100 XP

Introduction

3 minutes

Azure Key Vault is a cloud service for securely storing and accessing secrets. A secret is anything that you want to tightly control access to, such as API keys, passwords, certificates, or cryptographic keys.

After completing this module, you'll be able to:

- Describe the benefits of using Azure Key Vault
- Explain how to authenticate to Azure Key Vault
- Set and retrieve a secret from Azure Key Vault by using the Azure CLI

Next unit: Explore Azure Key Vault

Continue

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Explore Azure Key Vault

3 minutes

The Azure 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. Managed HSM pools only support HSM-backed keys.

Azure Key Vault helps solve the following problems:

- **Secrets Management:** Azure Key Vault can be used to Securely store and tightly control access to tokens, passwords, certificates, API keys, and other secrets
- **Key Management:** Azure Key Vault can also be used as a Key Management solution. Azure Key Vault makes it easy to create and control the encryption keys used to encrypt your data.
- **Certificate Management:** Azure Key Vault is also a service that lets you easily provision, manage, and deploy public and private Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with Azure and your internal connected resources.

Azure Key Vault has two service tiers: Standard, which encrypts with a software key, and a Premium tier, which includes hardware security module(HSM)-protected keys. To see a comparison between the Standard and Premium tiers, see the Azure Key Vault pricing page .

Key benefits of using Azure Key Vault

- **Centralized application secrets:** Centralizing storage of application secrets in Azure Key Vault allows you to control their distribution. For example, instead of storing the connection string in the app's code you can store it securely in Key Vault. Your applications can securely access the information they need by using URIs. These URIs allow the applications to retrieve specific versions of a secret.
- **Securely store secrets and keys:** Access to a key vault requires proper authentication and authorization before a caller (user or application) can get access. Authentication is done via Azure Active Directory. Authorization may be done via Azure role-based access control (Azure RBAC) or Key Vault access policy. Azure RBAC is used when dealing with the

management of the vaults and key vault access policy is used when attempting to access data stored in a vault. Azure Key Vaults may be either software-protected or, with the Azure Key Vault Premium tier, hardware-protected by hardware security modules (HSMs).

- **Monitor access and use:** You can monitor activity by enabling logging for your vaults. You have control over your logs and you may secure them by restricting access and you may also delete logs that you no longer need. Azure Key Vault can be configured to:
  - Archive to a storage account.
  - Stream to an event hub.
  - Send the logs to Azure Monitor logs.

- **Simplified administration of application secrets:** Security information must be secured, it must follow a life cycle, and it must be highly available. Azure Key Vault simplifies the process of meeting these requirements by:
  - Removing the need for in-house knowledge of Hardware Security Modules
  - Scaling up on short notice to meet your organization's usage spikes.
  - Replicating the contents of your Key Vault within a region and to a secondary region. Data replication ensures high availability and takes away the need of any action from the administrator to trigger the failover.
  - Providing standard Azure administration options via the portal, Azure CLI and PowerShell.
  - Automating certain tasks on certificates that you purchase from Public CAs, such as enrollment and renewal.

Next unit: Discover Azure Key Vault best practices

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Discover Azure Key Vault best practices

3 minutes

Azure Key Vault is a tool for securely storing and accessing secrets. A secret is anything that you want to tightly control access to, such as API keys, passwords, or certificates. A vault is logical group of secrets.

Authentication

To do any operations with Key Vault, you first need to authenticate to it. There are three ways to authenticate to Key Vault:

- **Managed identities for Azure resources**: When you deploy an app on a virtual machine in Azure, you can assign an identity to your virtual machine that has access to Key Vault. You can also assign identities to other Azure resources. The benefit of this approach is that the app or service isn't managing the rotation of the first secret. Azure automatically rotates the service principal client secret associated with the identity. We recommend this approach as a best practice.
- **Service principal and certificate**: You can use a service principal and an associated certificate that has access to Key Vault. We don't recommend this approach because the application owner or developer must rotate the certificate.
- **Service principal and secret**: Although you can use a service principal and a secret to authenticate to Key Vault, we don't recommend it. It's hard to automatically rotate the bootstrap secret that's used to authenticate to Key Vault.

Encryption of data in transit

Azure Key Vault enforces Transport Layer Security (TLS) protocol to protect data when it's traveling between Azure Key Vault and clients. Clients negotiate a TLS connection with Azure Key Vault. TLS provides strong authentication, message privacy, and integrity (enabling detection of

message tampering, interception, and forgery), interoperability, algorithm flexibility, and ease of deployment and use.

Perfect Forward Secrecy (PFS) protects connections between customers' client systems and Microsoft cloud services by unique keys. Connections also use RSA-based 2,048-bit encryption key lengths. This combination makes it difficult for someone to intercept and access data that is in transit.

Azure Key Vault best practices

- **Use separate key vaults:** Recommended to use a vault per application per environment (Development, Pre-Production and Production). This helps you not share secrets across environments and also reduces the threat in case of a breach.
- **Control access to your vault:** Key Vault data is sensitive and business critical, you need to secure access to your key vaults by allowing only authorized applications and users.
- **Backup:** Create regular back ups of your vault on update/delete/create of objects within a Vault.
- **Logging:** Be sure to turn on logging and alerts.
- **Recovery options:** Turn on soft-delete and purge protection if you want to guard against force deletion of the secret.

