



KubeCon



CloudNativeCon

North America 2018

Protect Your Kubernetes Data

Friends Don't Let Friends Leave Their Kubernetes Data Unprotected

Rita Zhang
Principal Software Engineer @ Microsoft

@ritazzhang



Rita Zhang

- Software engineer @ Microsoft, San Francisco
- Kubernetes upstream features, Azure Kubernetes Service
- Maintainer for K8s KMS plugin for Azure Key Vault, Keyvault-flexvolume, aad-pod-identity (along with @khnidk @nikhilbhatia @kkwriting @cpuguy83)



TRANSPORTATION

CARS

TESLA

Tesla's cloud was used by hackers to mine cryptocurrency

Mining bitcoin on Elon's dime

By [Andrew J. Hawkins](#) | [@andyjayhawk](#) | Feb 20, 2018, 1:39pm EST

The initial point of entry for the Tesla cloud breach, Tuesday's report said, was an unsecured administrative console for **Kubernetes**, an open source package used by companies to deploy and manage large numbers of cloud-based applications and resources.

The screenshot shows the Kubernetes dashboard interface. At the top, the browser address bar displays a URL with a red warning icon and the text "Not Secure". The dashboard header includes the Kubernetes logo and a search bar. A blue navigation bar shows the breadcrumb "Config and storage > Secrets > aws-s3-credentials". On the left, a sidebar menu lists various Kubernetes resources, with "Secrets" being the active section. The main content area is divided into two panels: "Details" and "Data". The "Details" panel shows the following information:

- Name: aws-s3-credentials
- Namespace: default
- Creation time: 2017-10-12T22:29
- Type: Opaque

The "Data" panel displays two entries, each with a redacted value:

- aws-s3-access-key-id: [Redacted]
- aws-s3-secret-access-key: [Redacted]

A vertical watermark "RedLock" is visible on the left side of the dashboard.

Kubernetes Database

- Uses etcd as its persistent storage for API objects
- Stores secrets as base64 encoded plaintext

The security footgun in etcd



Giovanni Collazo

March 16, 2018

security



“Authentication was added in etcd 2.1. ... etcd before 2.1 was a completely open system; anyone with access to the API could change keys. In order to preserve backward compatibility and upgradability, this feature is off by default.” [Read more from coreos etcd doc](#)

<https://elweb.co/the-security-footgun-in-etcd/>

I did a [simple search on shodan](#) and came up with **2,284 etcd servers on the open internet**.

CREDENTIALS, a lot of CREDENTIALS. Credentials for things like cms_admin, mysql_root, postgres, etc. Passwords for databases of all kinds, AWS secret keys, and API keys and secrets for a bunch of services.

GET http://<ip address>:2379/v2/keys/?recursive=true

password	8781
aws_secret_access_key	650
secret_key	23
private_key	8

```
- {
  key: "/registry/secrets/ci",
  dir: true,
  nodes: [
    - {
      key: "/registry/secrets/ci/docker-repo-key",
      value: "{\"kind\":\"Secret\",\"apiVersion\":\"v1\",\"metadata\":{\"name\":\"docker-repo-key\",\"namespace\":\"ci\",\"uid\":\"c98f370a-d825-11e6-a077-bc764e04\"},\"data\":{\"dockerRepoKey\":\"\"}}",
      modifiedIndex: 12910772,
      createdIndex: 12910772
    },
    - {
      key: "/registry/secrets/ci/default-token-dt40q",
      value: "{\"kind\":\"Secret\",\"apiVersion\":\"v1\",\"metadata\":{\"name\":\"default-token-dt40q\",\"namespace\":\"ci\",\"uid\":\"f849c1c5-ea76-11e7-94ad-bc764e04\"},\"data\":{\"ca.crt\":\"LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUZVVENDQXptZ0F3SUJBZ0lKQUlydVRyaThFTnBjTUEwR0NTcUdTSWlzMkRFRkQkN3VUFNRDh4Q3pBSkFnTlYKQ\"}}",
      modifiedIndex: 651393897,
      createdIndex: 651393897
    }
  ],
  modifiedIndex: 4005597,
  createdIndex: 4005597
}
```

An attacker who can successfully access your cluster database can compromise your entire cluster and have access to your cloud resources.



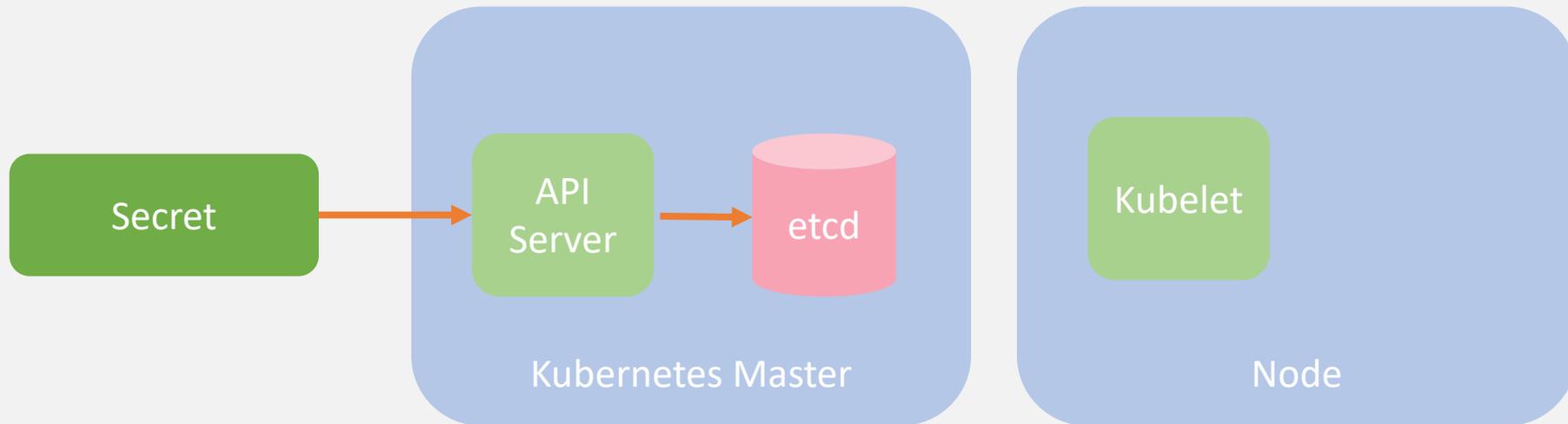
THIS IS FINE.

