application managers. Therefore, implanting/enforcing RBAC permission model currently will bring many challenges.*
* *However, XYZ is working on future enablement of an alternative using Constrained Delegation (currently in Private Preview from Microsoft).*
* *Enforcing RBAC will be revised after Constrained Delegation will become publicly available from Microsoft.*
* *Therefore, for all existing Key Vaults (as of the time of releasing this document), and for all new key vault, Access Policy model must remain due to the points mentioned above and due to the* ***migration constraints****.*

# Ownership of Key Vaults

## Application/resources owners

Each application owner (or resource owner) will have their own key vault per environment (development, pre-production and production).

It’s not allowed to mix secrets of different environments in the same vault. But since this control is not feasible to be enforced, key vault owners/users need to be educated and made aware of the importance of such rule.

## Roles and responsibilities of key vault owners

A key vault owner (XYZ-KV-Owner) is responsible for:

* Managing the key vault
* Managing permissions and role assignments to the key vault
* Managing objects and their Access Control
* Managing Key Vault Settings:
  + Access configuration
  + Key vault Network level security
  + Key Vault Properties (soft-delete and Purge protection)
  + Logging and alerting
* Ensuring compliance of the Key Vault
* Closing remediation tasks will be handled via opsWork(using central automation “opswork” to create WI for app managers to remediate)

## Compliance and remediation tasks

Every Key Vault MUST be compliant with the policies defined in this governance model and in section-8 of this document.

It’s the responsibility of the Key Vault owner (XYZ-KV-Owner) to make sure that their Key Vault is compliant with those policies.

Some of the policies can be enforced by Azure Policies (section-8) and some cannot (e.g., assigning the right roles/permission for apps consuming objects in the key vault).

For Azure Policies, Key Vault owners can go and check if their vault is compliant under Policy -> Compliance.

For the policies that are not enforced by Azure Policies, every Key Vault owner MUST assure that their vault is compliant with those policies manually by auditing the key vault regularly.

# Recovery and Protection

## Availability and Redundancy

Azure Key Vault features multiple layers of redundancy to make sure that your keys and secrets remain available to your application even if individual components of the service fail[g].

The contents of the key vault are replicated within the region and to a secondary region at least 150 miles away, but within the same geography to maintain high durability of the keys and secrets.

If individual components within the key vault service fail, alternate components within the region step in to serve the request to make sure that there is no degradation of functionality. There’s no action needed to be taken to start this process, it happens automatically and will be transparent.

In the rare event that an entire Azure region is unavailable, the requests that is made of Azure Key Vault in that region are automatically routed (failed over) to a secondary region.

Through this high availability design, Azure Key Vault requires no downtime for maintenance activities.

**Note**:

*In the event of a region failover, it may take a few minutes for the service to fail over. Requests that are made during this time before failover may fail.*

During failover, the key vault is in read-only mode. Requests that are supported in this mode are:

List certificates. Get certificates. List secrets. Get secrets. List keys. Get (properties of) keys. Encrypt. Decrypt. Wrap. Unwrap. Verify. Sign. Backup.

During failover, it’s not possible to make changes to key vault properties, change access policy or firewall configurations and settings.

After a failover is failed back, all request types (including read and write requests) are available.

## Preventing malicious and accidental deletion

Azure Key Vault has two recovery features, soft delete and purge protection. Those features are there to protect against accidental/malicious deletion of the key vault and keys, secrets and certificates stored inside the key vault and to protect for purging after deletion (allow objects to remain for a period before they’re permanently deleted).

## Soft-delete

Soft delete is designed to prevent accidental deletion of your key vault and keys, secrets, and certificates stored inside key vault. Think of soft-delete like a recycle bin. When you delete a key vault or a key vault object, it will remain recoverable for a user configurable retention period or a default of 90 days. Key vaults in the soft deleted state can also be purged which means they are permanently deleted. This allows you to recreate key vaults and key vault objects with the same name. Both recovering and deleting key vaults and objects require elevated access policy permissions. Once soft delete has been enabled, it cannot be disabled.

**Important notes about soft-delete**:

*Soft-delete MUST be enabled immediately. The ability to opt out of soft-delete has been deprecated by Microsoft and will be removed in February 2025*[d]*.*

*When a Key Vault is soft-deleted, services that are integrated with the Key Vault will be deleted. For example: Azure RBAC roles assignments and Event Grid subscriptions. Recovering a soft-deleted Key Vault will not restore these services. They will need to be recreated*[d]*.*

It is important to note that **key vault names are globally unique**, so it’s not possible to create a key vault with the same name as a key vault in the soft deleted state. Similarly, the names of keys, secrets, and certificates are unique within a key vault. It’s not possible to create a secret, key, or certificate with the same name as another in the soft deleted state.

