**Azure vault-** Azure Key Vault is a cloud service that securely stores and manages sensitive information such as secrets, keys, and certificates. It enables you to safeguard cryptographic keys and secrets used by cloud applications and services, ensuring that this sensitive data is protected and access is controlled.

Key Vault offers features including:

* **Secrets Management**: Securely store and control access to tokens, passwords, certificates, API keys, and other secrets.
* **Key Management**: Create, import, and manage cryptographic keys used for data encryption.
* **Certificate Management**: Provision, manage, and deploy public and private SSL/TLS certificates for your Azure and internal connected resources.
* **Hardware Security Modules (HSMs)**: Use FIPS 140-2 Level 2 validated HSMs to protect your keys.

**Step-by-Step Guide to Practicing Azure Key Vault**

Azure Key Vault is a cloud service that provides a secure environment for storing and managing sensitive information like secrets, keys, and certificates. Follow the steps below to set up and practice using Azure Key Vault:

**Prerequisites**

1. **Azure Subscription**: Ensure you have an active Azure subscription.
2. **Azure CLI**: Install the Azure CLI on your local machine ([download here](https://learn.microsoft.com/en-us/cli/azure/install-azure-cli)).
3. **Azure Portal Access**: Familiarize yourself with navigating the Azure portal ([access here](https://portal.azure.com)).

**Step 1: Set Up Azure Key Vault**

**Using Azure Portal**

1. Sign in to the [Azure Portal](https://portal.azure.com).
2. Navigate to **Create a resource** > **Security + Identity** > **Key Vault**.
3. Fill in the required details:
   * **Subscription**: Select your subscription.
   * **Resource Group**: Create a new resource group or select an existing one.
   * **Key Vault Name**: Enter a unique name for your Key Vault.
   * **Region**: Choose a region close to your location.
   * **Pricing Tier**: Select the desired pricing tier (Standard or Premium).
4. Click **Review + Create** and then **Create**.

**Using Azure CLI**

1. Open a terminal and log in to Azure:
2. az login
3. Create a resource group:
4. az group create --name MyResourceGroup --location eastus
5. Create a Key Vault:
6. az keyvault create --name MyKeyVault --resource-group MyResourceGroup --location eastus

**Step 2: Add Secrets, Keys, and Certificates**

**1) Add a Secret**

**Azure Portal**

1. Go to your Key Vault in the Azure Portal.
2. Select **Secrets** > **Generate/Import**.
3. Enter the secret details (e.g., name and value).
4. Click **Create**.

**Azure CLI**

az keyvault secret set --vault-name MyKeyVault --name MySecretName --value MySecretValue

**2) Add a Key**

**Azure Portal**

1. Select **Keys** > **Generate/Import**.
2. Choose the key type (e.g., RSA or EC) and key size.
3. Click **Create**.

**Azure CLI**

az keyvault key create --vault-name MyKeyVault --name MyKeyName --protection software

**Add a Certificate**

**Azure Portal**

1. Select **Certificates** > **Generate/Import**.
2. Configure the certificate policy and settings.
3. Click **Create**.

**Azure CLI**

az keyvault certificate create --vault-name MyKeyVault --name MyCertName --policy @policy.json

(Note: Replace @policy.json with the path to your certificate policy file.)

**Step 3: Access and Manage Stored Items**

**1) Access a Secret**

**Azure CLI**

az keyvault secret show --vault-name MyKeyVault --name MySecretName

**Access a Key**

**Azure CLI**

az keyvault key show --vault-name MyKeyVault --name MyKeyName

**2) Access a Certificate**

**Azure CLI**

az keyvault certificate show --vault-name MyKeyVault --name MyCertName

**Step 4: Implement Access Policies**

1. In the Azure Portal, navigate to your Key Vault.
2. Select **Access policies** > **Add Access Policy**.
3. Configure the permissions for the desired user or application.
4. Save the changes.

**Step 5: Integrate Azure Key Vault with Applications**

**Using Azure SDKs**

1. Install the Azure SDK for your programming language (e.g., Python, .NET, or Java).
2. Authenticate using Azure credentials.
3. Use the SDK to access secrets, keys, and certificates programmatically.

**Example (Python):**

from azure.identity import DefaultAzureCredential

from azure.keyvault.secrets import SecretClient

# Set up the client

vault\_url = "https://MyKeyVault.vault.azure.net"

credential = DefaultAzureCredential()

client = SecretClient(vault\_url=vault\_url, credential=credential)

# Retrieve a secret

retrieved\_secret = client.get\_secret("MySecretName")

print(f"Secret Value: {retrieved\_secret.value}")

**Step 6: Monitor and Audit Key Vault**

1. In the Azure Portal, navigate to your Key Vault.
2. Select **Monitoring** > **Insights**.
3. Review metrics like secret usage, key access, and certificate status.
4. Enable diagnostic logs for detailed tracking.

**Step 7: Clean Up Resources**

To avoid unnecessary charges, delete the resource group when done:

az group delete --name MyResourceGroup --yes --no-wait