So...How do I secure my cluster?

- There are many things you can do
 - Control access to the Kubernetes APIs
 - Control access to the Kubelet
 - Control privileges containers run with
 - Restrict network access
 - Restrict resource access
 - Restrict access to etcd
 - **Encrypt etcd data at rest**
 - **Store application secrets outside of Kubernetes**
 - **Restrict access to resources with pod identity**

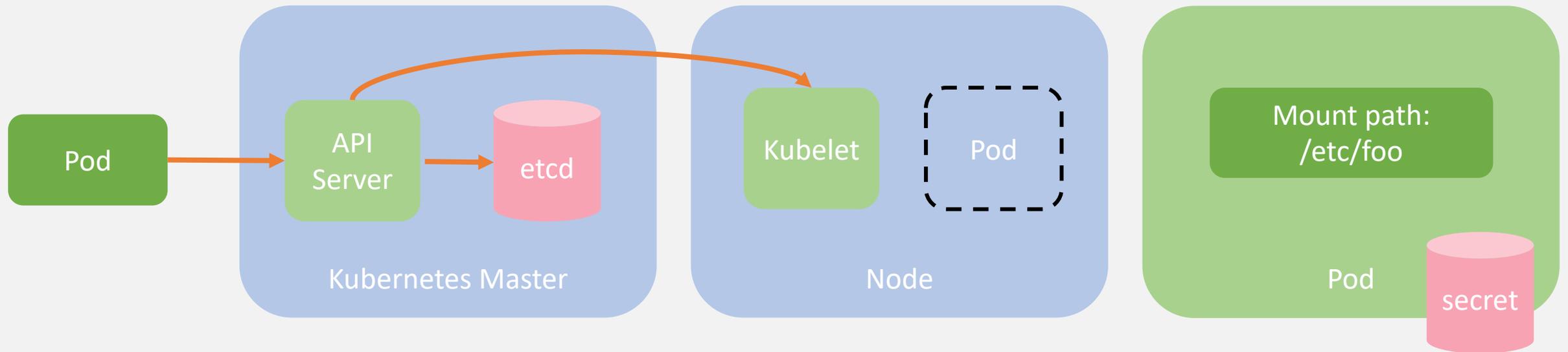
Secrets

`kubectl create secret generic secret1`

