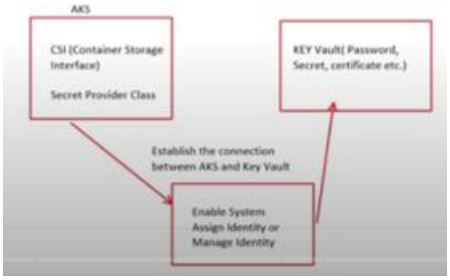
# Key Vault Integration In AKS

## Method One: Securing Secrets with Azure Key Vault Using Managed Identity

- **Enable the CSI Driver In AKS**
- **Create Key Vault**
- **Enable Key Vault Add-on In AKS**
- **Enable System Assign Identity or Managed Identity in AKS**
- **Create Secret provide Class and Pod deployment in AKS**
- **Verify status in AKS**



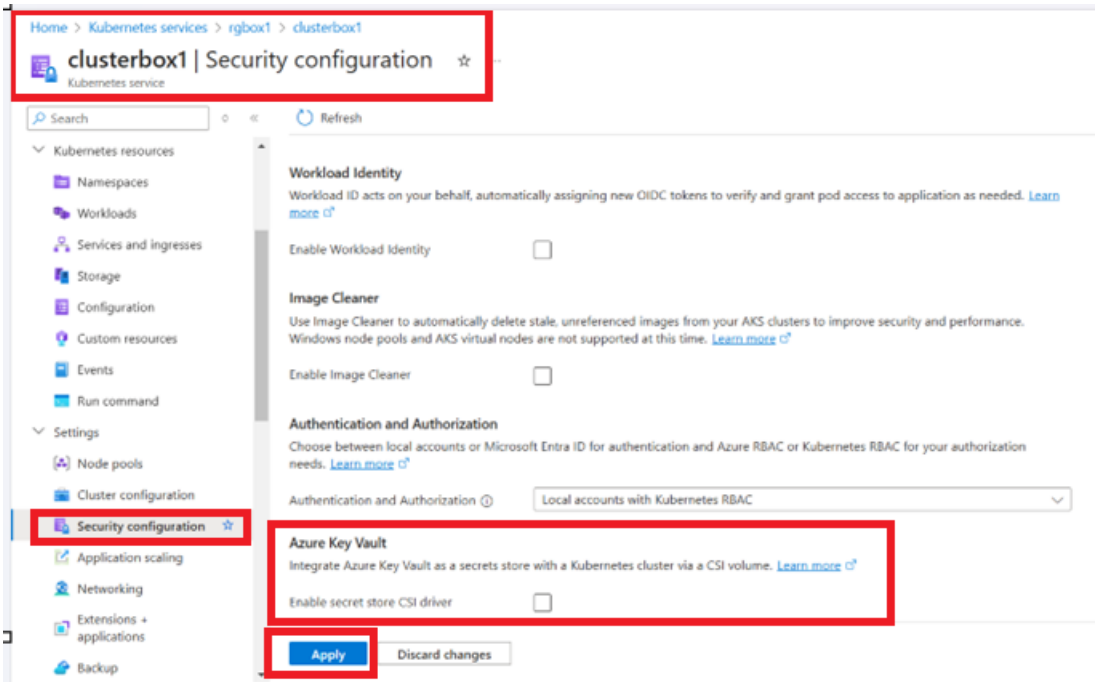

## Step1: Enabling the Secret Store CSI Driver in AKS

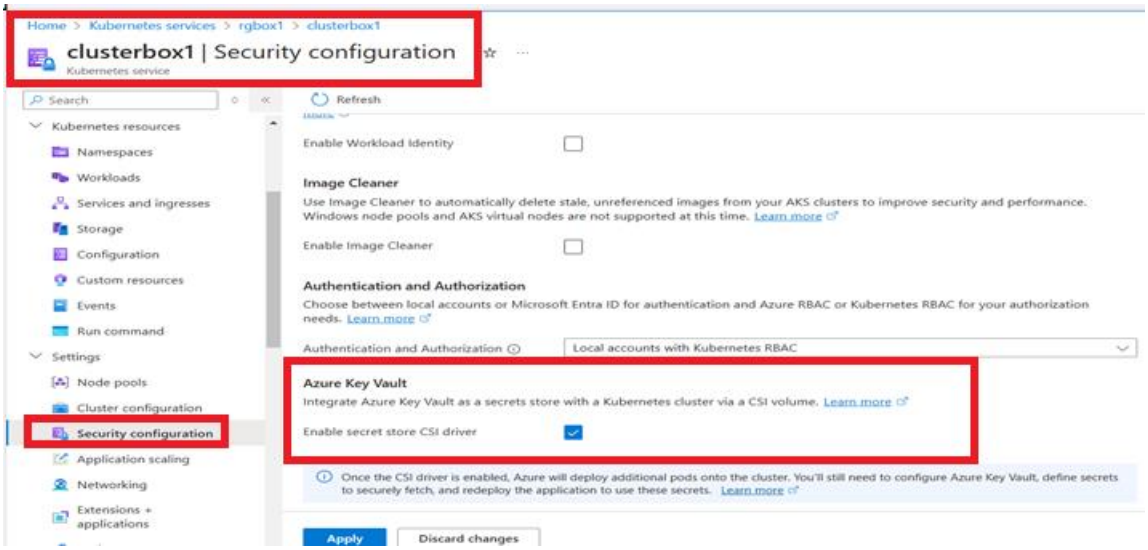
### CLI Method to Enable the CSI Driver in AKS

az aks update --name clusterbox1 --resource-group rgbox1 --enable-disk-driver --enable-file-driver --enable-blob-driver --enable-snapshot-controller

az aks update --name clusterbox1 --resource-group rgbox1 --enable-secret-store-csi-driver

### GUI Method to Enable the CSI Driver in AKS






## Step2: Verify Installation in AKS

kubectl get pods -n kube-system

```
PS F:\> kubectl get csidrivers
NAME                  ATTACHREQUIRED  PODINFOONMOUNT  STORAGECAPACITY  TOKENREQUESTS                REQUIRESREPUBLISH  MODES                AGE
blob.csi.azure.com    false           true            false            api://AzureADTokenExchange   false              Persistent,Ephemeral 100m
disk.csi.azure.com    true            false           false            <unset>                      false              Persistent           4h47m
file.csi.azure.com    false           true            false            api://AzureADTokenExchange   false              Persistent,Ephemeral 4h47m
PS F:\>
```

## Step3: Set Up Azure Key Vault

### i. Create an Azure Key Vault

**CLI Method to Create Azure Key Vault in AKS**

az keyvault create --name satkeyvault --resource-group rgbox1 --location eastus

Home > Key vaults >

# Create a key vault ...

Create new

Instance details

Key vault name * arkeyvaultq

Region * East US

Pricing tier * Standard

Recovery options

Soft delete protection will automatically be enabled on this key vault. This feature allows you to recover or permanently delete a key vault and secrets for the duration of the retention period. This protection applies to the key vault and the secrets stored within the key vault.

To enforce a mandatory retention period and prevent the permanent deletion of key vaults or secrets prior to the retention period elapsing, you can turn on purge protection. When purge protection is enabled, secrets cannot be purged by users or by Microsoft.

Soft-delete Enabled

Days to retain deleted vaults * 90

Purge protection
- (•) Disable purge protection (allow key vault and objects to be purged during retention period)
- ( ) Enable purge protection (enforce a mandatory retention period for deleted

Previous Next Review + create

Home > Key vaults >

## Create a key vault

Grant data plane access by using a Azure RBAC or Key Vault access policy

- ( ) Azure role-based access control (recommended)
- (•) Vault access policy

**Resource access**

- [ ] Azure Virtual Machines for deployment
- [ ] Azure Resource Manager for template deployment
- [ ] Azure Disk Encryption for volume encryption

**Access policies**

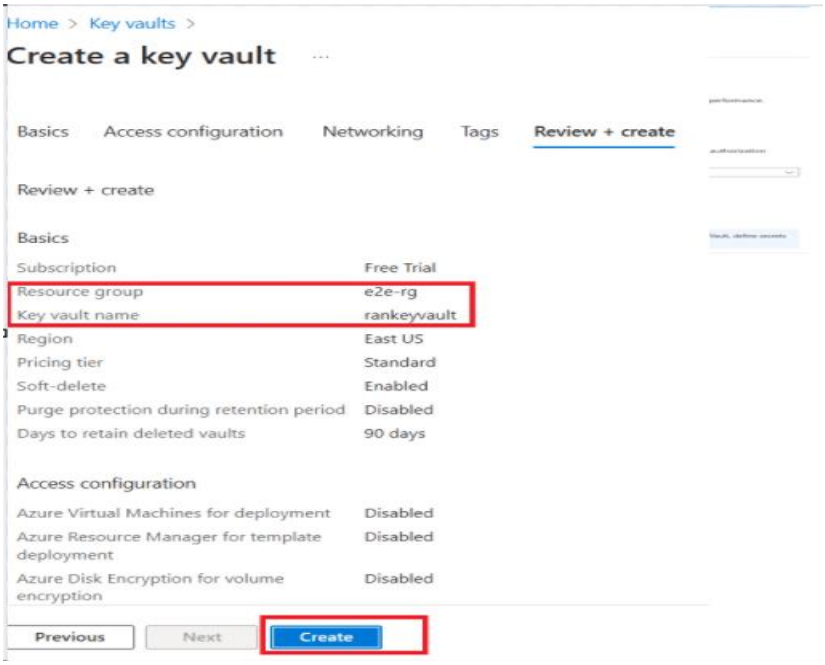
Access policies enable you to have fine grained control over access to vault items. Learn more

+ Create Edit Delete

| Name | Email | Key Permissions | Secret Permissions | Certificate Permissions |
|---|---|---|---|---|
| USER | | | | |
| Satish Ranjan | SatishRanjan@drgauravraj1993gmail.onmicros... | Get, List, Update, Create, Import, Delete, Recov... | Get, List, Set, Delete, Recover, Backup, Restore | Get, List, Update, Create, Import, Delete, Recov... |