Next unit: Authenticate to Azure Key Vault

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Authenticate to Azure Key Vault

3 minutes

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

For applications, there are two ways to obtain a service principal:

- Enable a system-assigned **managed identity** for the application. With managed identity, Azure internally manages the application's service principal and automatically authenticates the application with other Azure services. Managed identity is available for applications deployed to a variety of services.
- If you cannot use managed identity, you instead register the application with your Azure AD tenant. Registration also creates a second application object that identifies the app across all tenants.

> **Note**
>
> It is recommended to use a system-assigned managed identity.

Below is information on authenticating to Key Vault without using a managed identity.

Authentication to Key Vault in application code

Key Vault SDK is using Azure Identity client library, which allows seamless authentication to Key Vault across environments with same code. The table below provides information on the Azure Identity client libraries:

.NET	Python	Java	JavaScript
Azure Identity SDK .NET	Azure Identity SDK Python	Azure Identity SDK Java	Azure Identity SDK JavaScript

Authentication to Key Vault with REST

Access tokens must be sent to the service using the HTTP Authorization header:

HTTP Copy

```
PUT /keys/MYKEY?api-version=<api_version>  HTTP/1.1
Authorization: Bearer <access_token>
```

When an access token is not supplied, or when a token is not accepted by the service, an HTTP 401 error will be returned to the client and will include the `WWW-Authenticate` header, for example:

HTTP Copy

```
401 Not Authorized
WWW-Authenticate: Bearer authorization="…", resource="…"
```

The parameters on the `WWW-Authenticate` header are:

- authorization: The address of the OAuth2 authorization service that may be used to obtain an access token for the request.
- resource: The name of the resource (`https://vault.azure.net`) to use in the authorization request.

Additional resources

- Azure Key Vault developer's guide
- Azure Key Vault availability and redundancy

Next unit: Exercise: Set and retrieve a secret from Azure Key Vault by using Azure CLI

Continue

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Exercise: Set and retrieve a secret from Azure Key Vault by using Azure CLI

3 minutes

In this exercise you'll learn how to perform the following actions by using the Azure CLI:

- Create a Key Vault
- Add and retrieve a secret

Prerequisites

- An **Azure account** with an active subscription. If you don't already have one, you can sign up for a free trial at https://azure.com/free

Log in to Azure and start the Cloud Shell

1. Log in to the Azure portal and open the Cloud Shell.

2. After the shell opens be sure to select the **Bash** environment.

Create a Key Vault

1. Let's set some variables for the CLI commands to use to reduce the amount of retyping. Replace the `<myLocation>` variable string below with a region that makes sense for you. The Key Vault name needs to be a globally unique name, and the script below generates a random string.

Bash Copy

```
myKeyVault=az204vault-$RANDOM
myLocation=<myLocation>
```

2. Create a resource group.

Azure CLI Copy

```
az group create --name az204-vault-rg --location $myLocation
```

3. Create a Key Vault by using the `az keyvault create` command.

Azure CLI Copy

```
az keyvault create --name $myKeyVault --resource-group az204-vault-rg --location
$myLocation
```

> **Note**
>
> This can take a few minutes to run.

Add and retrieve a secret

To add a secret to the vault, you just need to take a couple of additional steps.

1. Create a secret. Let's add a password that could be used by an app. The password will be called **ExamplePassword** and will store the value of **hVFkk965BuUv** in it.

Azure CLI Copy

```
az keyvault secret set --vault-name $myKeyVault --name "ExamplePassword" --value
"hVFkk965BuUv"
```

2. Use the `az keyvault secret show` command to retrieve the secret.

Azure CLI Copy

```
az keyvault secret show --name "ExamplePassword" --vault-name $myKeyVault
```

This command will return some JSON. The last line will contain the password in plain text.

JSON Copy

```
"value": "hVFkk965BuUv"
```

You have created a Key Vault, stored a secret, and retrieved it.

Clean up resources

When you no longer need the resources in this exercise use the following command to delete the resource group and associated Key Vault.

Azure CLI Copy

```
az group delete --name az204-vault-rg --no-wait
```

Next unit: Knowledge check

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

200 XP

Knowledge check

3 minutes

Check your knowledge

1. Which of the below methods of authenticating to Azure Key Vault is recommended for most scenarios?

- (●) Service principal and certificate

  ✗ That's incorrect. This method is not recommended because it is difficult to automatically rotate the bootstrap secret that's used to authenticate to Key Vault.

- ( ) Service principal and secret
- ( ) Managed identities

  ✓ That's correct. The benefit of this approach is that Azure automatically rotates the identity.

2. Azure Key Vault protects data when it is traveling between Azure Key Vault and clients. What protocol does it use for encryption?

- (●) Secure Sockets Layer

  ✗ That's incorrect. The Secure Sockets Layer protocol has been replaced with the Transport Layer Security protocol.

- ( ) Transport Layer Security

  ✓ That's correct. Azure Key Vault enforces Transport Layer Security protocol to protect data when it's traveling between Azure Key Vault and clients.

- ( ) Presentation Layer

Next unit: Summary

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Summary

3 minutes

In this module, you learned how to:

- Describe the benefits of using Azure Key Vault
- Explain how to authenticate to Azure Key Vault
- Set and retrieve a secret from Azure Key Vault by using the Azure CLI

Module complete:

Unlock achievement

How are we doing? ☆ ☆ ☆ ☆ ☆