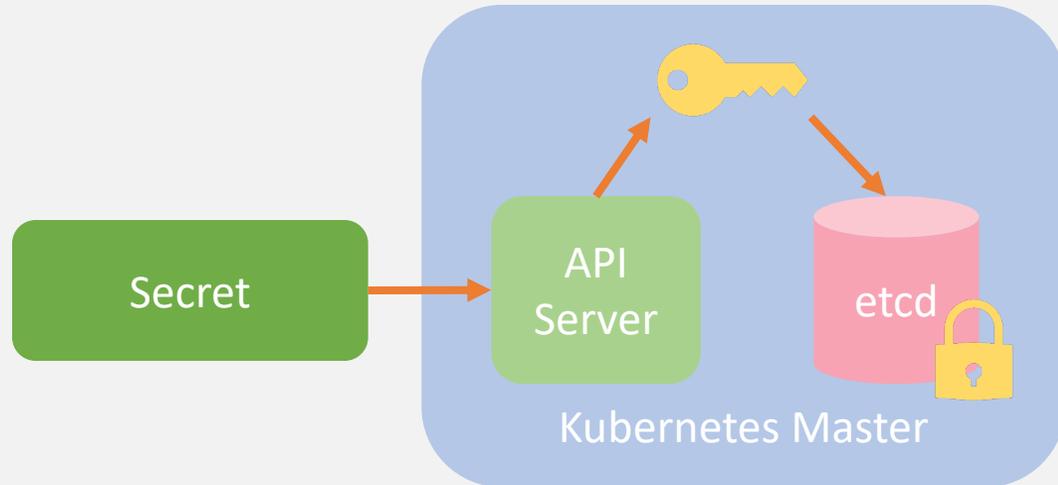


Pod using Secret

```
kubectl create -f pod-using-secret.yaml
```



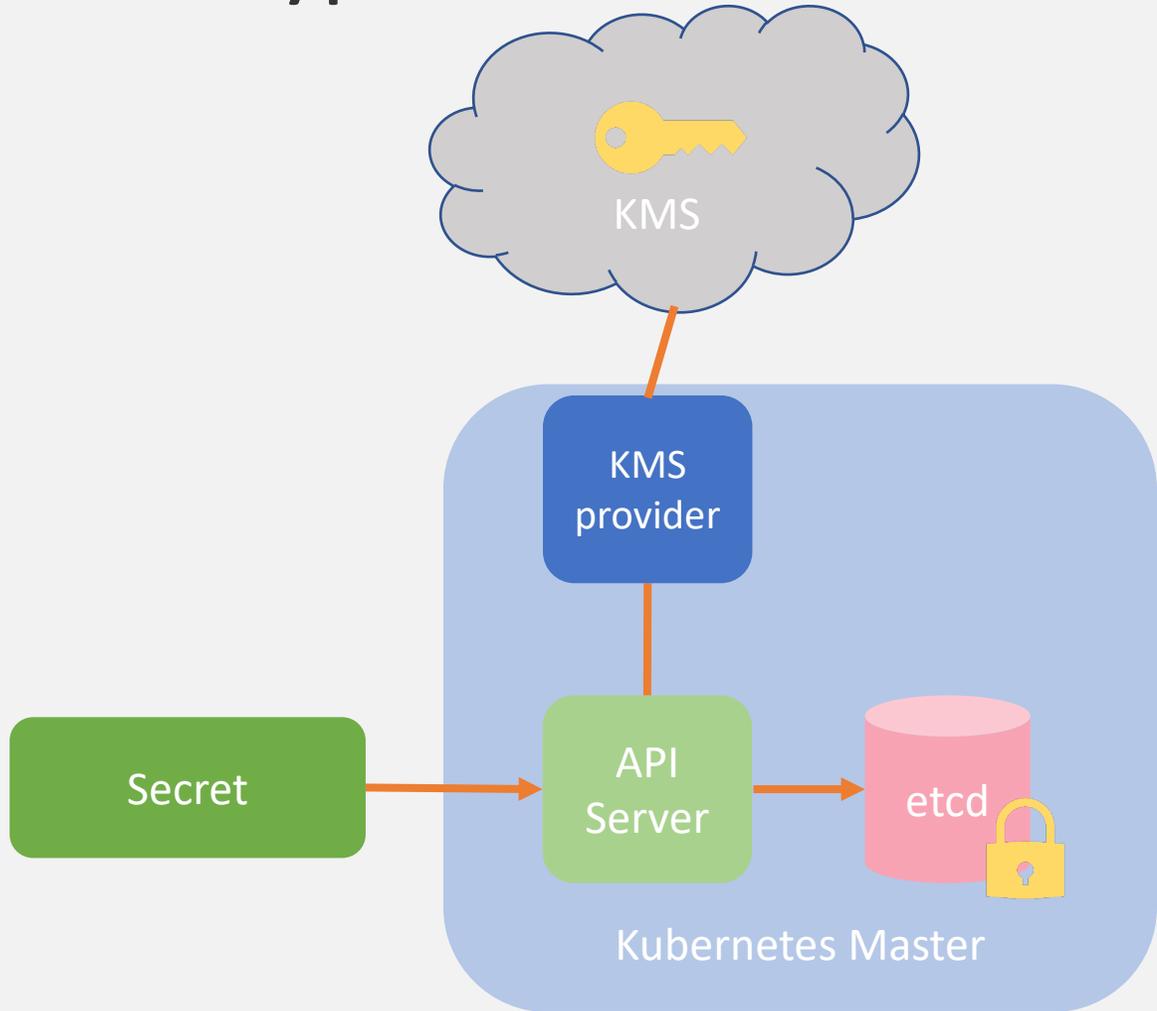
Encryption at Rest



- Kubernetes v1.7+
- etcd v3 required
- encryption using keys in config file on master
- Plain text, encoded with base64

```
kind: EncryptionConfig
apiVersion: v1
resources:
  - resources:
    - secrets
  providers:
    - aescbc:
      keys:
        - name: key1
          secret: <BASE 64 ENCODED SECRET>
- identity: {}
```