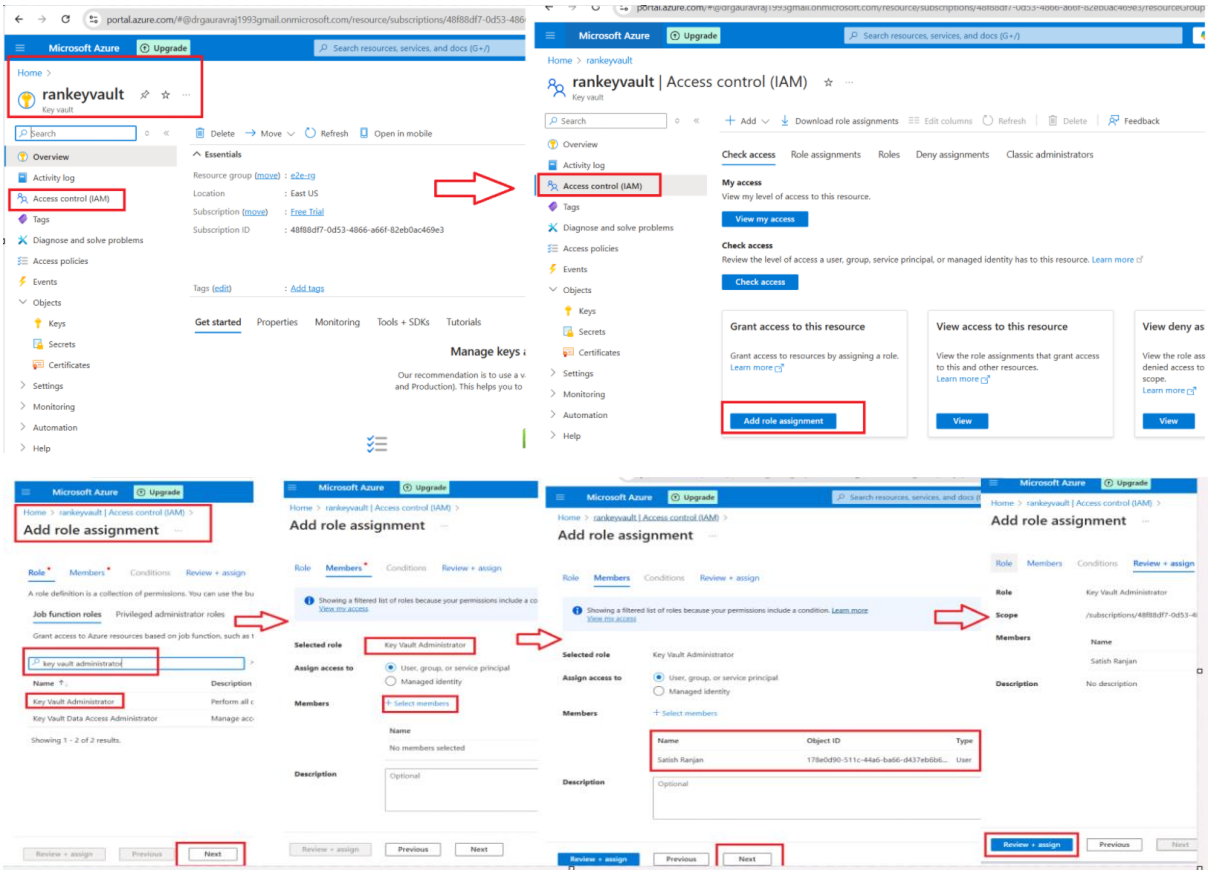
Previous Next Review + create

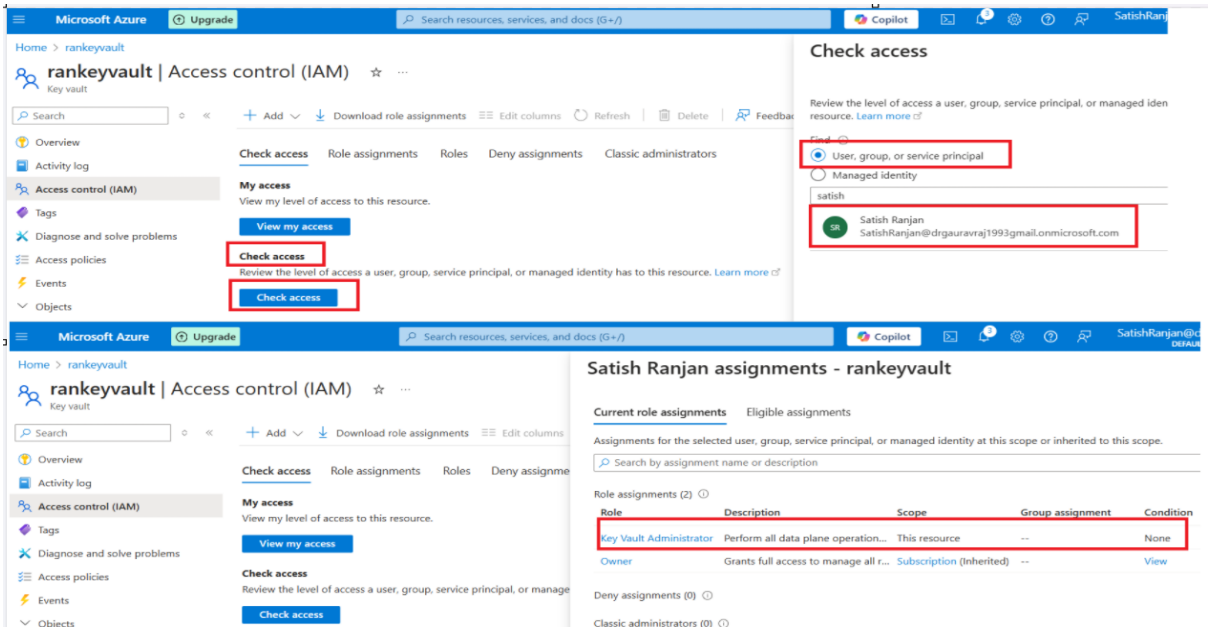
Give feedback




## ii. Key Vault Administrator role in Azure Key Vault's IAM (Identity and Access Management)

### GUI



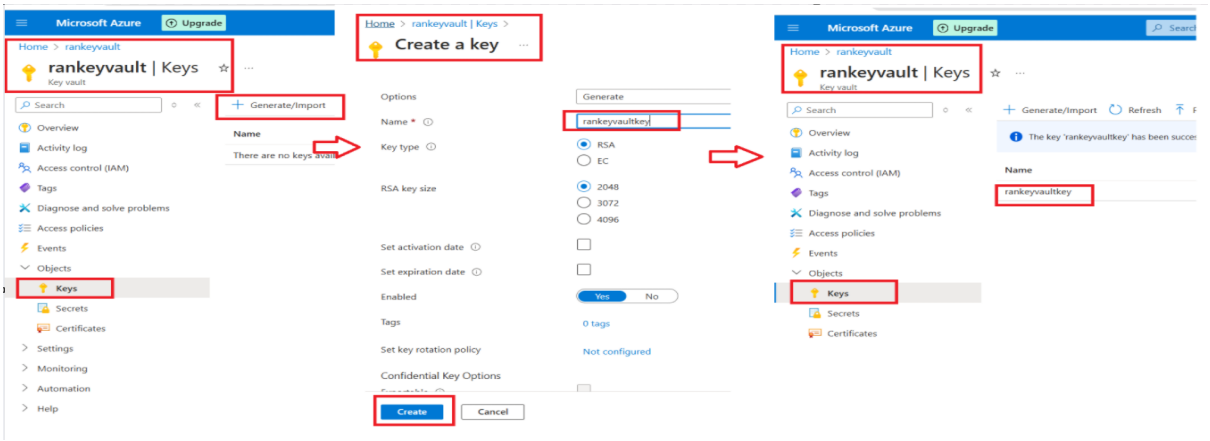


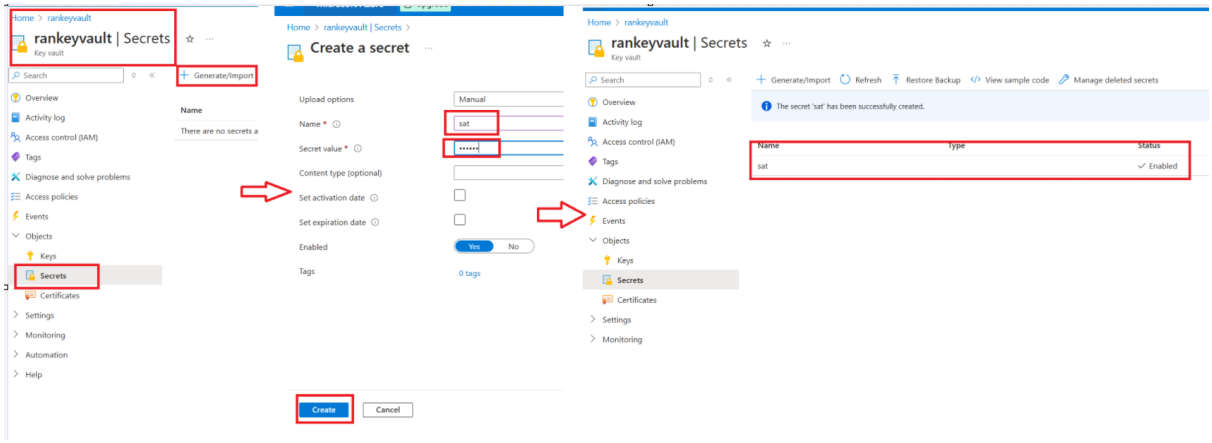

## iii. Add Secrets to the Vault

### CLI

az keyvault secret set --vault-name satkeyvault --name MySecret --value "SuperSecretValue"

### GUI






## iv. Enable KeyVault Add-on in AKS Or Enable CSI / Enable CSI Driver

**CLI**

az aks enable-addons --addons azure-keyvault-secrets-provider --name clusterbox1 --resource-group rgbox1

**To Verify**

kubectl get pod -n kube-system

```
PS F:\> kubectl get pod -n kube-system
NAME                                     READY   STATUS    RESTARTS   AGE
aks-secrets-store-csi-driver-mr5vr       3/3     Running   0          7m8s
aks-secrets-store-csi-driver-v98kz       3/3     Running   0          7m8s
aks-secrets-store-provider-azure-cmng4   1/1     Running   0          7m8s
aks-secrets-store-provider-azure-zcj68   1/1     Running   0          7m8s
azure-cns-2nldz                          1/1     Running   0          6h20m
azure-cns-bj74t                          1/1     Running   0          6h20m
```