## Purge protection

**Purge protection** is designed to prevent the deletion of key vault, keys, secrets, and certificates by a malicious insider. It can be thought of as a recycle bin with a time based lock. You can recover items at any point during the configurable retention period. **It’s not possible to permanently delete or purge a key vault until the retention period elapses.** Once the retention period elapses the key vault or key vault object will be purged automatically.

Purge Protection MUST be enabled on all Key Vaults to prevent accidental/malicious deletion.

Key vaults that are in DEV environment MAY have this setting disabled. This is to allow the deletion/recreation of key vaults as needed.

**Note**

*Purge Protection is designed so that no administrator role or permission can override, disable, or circumvent purge protection.* ***Once purge protection is enabled, it cannot be disabled or overridden by anyone including Microsoft****. This means you must recover a deleted key vault or wait for the retention period to elapse before reusing the key vault name.*

# Network Access Control

## Network Security

Access to the Key Vault needs to be secured on different layers, network access is one of them.

This section describes the use of firewall to control/limit the access to the key vault

## Firewall and virtual networks

By default, networking on Azure Key vault allows public access from all networks, which is not recommended and makes the key vault exposed to the public internet.

All key vaults MUST disable public access from all networks.

If public access is absolutely necessary, then it MUST be ONLY allowed from specific virtual networks and IP addresses.

## Private endpoint connections

Private endpoints connections allow access to the Key Vault resource using a private IP address from a virtual network. This is the recommended setup for the Key Vault.

Key Vault owners are REQUIRED to specific which private IP addresses to be allowed from virtual networks.

# Logging and Monitoring

Since Key Vault is a critical resource and contains restricted data, we need to monitor those resources for security, availability, performance and operation.

Key Vault logging saves information about the activities performed on a given vault.

Key Vault can be integrated with Event Grid so that notifications can be sent when the status of a key, certificate, or secret stored in key vault has changed.

It is also important to monitor the health of our key vaults, to make sure that our services are operating as intended.

This section covers how logging, monitoring and alerting should be setup across all key vaults at XYZ

## Logging

In order to be able to monitor how and when key vaults are accessed, and by whom, logging for Azure Key Vault must be enabled, which saves information in an Azure storage.

Logging information can be accessed within 10 minutes (at most) after the key vault operation.

What is logged:

* All authenticated REST API requests, including failed requests as a result of access permissions, system errors, or bad requests.
* Operations on the key vault itself, including creation, deletion, setting key vault access policies, and updating key vault attributes such as tags.
* Operations on keys and secrets in the key vault, including:
  + Creating, modifying, or deleting these keys or secrets.
  + Signing, verifying, encrypting, decrypting, wrapping and unwrapping keys, getting secrets, and listing keys and secrets (and their versions).
* Unauthenticated requests that result in a 401 response. Examples are requests that don't have a bearer token, that are malformed or expired, or that have an invalid token.
* Azure Event Grid notification events for the following conditions: expired, near expiration, and changed vault access policy (the new version event isn't logged). Events are logged even if there's an event subscription created on the key vault.

All key vault owners MUST enable logging on all their key vaults. Key vault owners can follow the instructions in the SOP document CSF-PKI-SOP-001 Configure Azure Key Vaults Based on XYZ Governance Model[h].

## Defender for Key vault

**Microsoft Defender for Key Vault** is an Azure-native threat protection service, which detects unusual and potentially harmful access to Key Vault accounts. It provides an additional layer of security intelligence for the keys, secrets and certificates stored in the Microsoft Key Vault by alerting you to suspicious or malicious access. This layer of security allows you to address threats without being a security expert, and without the need to manage third-party security monitoring systems.

When anomalous activities occur, Microsoft Defender for Key Vault shows alerts and optionally sends them via email to relevant members of your organization. These alerts include the details of suspicious activity and recommendations on how to investigate and remediate the threats.

Microsoft Defender for Key Vault will be enforced on all key vaults via Azure Policy.

## Alerts

When a key vault owner receives an alert from Microsoft Defender for Key Vault, CDC will investigate the alert with the Key Vault owner from which the alert has originated and verify the situation.

Microsoft Defender for Key Vaults protest applications and credentials, so even if the owner is familiar with the application or user that triggered the alert, it’s important to verify the situation for every alert.