Key Management Service (KMS) Provider for Encryption at Rest

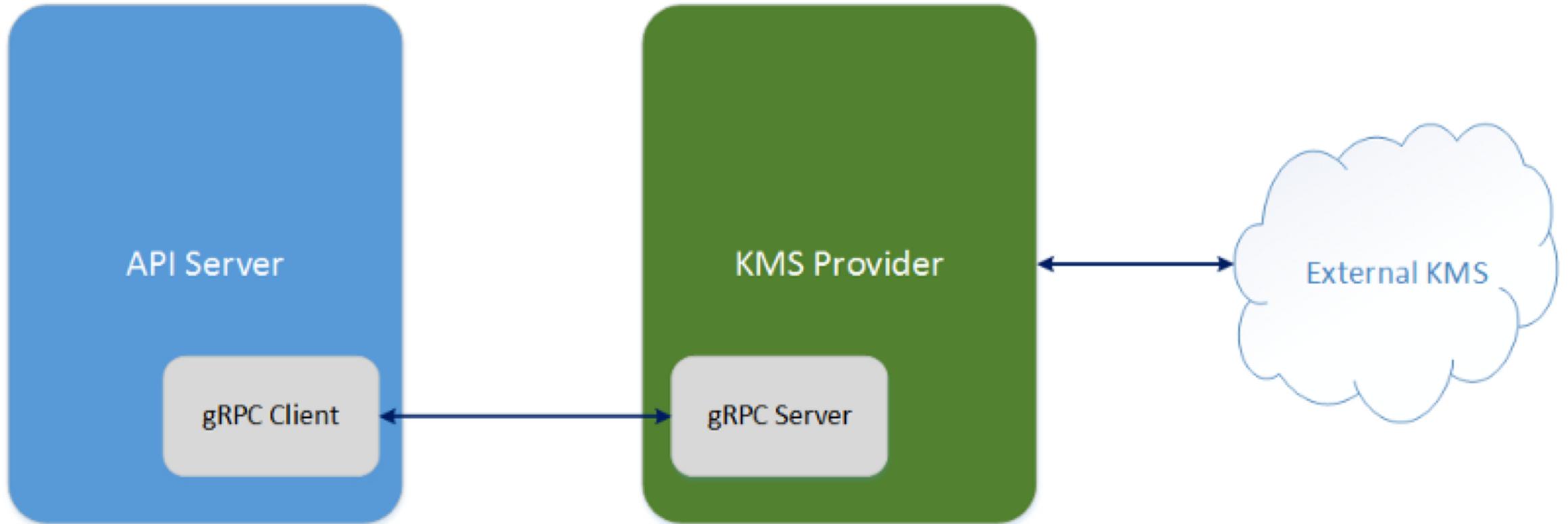


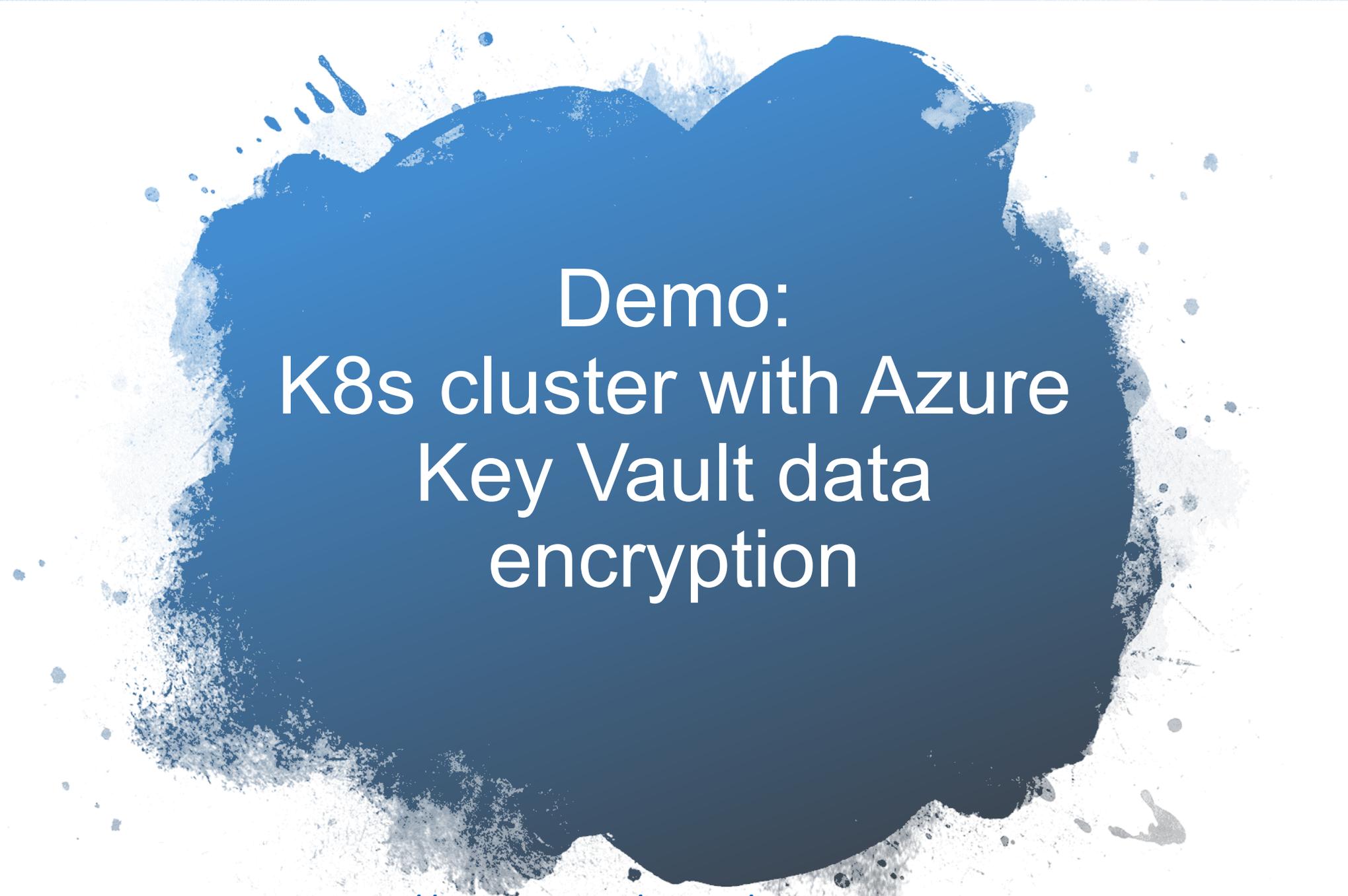
- Kubernetes v1.10, v1.13 stable
- etcd v3 required
- Separate key management from K8s cluster management
- Supports encryption using keys stored in external trusted Key Management Service (KMS), e.g. Azure Key Vault, Google Cloud KMS
- Hardware Security Modules (HSM)-protected keys