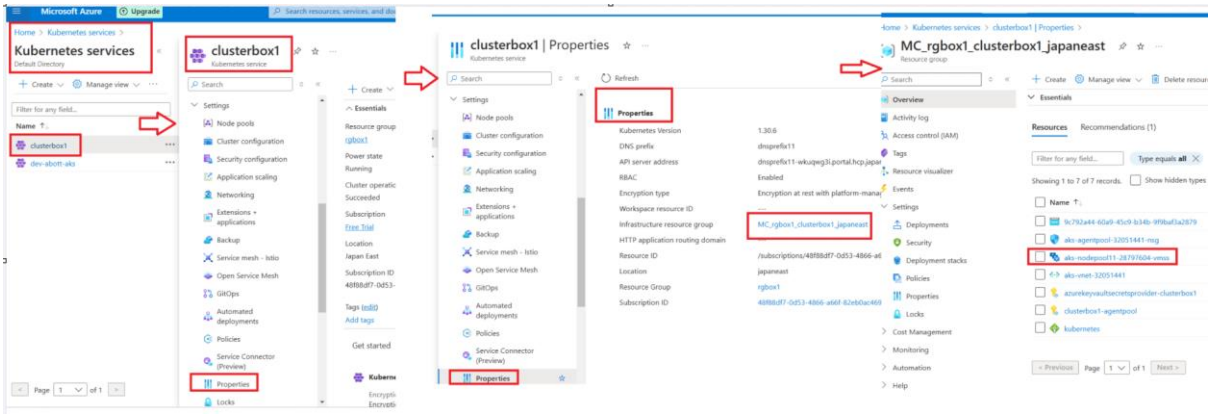
## Step4: Enable System Assign Identity or Managed Identity in AKS and Key Vault

## Permission of Managed Identity at AKS Side

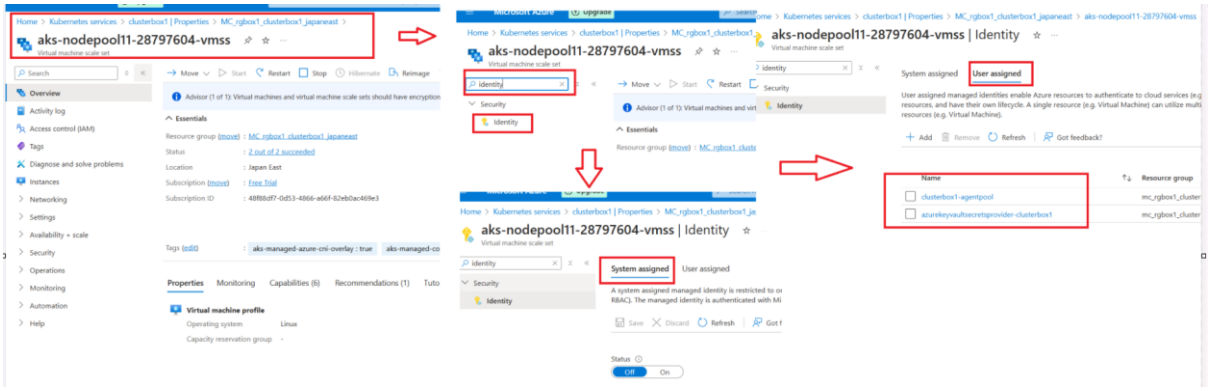
**CLI**

az aks update -g MyResourceGroup -n MyAKSCluster --assign-identity

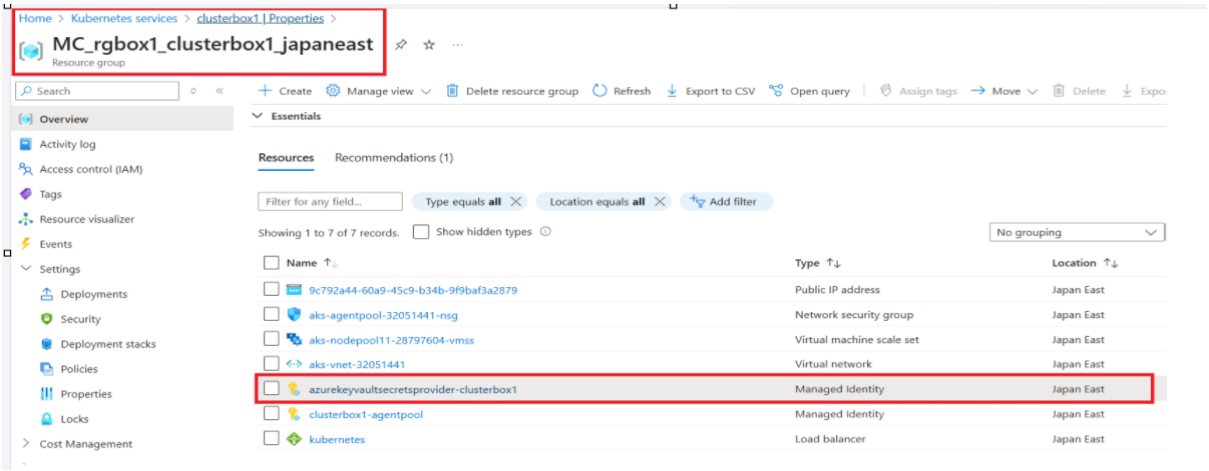
**GUI**



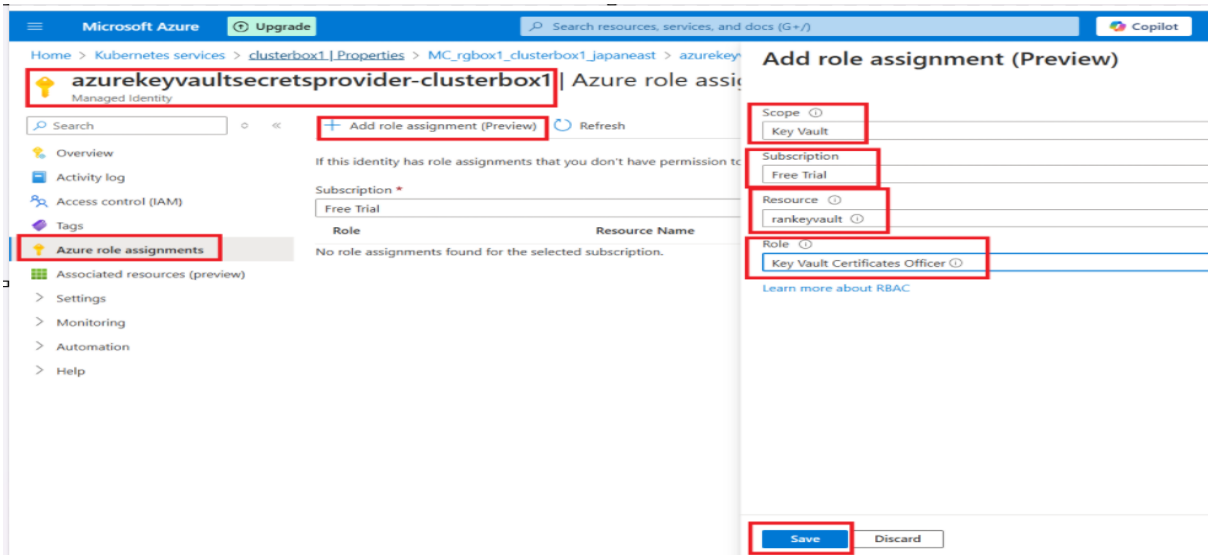
**Here in VMSS given system and user identity, but we need to use managed identity,**

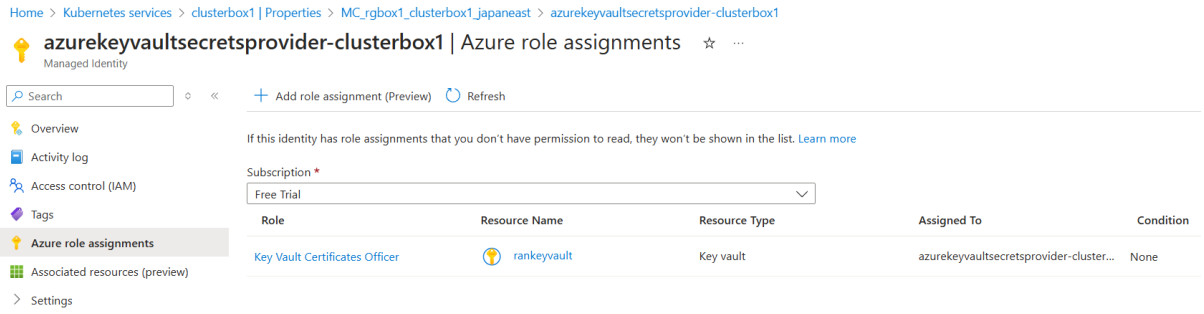


**But we will go with the Managed Identity, by default when enable the azure key vault provider in AKS , by default it created the managed identity.**