The following steps describe on a high-level how an alert will be investigated and reported upon.

* Step 1. Identify the resource
* Step 2. Respond accordingly
* Step 3. Measure the impact
* Step 4. Take action

## Responsibilities

Microsoft Defender for Key Vault (part of Microsoft Defender for cloud) is managed and monitored by CDC.

Whenever there’s an alert, CDC will investigate it with the key vault owner.

# Azure Policies

## Using Azure policies to enforce policies on Key Vaults

Azure Policy[f] is a governance tool that gives users the ability to audit and manage their Azure environment at scale. Azure Policy provides the ability to place guardrails on Azure resources to ensure they are compliant with assigned policy rules. It allows users to perform audit, real-time enforcement, and remediation of their Azure environment. The results of audits performed by policy will be available to users in a compliance dashboard where they will be able to see a drill down of which resources and components are compliant and which are not.

At XYZ, we opted to integrate Azure Key Vaults with Azure Policies mainly for the following reasons:

* We want to improve the security posture of XYZ by implementing requirements around minimum key sizes and maximum validity periods of certificates in our key vaults, but you don't know which teams will be compliant and which are not.
* We don’t’ want to rely on an external solution to perform an audit on the Key Vaults across XYZ.
* We want to limit the manual audits of our environment by asking individual teams within XYZ to report their compliance. So, we are looking for a way to automate this task, perform audits in real time, and guarantee the accuracy of the audit.
* We want to enforce XYZ’s security policies and stop individuals from performing actions that do not conform with XYZ’s security policy.
* We want to be sure that we can roll-back enforcement of new policies in the event of a live-site issue. We need a one-click solution to turn off enforcement of the policy when needed.

## Types of policy effects and guidance

**Audit**: When the effect of a policy is set to audit, the policy will not cause any breaking changes to the environment. It will only alert key vault owners to components such as certificates that do not comply with the policy definitions within a specified scope, by marking these components as non-compliant in the policy compliance dashboard. Audit is default if no policy effect is selected.

**Deny**: When the effect of a policy is set to deny, the policy will block the creation of new components such as certificates as well as block new versions of existing components that do not comply with the policy definition. Existing non-compliant resources within a key vault are not affected. The 'audit' capabilities will continue to operate.

## Policies Definitions

Azure Key Vault has created a set of policies, which can be used to manage key vaults and its key, certificate, and secret objects. These policies are 'Built-In', which means they don't require writing of any custom JSON to enable them, and they are available in the Azure portal for assignment. It’s still possible to customize certain parameters to fit our organization’s needs.

### Key vaults should have purge protection enabled (enforce)

Malicious deletion of a key vault can lead to permanent data loss. A malicious insider can potentially delete and purge key vaults. Purge protection protects you from insider attacks by enforcing a mandatory retention period for soft deleted key vaults. No one inside the organization or Microsoft will be able to purge key vaults during the soft delete retention period.

**This policy SHOULD NOT be enforced for key vaults that are in DEV environment.**

### Key vaults should have soft delete enabled (enforce)

Deleting a key vault without soft delete enabled permanently deletes all secrets, keys, and certificates stored in the key vault. Accidental deletion of a key vault can lead to permanent data loss. Soft delete allows recovery of an accidentally deleted key vault for a configurable retention period.

### Azure Key Vault should have firewall enabled (enforce)

Enable the key vault firewall so that the key vault is not accessible by default to any public IPs. A key vault owner can then configure specific IP ranges to limit access to those networks.

### Resource logs in Key Vault should be enabled (enforce)

Audit enabling of resource logs. This enables the recreation of activity trails to use for investigation purposes when a security incident occurs, or the network is compromised.

### Certificates should use allowed key types (enforce)

This allows us to manage XYZ’s CSF-PKI compliance requirements by restricting the key types allowed for certificates.

### Certificates should have the specified lifetime action triggers (optional)

This allows us to manage XYZ’s CSF-PKI compliance requirements by specifying whether a certificate lifetime action is triggered at a specific percentage of its lifetime or at a certain number of days prior to its expiration.

### Key Vault keys should have an expiration date (optional)

Cryptographic keys should have a defined expiration date and not be permanent. Keys that are valid forever provide a potential attacker with more time to compromise the key. It is a recommended security practice to set expiration dates on cryptographic keys.

### Key Vault secrets should have an expiration date (optional)

Secrets should have a defined expiration date and not be permanent. Secrets that are valid forever provide a potential attacker with more time to compromise them. It is a recommended security practice to set expiration dates on secrets.