V1.13

```
kind: EncryptionConfiguration
apiVersion: apiserver.config.k8s.io/v1
resources:
  - resources:
    - secrets
  providers:
    - kms:
      name: myKmsPlugin
      endpoint: unix:///tmp/socketfile.sock
      cachesize: 100
  - identity: {}
```

High Level Design





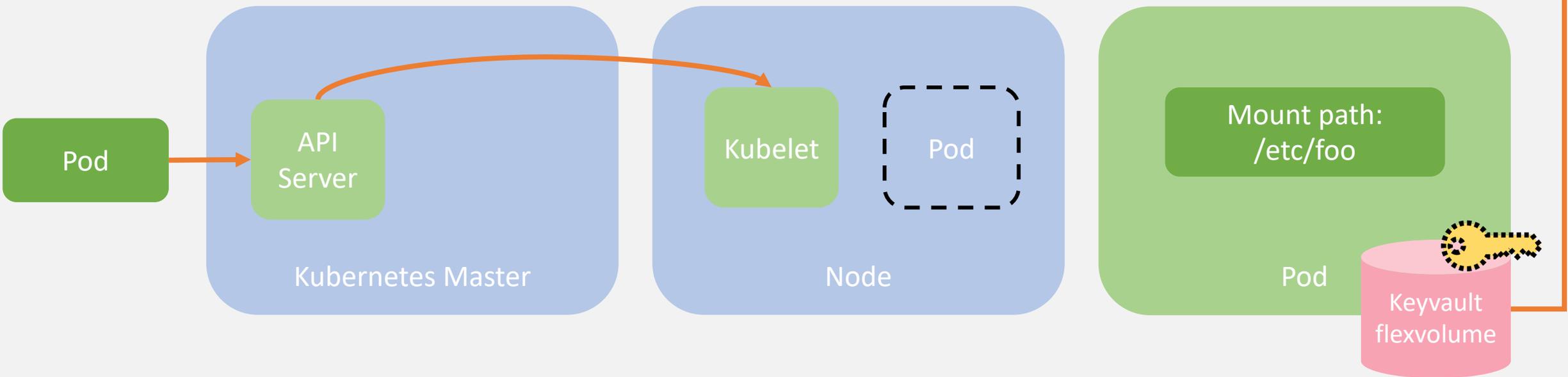
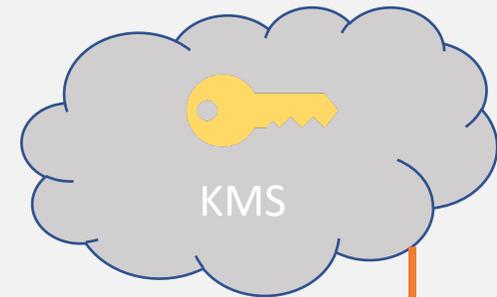
Demo:
K8s cluster with Azure
Key Vault data
encryption

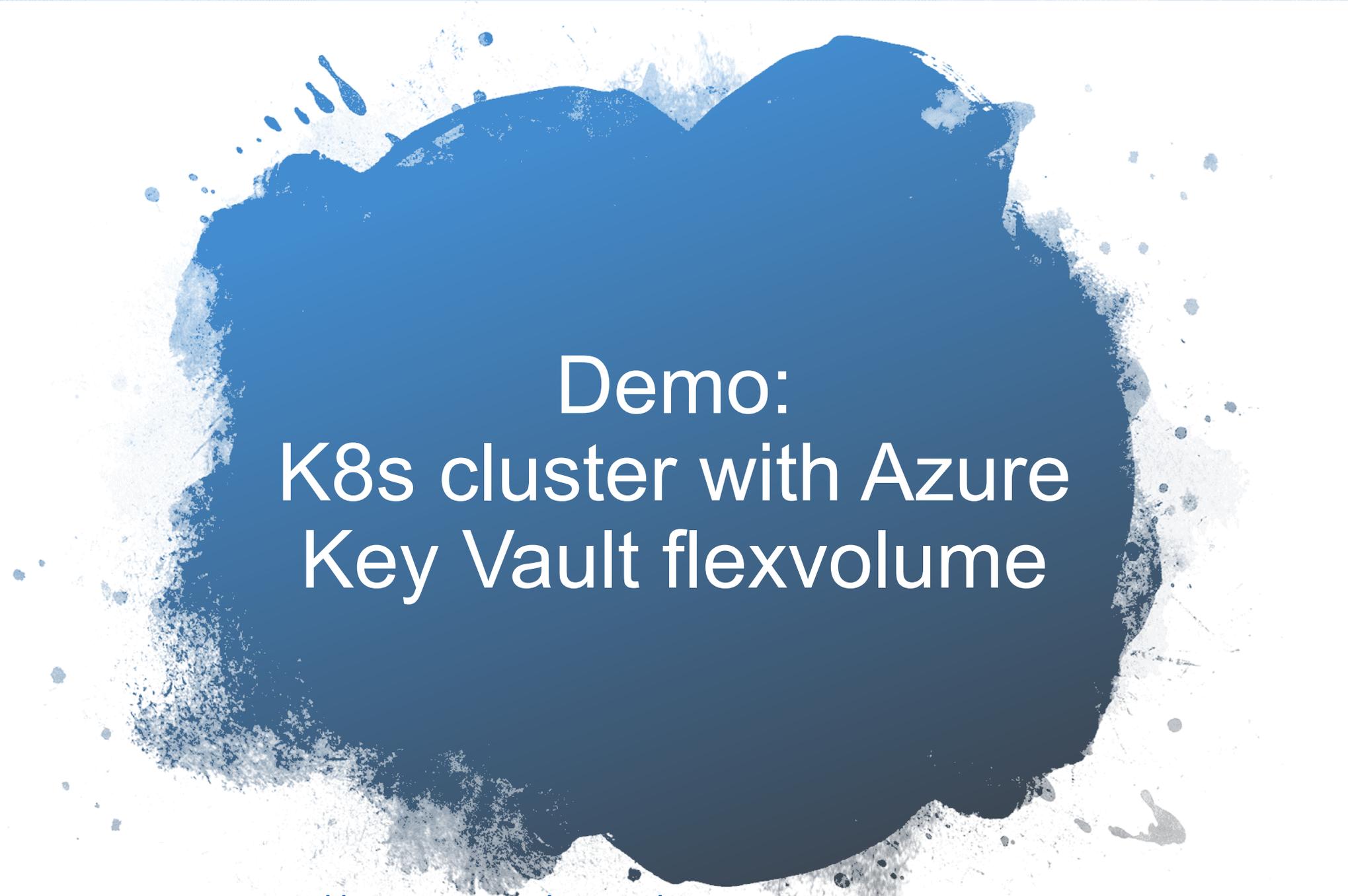
With the KMS provider plugin, we can encrypt Kubernetes data stored in etcd at rest with a KMS managed key.