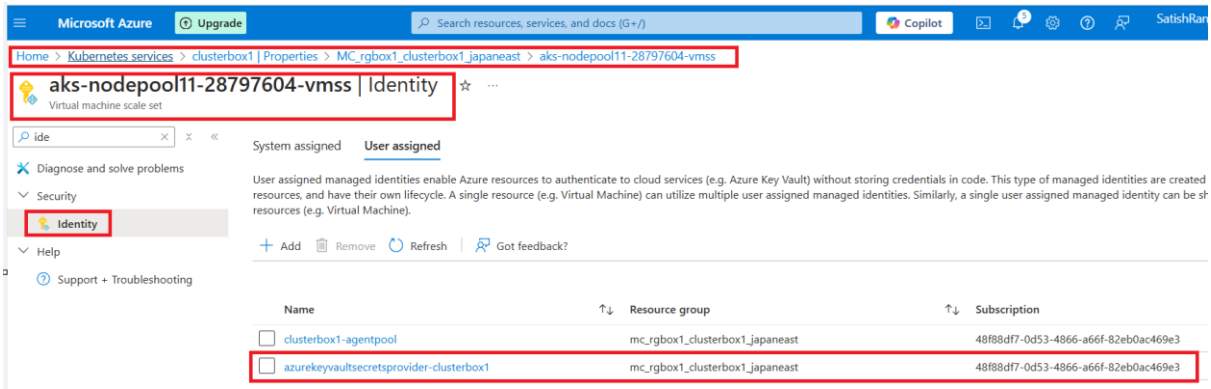



**Now Need to Assign Key vault certificate officer role for this managed identity.**




## To Verify

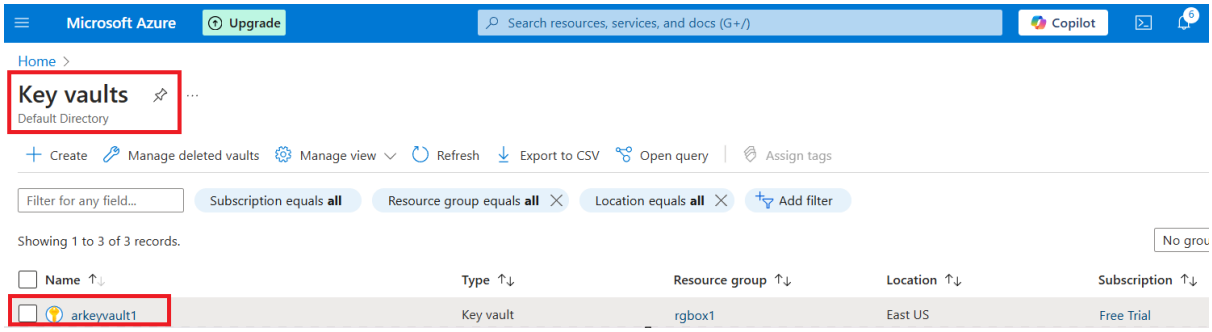


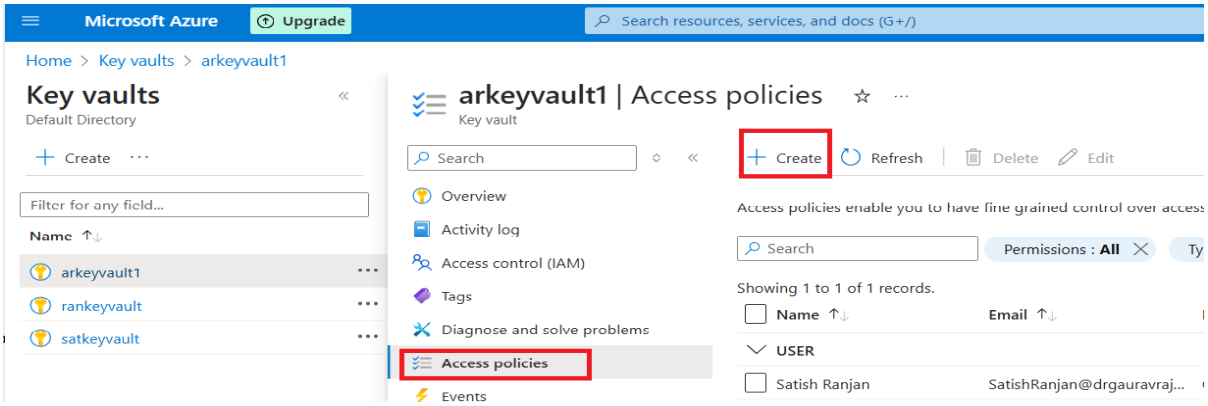

## Give Required Permission to Managed Identity at Key Vault Side

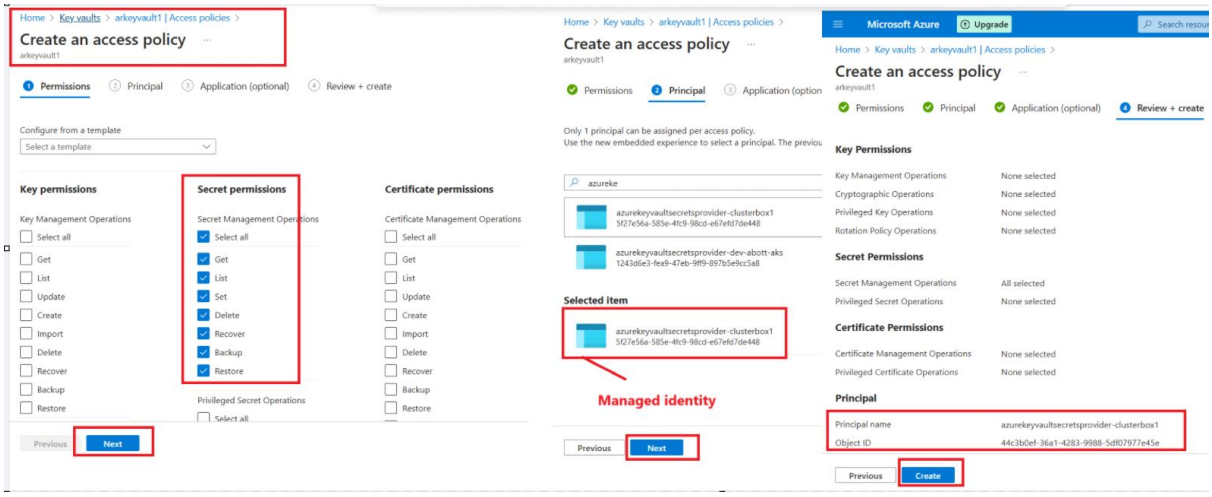
### CLI

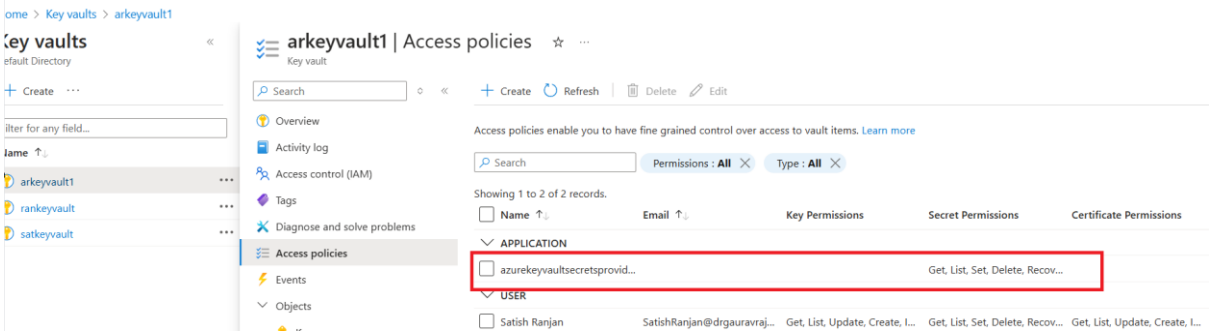
az keyvault set-policy -n MyKeyVault --secret-permissions get --spn <aks-identity-client-id>

### GUI




## Step5: Configure and Deploy to Kubernetes

1. **Create a SecretProviderClass for VM Managed Identity:**

**Yaml File for SecretProviderClass**

```yaml
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: azure-kvname-system-mi
  namespace: default
spec:
  parameters:
    cloudName: AzurePublicCloud
    keyvaultName: aksdemopg
    objects: |
      array:
        - |
          objectName: pgpassword
          objectType: secret        # object types: secret, key, or cert
          objectVersion: ""         # [OPTIONAL] object versions, default to
    tenantId: 4cd93cbc-b026-4c7d-86f1-76fcb7292573 # The Directory ID of the
    usePodIdentity: "false"
    useVMManagedIdentity: "true"
    userAssignedIdentityID: 3457145b-5926-4a60-8d5f-7bb929192891 #Manage iden
  provider: azure
```

2. **Deploy Your Application:**

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: busybox-secrets-store-inline-system-msi
  namespace: default
spec:
  containers:
    - name: busybox
      image: k8s.gcr.io/e2e-test-images/busybox:1.29-4
      command:
        - /bin/sleep
        - '10000'
      resources: {}
      volumeMounts:
        - name: secrets-store01-inline
          readOnly: true
          mountPath: /mnt/secrets-store
  volumes:
    - name: secrets-store01-inline
      csi:
        driver: secrets-store.csi.k8s.io
        readOnly: true
        volumeAttributes:
          secretProviderClass: azure-kvname-system-mi
```

**Step 5: Access Your Secrets**