### Keys should have the specified maximum validity period (optional)

This allows us to manage XYZ’s CSF-PKI compliance requirements by specifying the maximum amount of time in days that a key can be valid within a key vault.

### Secrets should have the specified maximum validity period (optional)

This allows us to manage XYZ’s CSF-PKI compliance requirements by specifying the maximum amount of time in days that a secret can be valid within a key vault.

### Secrets should have content type set (optional)

A content type tag helps identify whether a secret is a password, connection string, etc. Different secrets have different rotation requirements. Content type tag should be set on secrets.

### Keys using RSA cryptography should have a specified minimum key size (enforce)

Use of RSA keys with small key sizes is not a secure practice and doesn't meet many industry certification requirements. At XYZ, the minimum allowed key size for use within our key vaults is 2048 bits.

### Keys should not be active for longer than the specified number of days (optional)

We need to specify the number of days that a key should be active. Keys that are used for an extended period of time increase the probability that an attacker could compromise the key. As a good security practice, we need to make sure that keys have not been active for longer than two years.

### Certificates using RSA cryptography should have the specified minimum key size (enforce)

This allows us to manage XYZ’s CSF-PKI compliance requirements by specifying a minimum key size of 2048 bits for RSA certificates stored in the key vaults.

## Compliance and remediation

Key Vaults that are non-compliant to policies with **deployIfNotExists** or **modify** effect can be put into a compliant state through Remediation. Remediation is accomplished through remediation tasks that deploy the **deployIfNotExists** template or the **modify** operations of the assigned policy on a given Key Vault.

Key vaults owners can view the compliance report on the Azure Portal and identify non-compliant policy assignments related to their vault.

It’s the responsibility of the key vault owners to remediate the non-compliant policy assignments on their own.

# Appendix

## Glossary

|  |  |
| --- | --- |
| Name | Definition |
| Tenant | A tenant is the organization that owns and manages a specific instance of Microsoft cloud services. It's most often used to refer to the set of Azure and Microsoft 365 services for an organization. |
| Vault owner | A vault owner can create a key vault and gain full access and control over it. The vault owner can also set up auditing to log who accesses secrets and keys. Administrators can control the key lifecycle. They can roll to a new version of the key, back it up, and do related tasks. |
| Vault consumer | A vault consumer can perform actions on the assets inside the key vault when the vault owner grants the consumer access. The available actions depend on the permissions granted. |
| Managed HSM Administrators | Users who are assigned the Administrator role have complete control over a Managed HSM pool. They can create more role assignments to delegate controlled access to other users. |
| Managed HSM Crypto Officer/User | Built-in roles that are usually assigned to users or service principals that will perform cryptographic operations using keys in Managed HSM. Crypto User can create new keys but cannot delete keys. |
| Resource | A resource is a manageable item that's available through Azure. Common examples are virtual machine, storage account, web app, database, and virtual network. There are many more. |
| Resource group | A resource group is a container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups, based on what makes the most sense for the organization. |
| Security principal | An Azure security principal is a security identity that user-created apps, services, and automation tools use to access specific Azure resources. Think of it as a "user identity" (username and password or certificate) with a specific role, and tightly controlled permissions. A security principal should only need to do specific things, unlike a general user identity. It improves security if you grant it only the minimum permission level that it needs to perform its management tasks. A security principal used with an application or service is specifically called a service principal. |
| Azure Active Directory (Azure AD) | Azure AD is the Active Directory service for a tenant. Each directory has one or more domains. A directory can have many subscriptions associated with it, but only one tenant. |
| Azure tenant ID | A tenant ID is a unique way to identify an Azure AD instance within an Azure subscription. |
| Managed identities | Azure Key Vault provides a way to securely store credentials and other keys and secrets, but your code needs to authenticate to Key Vault to retrieve them. Using a managed identity makes solving this problem simpler by giving Azure services an automatically managed identity in Azure AD. You can use this identity to authenticate to Key Vault or any service that supports Azure AD authentication, without having any credentials in your code. For more information, see the following image and the overview of managed identities for Azure resources. |
| Caller | A user or application that makes calls to the key vault to obtain objects or to manage the key vault |

## Acronym & Abbreviation

|  |  |
| --- | --- |
| Name | Definition |
| HSM | Hardware Security Module |
| PKI | Public Key Infrastructure |
| SIEM | Security Information and Event Management |
| RBAC | Role-Based Access Control |
| PCFv1 | Public Cloud Framework – version 1 |