What if instead of storing my secrets in etcd, I want to store and manage access outside of Kubernetes?

Pod using Key Vault Flexvolume

```
kubectl create -f pod-using-keyvaultflexvolume.yaml
```





Demo:
K8s cluster with Azure
Key Vault flexvolume

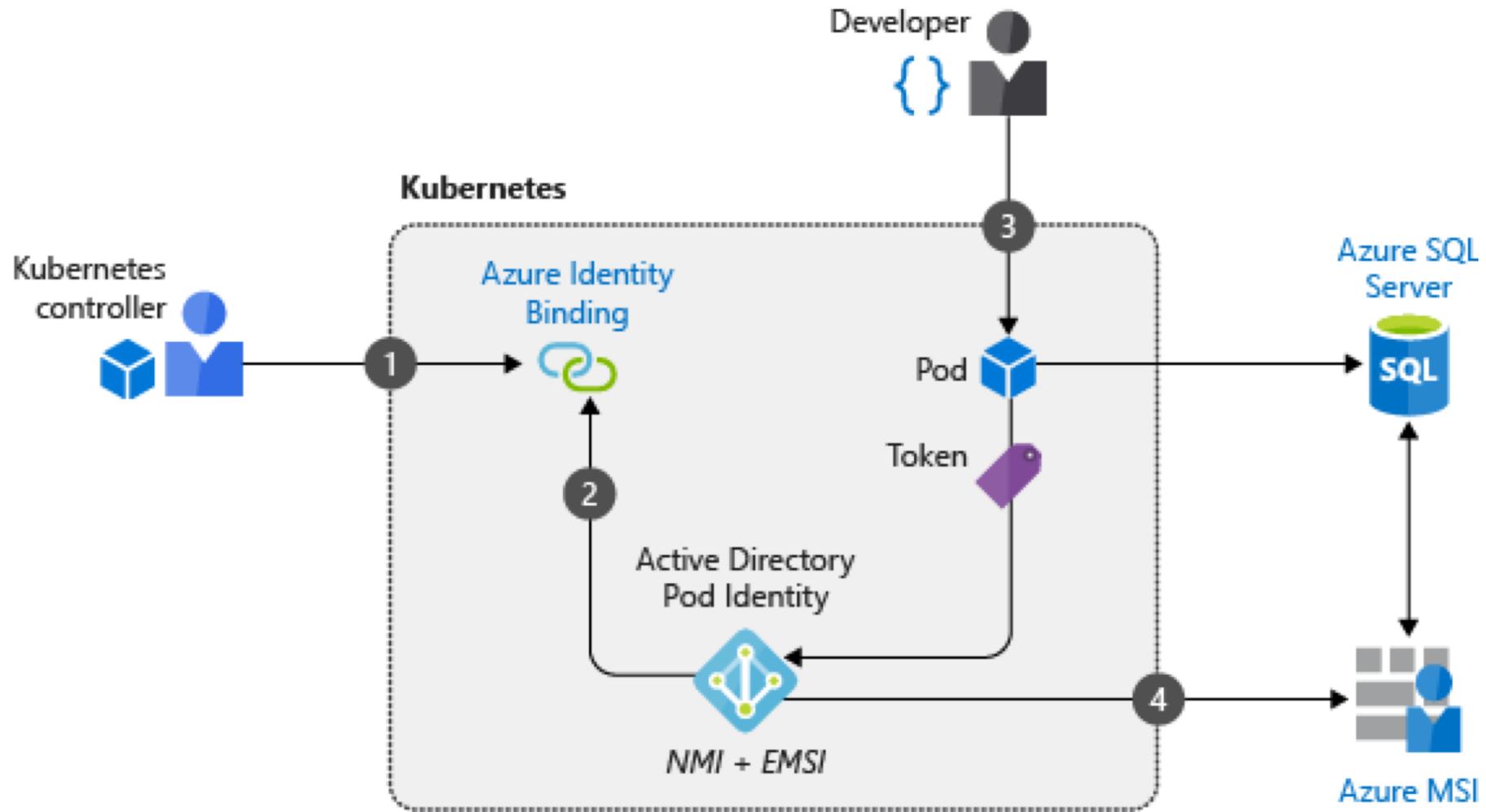
With the Kubernetes Key Vault FlexVolume driver, we can store and retrieve secrets from a Key Management System and mount the data as a volume to containers.