```
PS C:\Users\NIRAV> kubectl exec busybox-secrets-store-inline-system-msi -- ls /mnt/secrets-store/
postgrespw
PS C:\Users\NIRAV> kubectl exec busybox-secrets-store-inline-system-msi -- cat /mnt/secrets-store/postgrespw
password
PS C:\Users\NIRAV>
```

**cat /mnt/secrets-store/MySecret**

## Method 2: Securing Secrets with Azure Key Vault Using User Assigned Identity

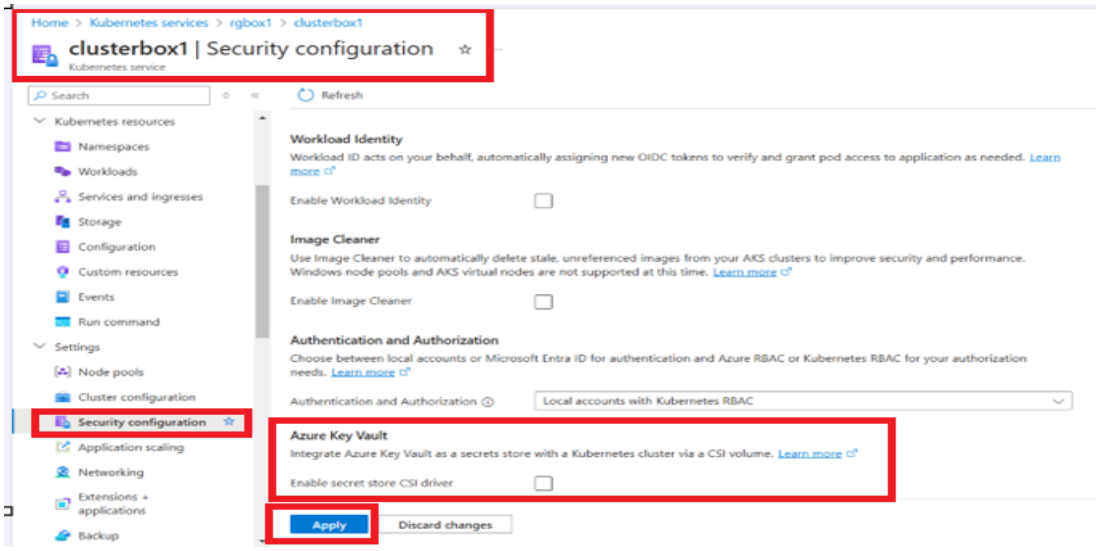
### Step1: Enabling the Secret Store CSI Driver in AKS

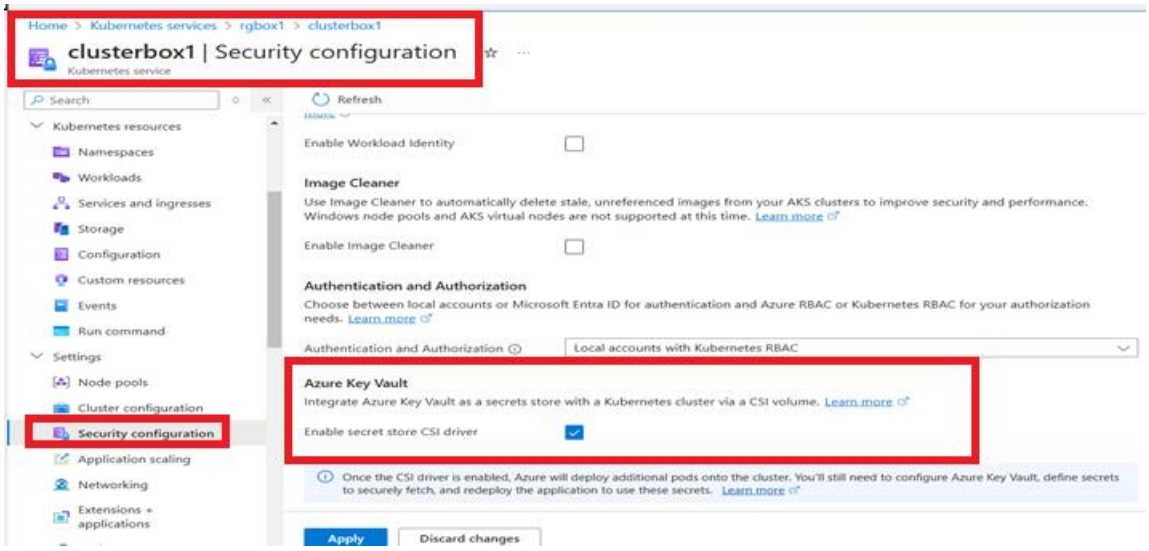
**CLI Method to Enable the CSI Driver in AKS**

az aks update --name clusterbox1 --resource-group rgbox1 --enable-disk-driver --enable-file-driver --enable-blob-driver --enable-snapshot-controller

az aks update --name clusterbox1 --resource-group rgbox1 --enable-secret-store-csi-driver

**GUI Method to Enable the CSI Driver in AKS**




### Step2: Verify Installation in AKS

kubectl get pods -n kube-system

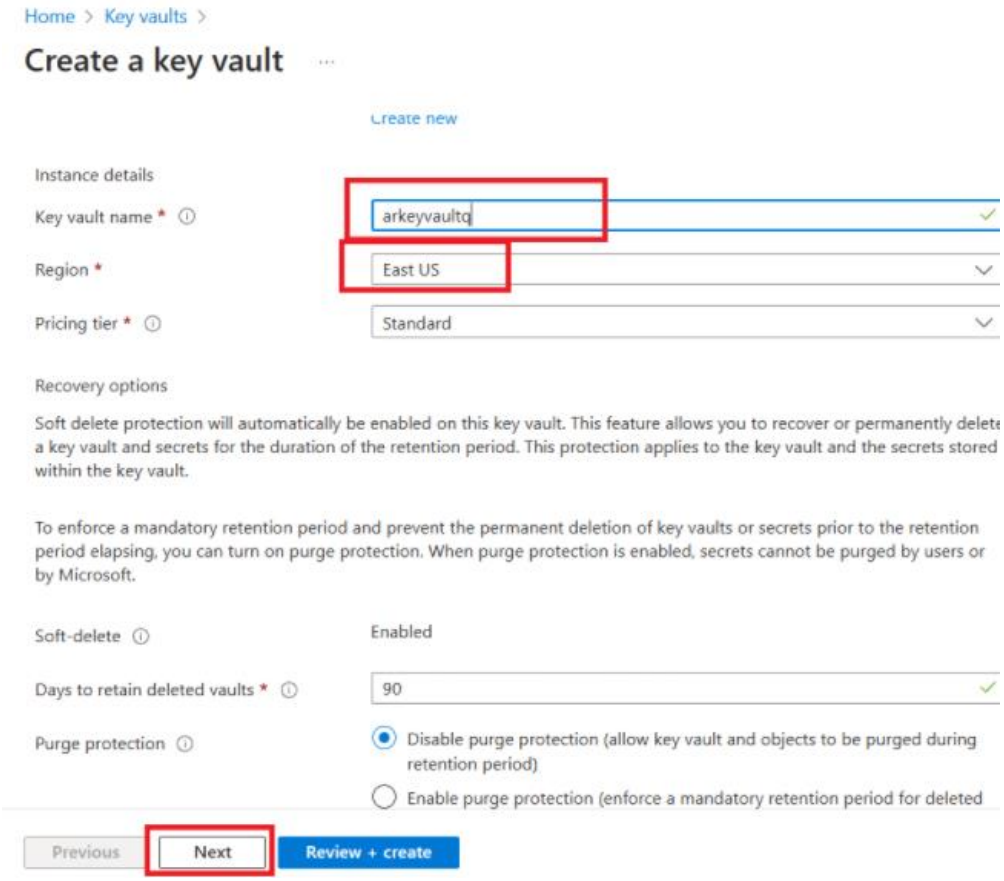
```
PS F:\> kubectl get csidrivers
NAME                  ATTACHREQUIRED  PODINFOONMOUNT  STORAGECAPACITY  TOKENREQUESTS                REQUIRESREPUBLISH  MODES                 AGE
blob.csi.azure.com    false           true            false            api://AzureADTokenExchange   false              Persistent,Ephemeral  100m
disk.csi.azure.com    true            false           false            <unset>                      false              Persistent            4h47m
file.csi.azure.com    false           true            false            api://AzureADTokenExchange   false              Persistent,Ephemeral  4h47m
PS F:\>
```

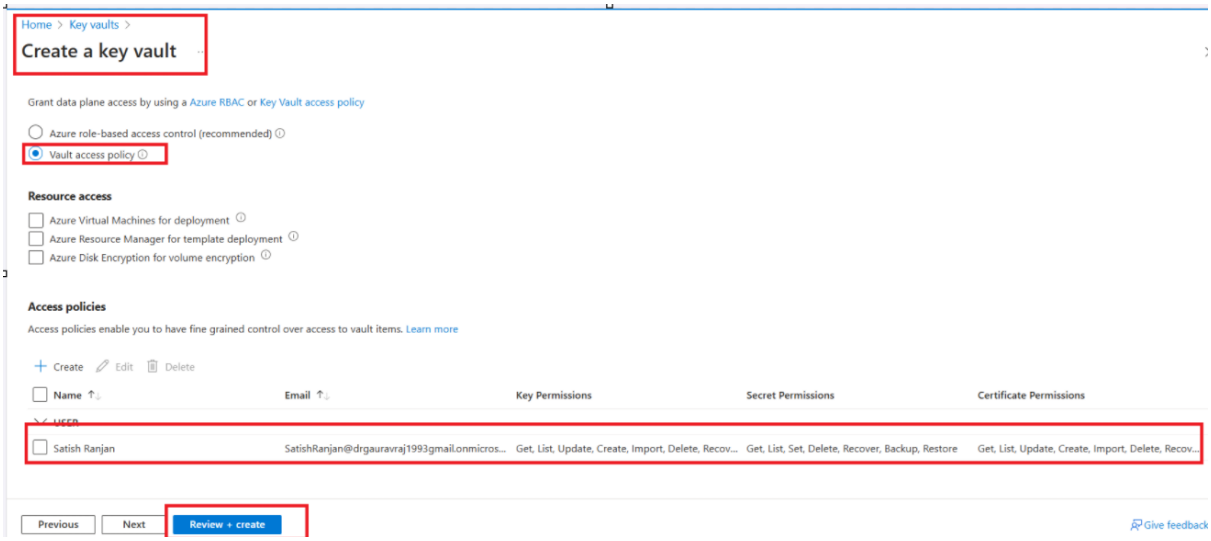
## Step3: Set Up Azure Key Vault

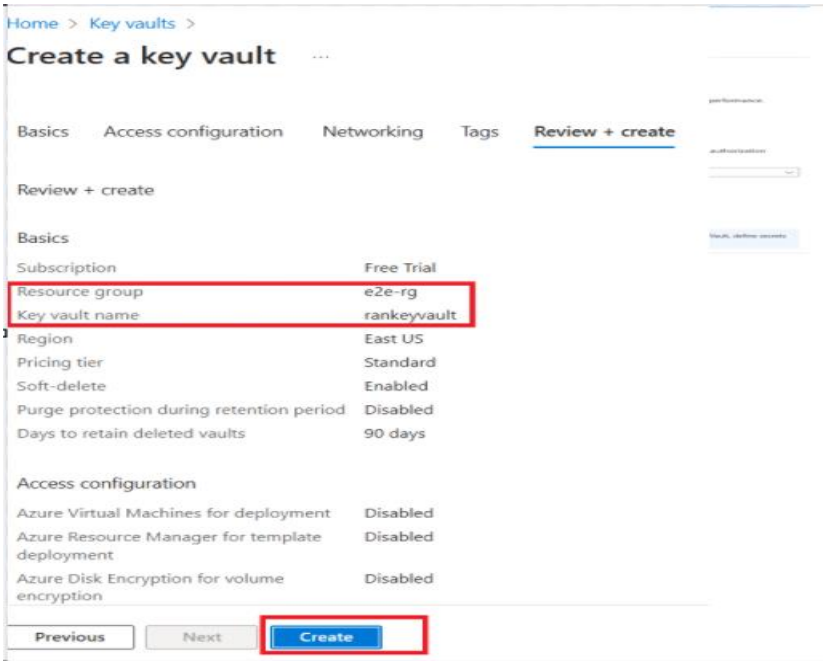
### i. Create an Azure Key Vault

**CLI Method to Create Azure Key Vault in AKS**

az keyvault create --name satkeyvault --resource-group rgbox1 --location eastus

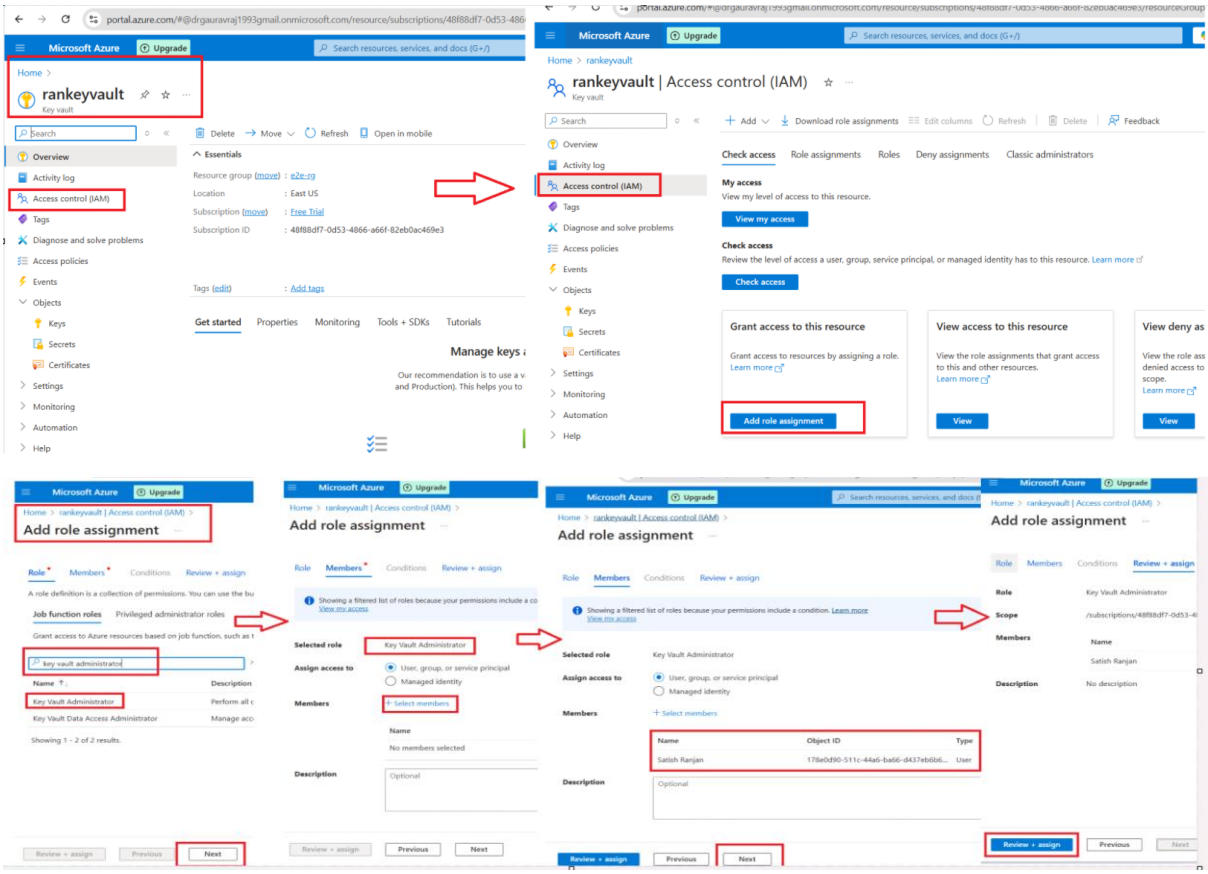


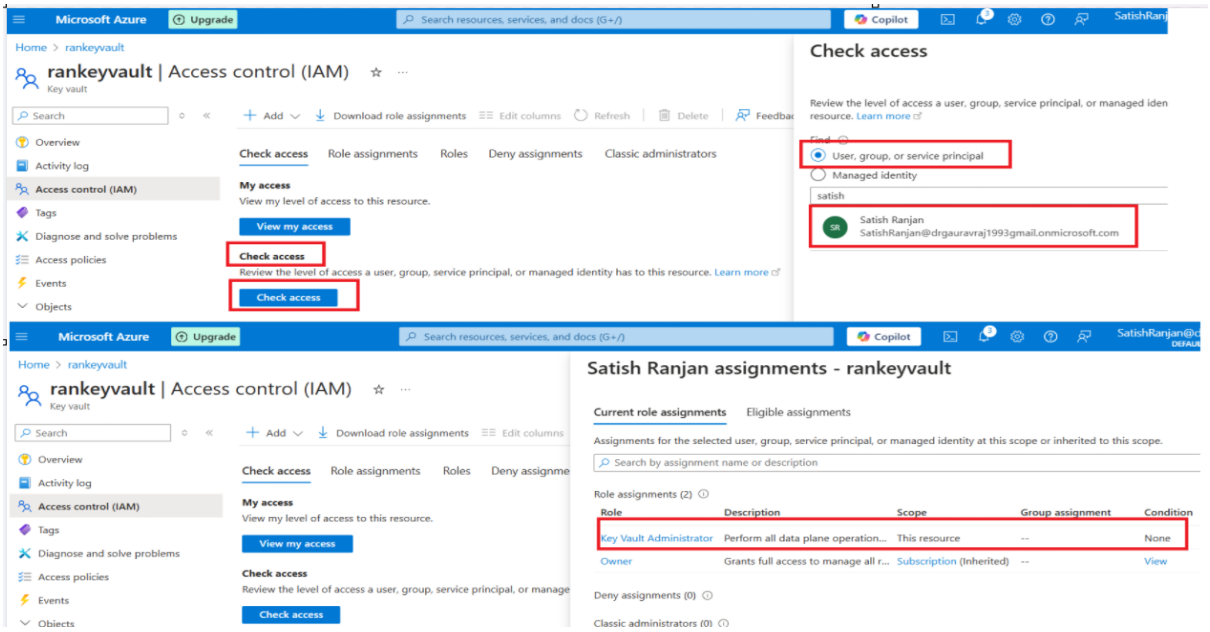






## ii. Key Vault Administrator role in Azure Key Vault's IAM (Identity and Access Management)

### GUI



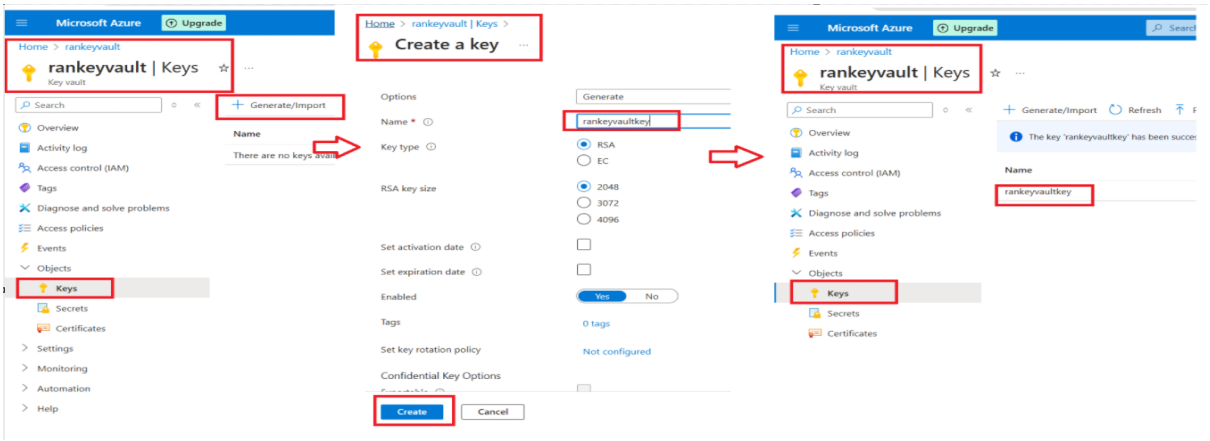


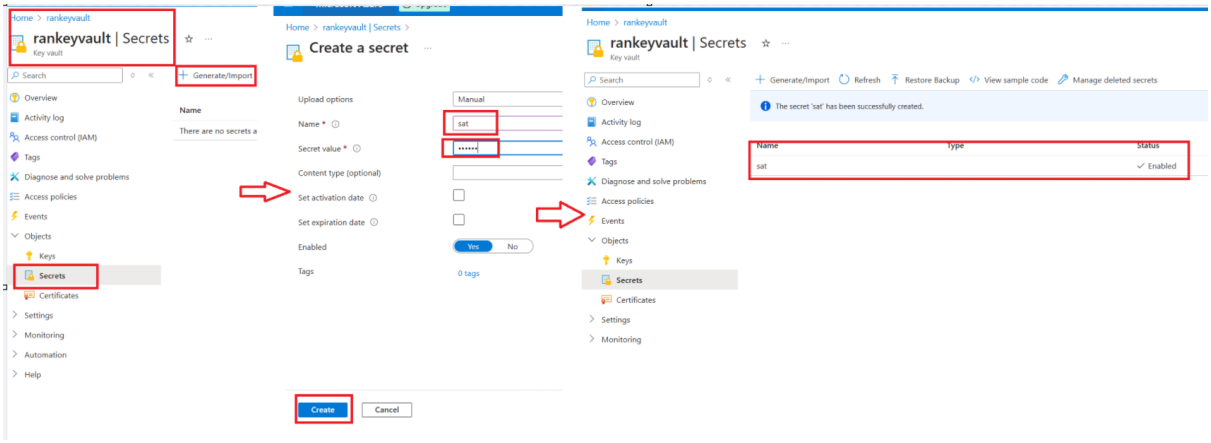

## iii. Add Secrets to the Vault

### CLI

az keyvault secret set --vault-name satkeyvault --name MySecret --value "SuperSecretValue"

### GUI






## iv. Enable KeyVault Add-on in AKS Or Enable CSI / Enable CSI Driver

**CLI**

az aks enable-addons --addons azure-keyvault-secrets-provider --name clusterbox1 --resource-group rgbox1

**To Verify**

kubectl get pod -n kube-system

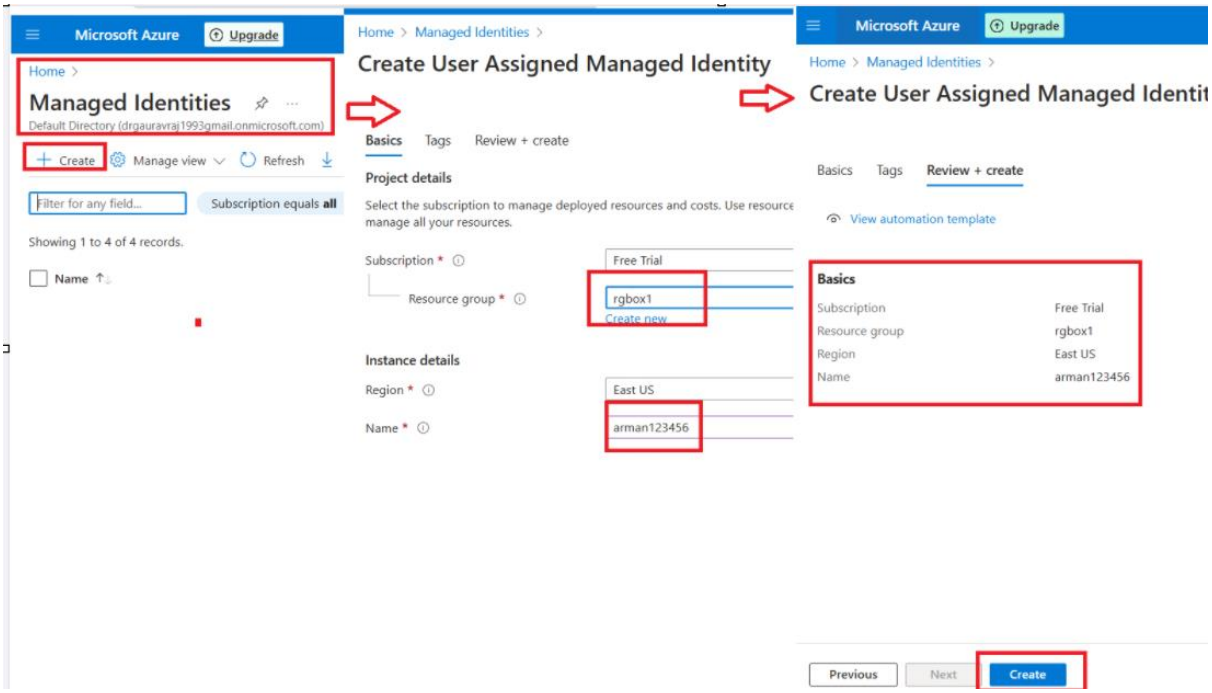
```
PS F:\> kubectl get pod -n kube-system
NAME                                     READY   STATUS    RESTARTS   AGE
aks-secrets-store-csi-driver-mr5vr       3/3     Running   0          7m8s