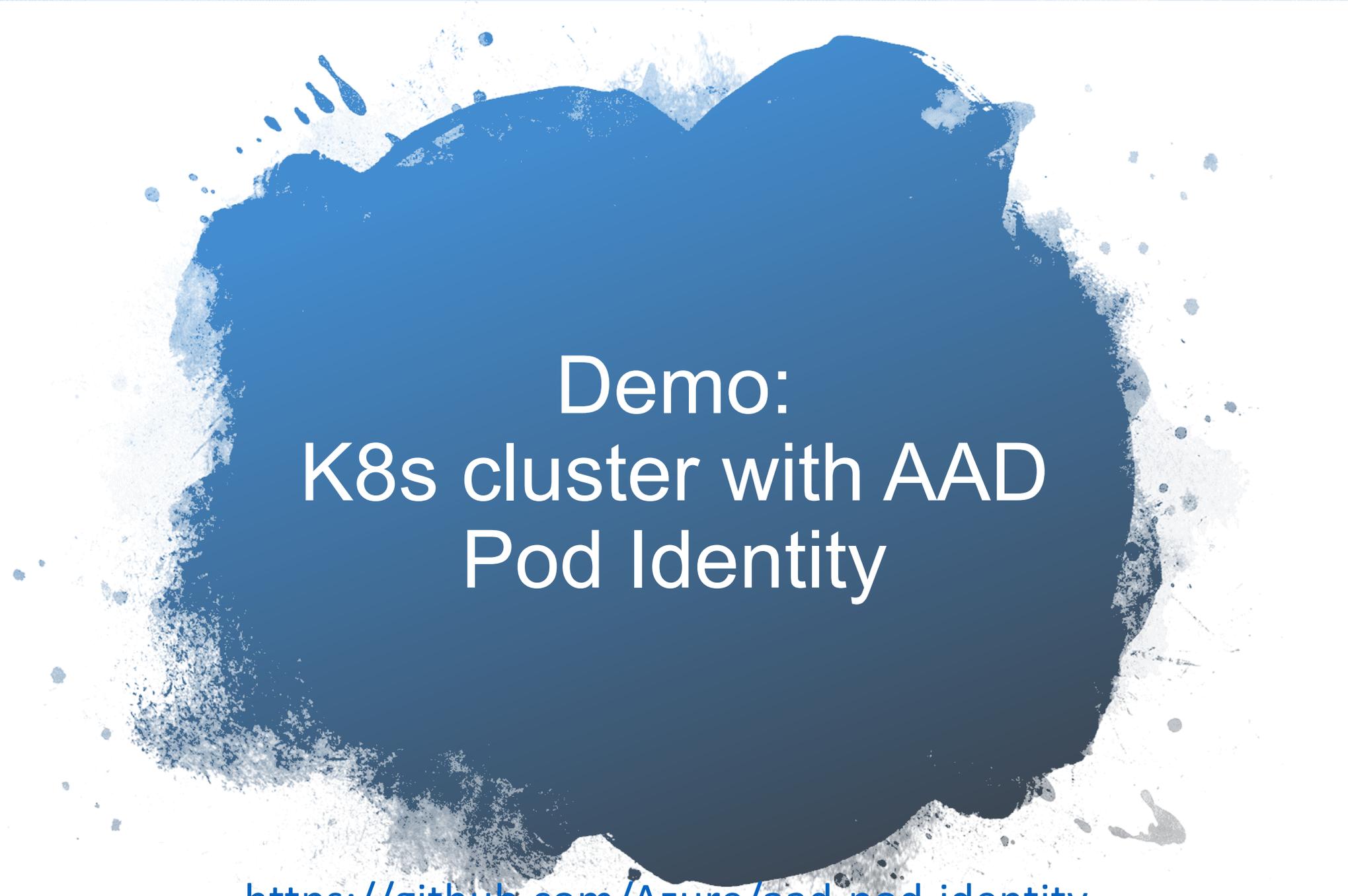
What if I want to restrict access to my key vault or my cloud resources to specific pods?

AAD Pod Identity

- Restrict/enable pods to access individual resources that depend on Azure AD for access with its own identity (e.g. Azure SQL server or your own custom API that uses AAD)
- Kubernetes Custom Resource Definition objects that map pods to Azure AD identities
- When pods request access to a resource that uses Azure AD for access, a matching Azure identity is assigned