aks-secrets-store-csi-driver-v98kz       3/3     Running   0          7m8s
aks-secrets-store-provider-azure-cmng4   1/1     Running   0          7m8s
aks-secrets-store-provider-azure-zcj68   1/1     Running   0          7m8s
azure-cns-2nldz                          1/1     Running   0          6h20m
azure-cns-bj74t                          1/1     Running   0          6h20m
```

## Step4: Set up a User Managed Identity in AKS and Key Vault

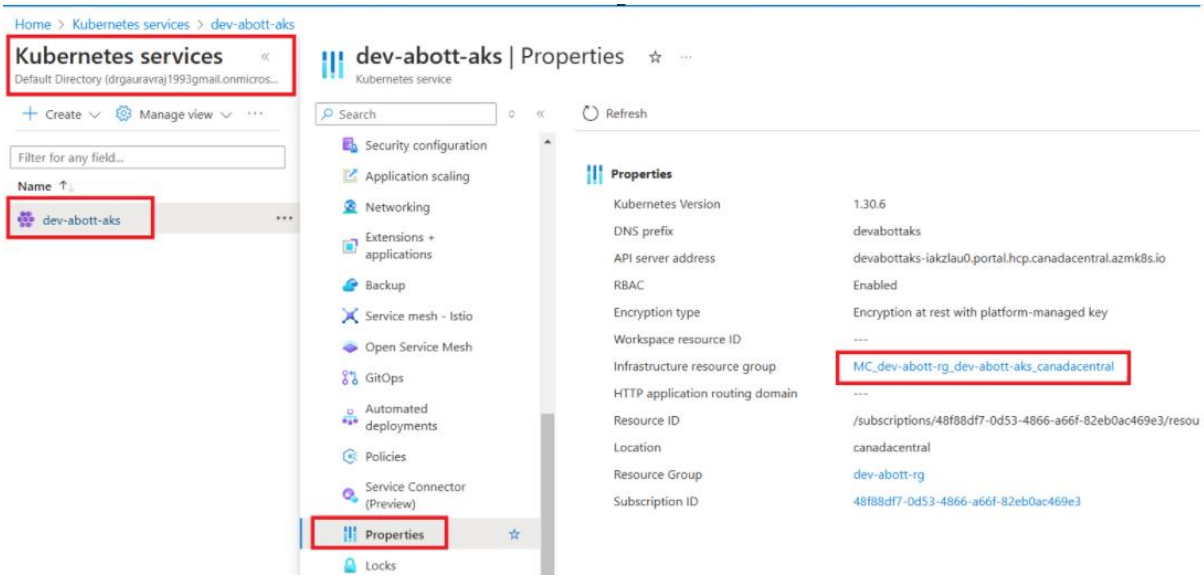
**CLI**

**GUI**

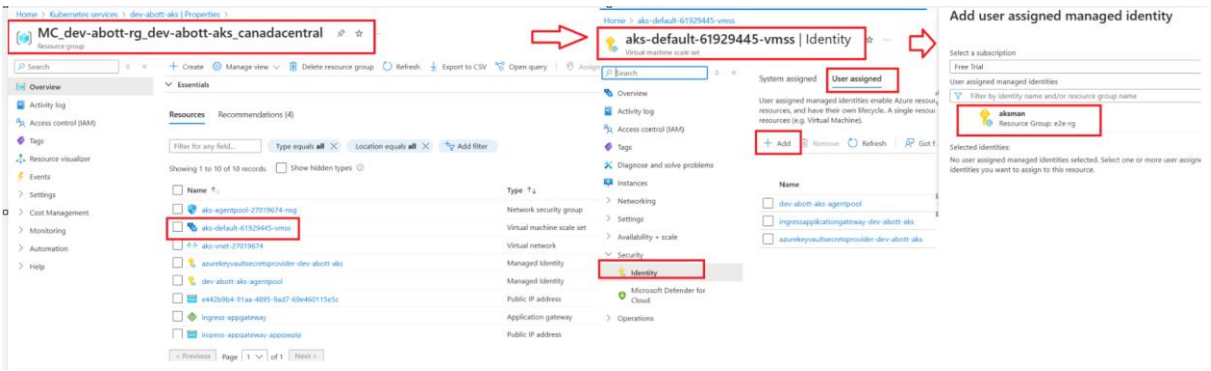
**Create one User Managed Identity in AKS**

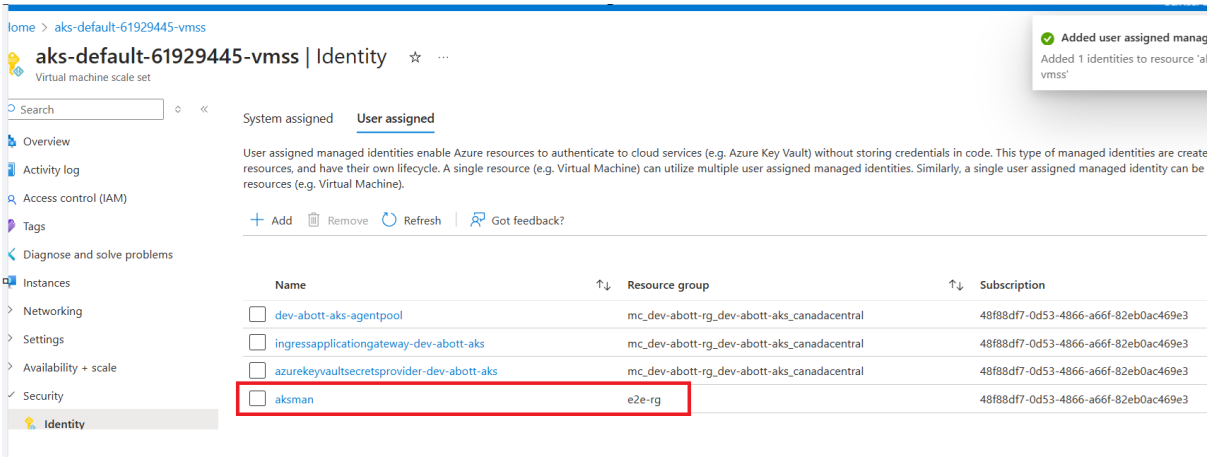



## Permission of User Managed Identity at AKS Side

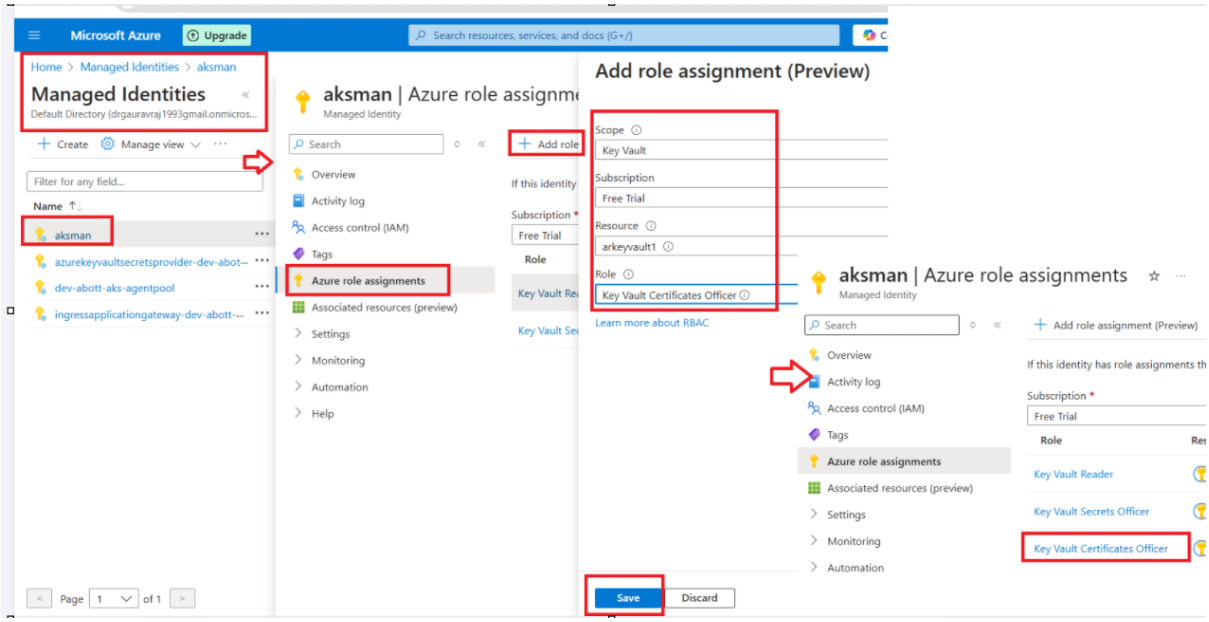



## Permission of User Managed Identity at Keyvault side






**Now Need to Assign Key vault certificate officer role for this user managed identity.**

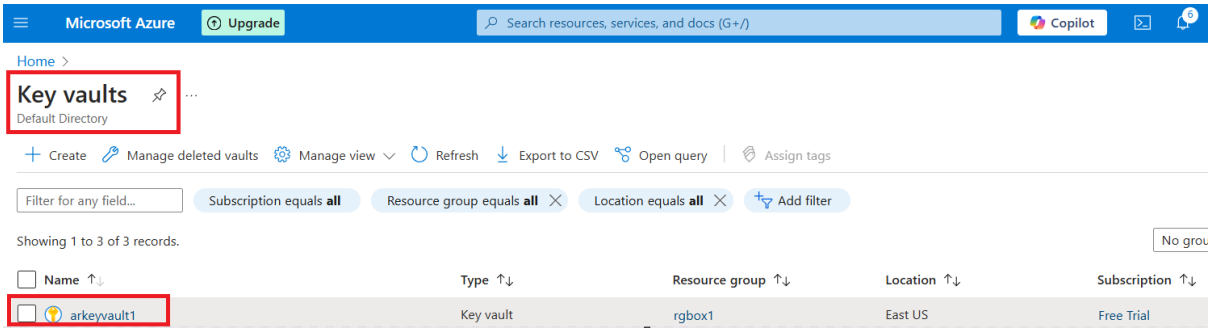


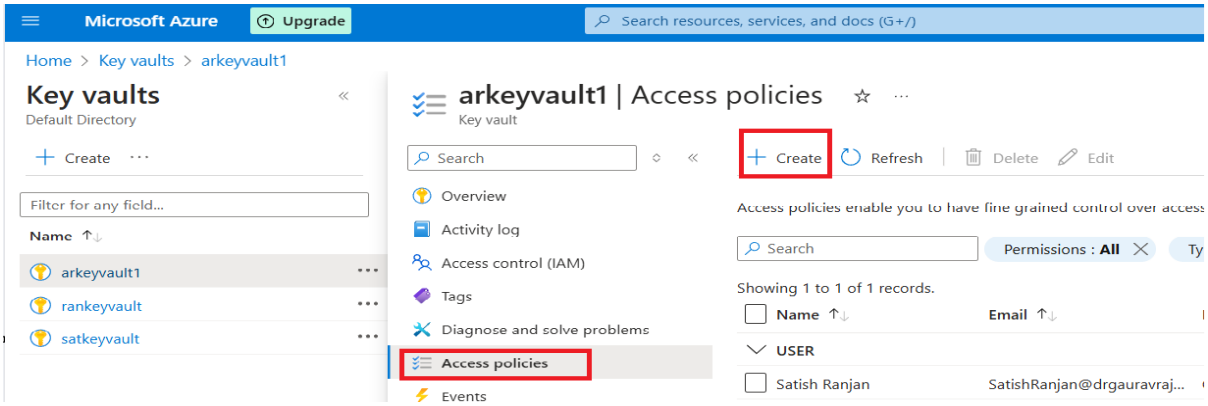

**Give Required Permission to Managed Identity at Key Vault Side**

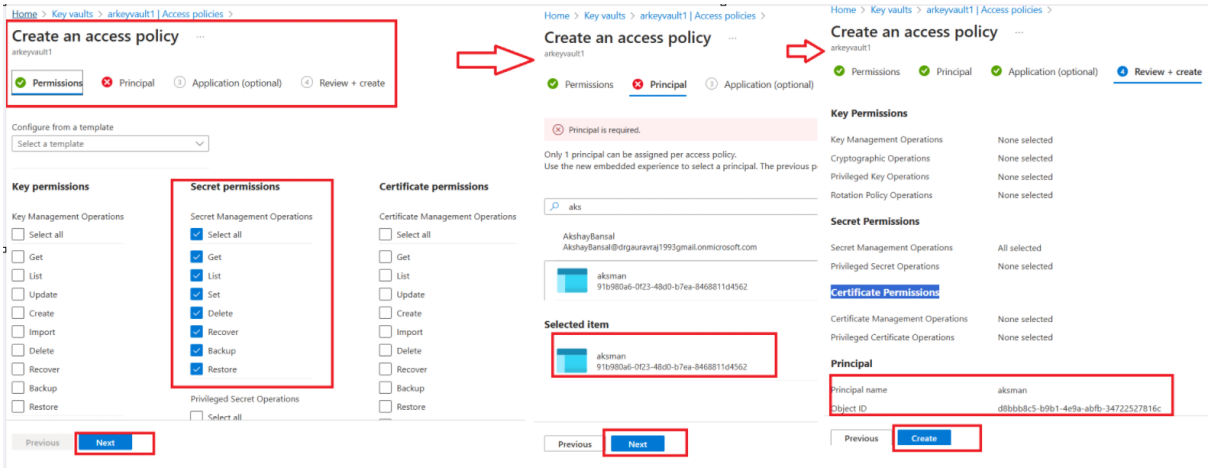
**CLI**

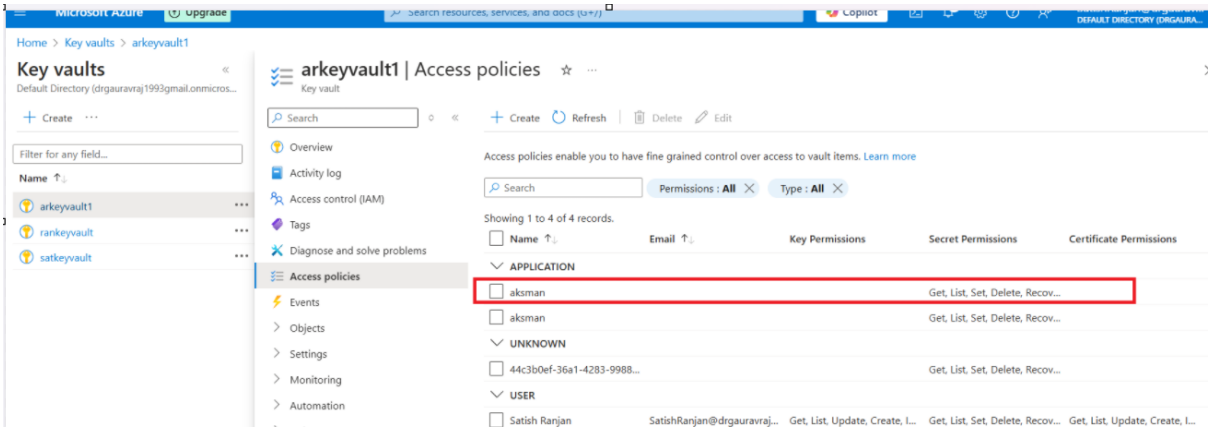
az keyvault set-policy -n MyKeyVault --secret-permissions get --spn <aks-identity-client-id>

**GUI**






## Step5: Configure and Deploy to Kubernetes

1. **Create a SecretProviderClass for VM Managed Identity:**

**Yaml File for SecretProviderClass**

```yaml
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: azure-kvname-system-mi
  namespace: default
spec:
  parameters:
    cloudName: AzurePublicCloud
    keyvaultName: aksdemopg
    objects: |
      array:
        - |
          objectName: pgpassword
          objectType: secret        # object types: secret, key, or cert
          objectVersion: ""         # [OPTIONAL] object versions, default to
    tenantId: 4cd93cbc-b026-4c7d-86f1-76fcb7292573 # The Directory ID of the
    usePodIdentity: "false"
    useVMManagedIdentity: "true"
    userAssignedIdentityID: 3457145b-5926-4a60-8d5f-7bb929192891 #Manage iden
  provider: azure
```

2. **Deploy Your Application:**

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: busybox-secrets-store-inline-system-msi
  namespace: default
spec:
  containers:
    - name: busybox
      image: k8s.gcr.io/e2e-test-images/busybox:1.29-4
      command:
        - /bin/sleep
        - '10000'
      resources: {}
      volumeMounts:
        - name: secrets-store01-inline
          readOnly: true
          mountPath: /mnt/secrets-store
  volumes:
    - name: secrets-store01-inline
      csi:
        driver: secrets-store.csi.k8s.io
        readOnly: true
        volumeAttributes:
          secretProviderClass: azure-kvname-system-mi
```

## Step 5: Access Your Secrets

```
PS C:\Users\NIRAV> kubectl exec busybox-secrets-store-inline-system-msi -- ls /mnt/secrets-store/
postgrespw
PS C:\Users\NIRAV> kubectl exec busybox-secrets-store-inline-system-msi -- cat /mnt/secrets-store/postgrespw
password
PS C:\Users\NIRAV>
```

**cat /mnt/secrets-store/MySecret**