AAD Pod Identity

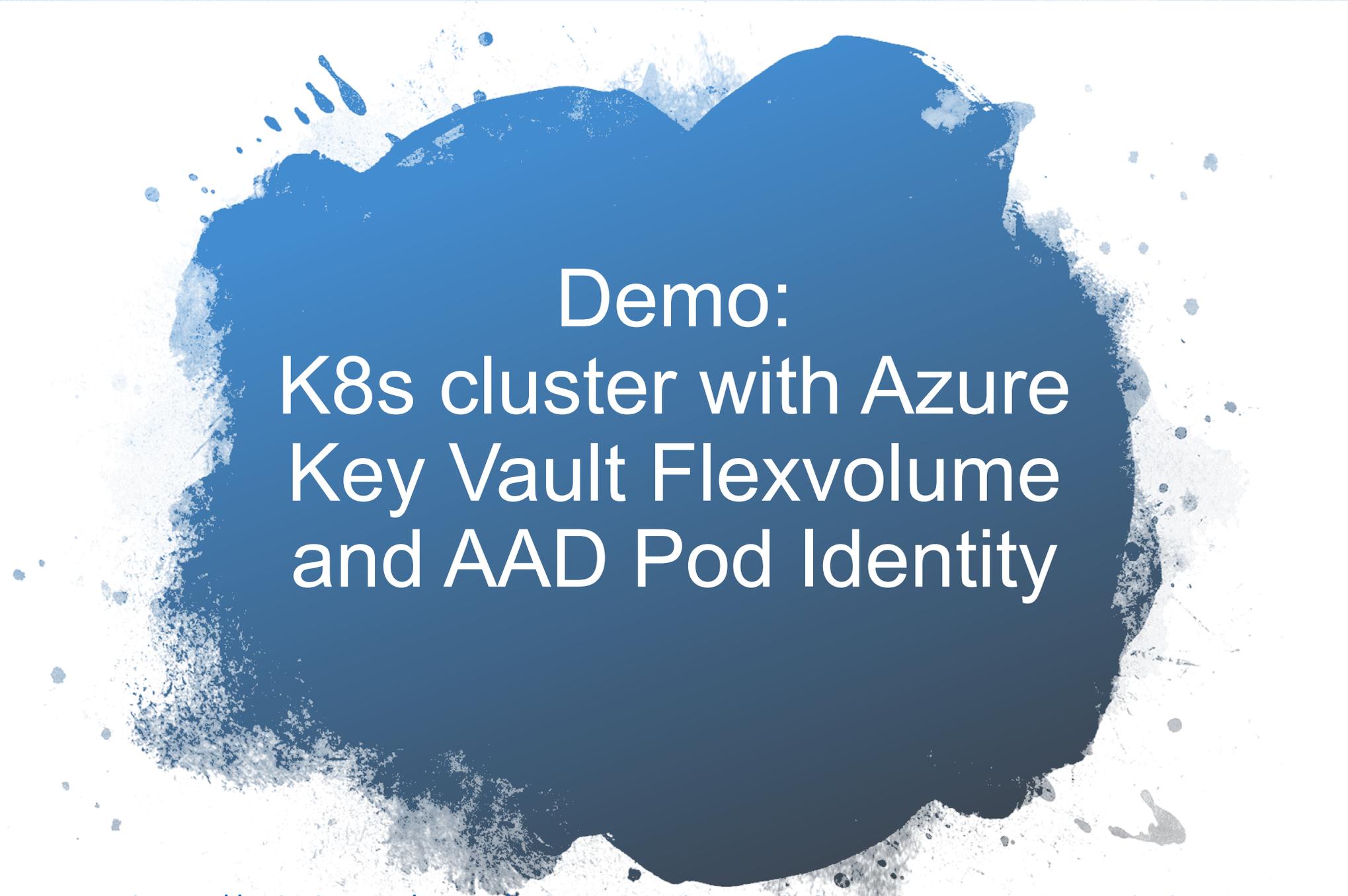




Demo: K8s cluster with AAD Pod Identity

<https://github.com/Azure/aad-pod-identity>

With AAD Pod Identity, we can restrict and enable specific pods access to resources that need Azure AD for access based on its identity.



Demo:
K8s cluster with Azure
Key Vault Flexvolume
and AAD Pod Identity

Recap

Azure Key Vault KMS plugin

- Use a key, HSM-protected, in Key Vault for etcd encryption
- Secrets/keys/certs are stored in etcd, managed as part of Kubernetes
- Restrict access using K8s concepts: RBAC, Service Accounts, namespaces
- Bring your own keys
- Available on AKS-engine

Key Vault Flexvol + AAD Pod Identity

- Mounts secrets/keys/certs to the pod using a flexvolume
- Secrets/keys/certs are stored in Key Management System
- Restrict access to secrets/keys/certs with specific identities assigned to a pod
- Manage secrets/keys/certs in Key Management System
- Separation of concerns/role-based management access to secrets
- Import or generate HSM-protected (Hardware Security Modules) keys
- Industry compliance
- More granular RBAC at the pod level

Resources

- Blog post: <https://ritazh.com/using-azure-key-vault-for-kubernetes-data-encryption-d5eac8daee71>
- Kubernetes doc: <https://kubernetes.io/docs/tasks/administer-cluster/kms-provider/>
- aks-engine doc: <https://github.com/Azure/aks-engine/blob/master/docs/kubernetes/features.md#azure-key-vault-data-encryption>
- KMS plugin service PR: <https://github.com/kubernetes/kubernetes/pull/55684>
- Kubernetes KMS Plugin for Azure Key Vault: <https://github.com/Azure/kubernetes-kms>
- Kubernetes KMS Plugin for Google CloudKMS: <https://github.com/GoogleCloudPlatform/k8s-cloudkms-plugin/>
- Kubernetes Key Vault FlexVolume: <https://github.com/Azure/kubernetes-keyvault-flexvol>
- AAD Pod Identity: <https://github.com/Azure/aad-pod-identity>



KubeCon

CloudNativeCon

————— **North America 2018** —————

