

# Protect Your Kubernetes Data with Azure Key Vault

✓ Friends don't let friends leave their Kubernetes data unprotected

Rita Zhang  
Principal SDE @ Microsoft

 ritazh  
 ritazzhang



# About Me

- Software engineer, Azure Kubernetes Service team, San Francisco
- Partner with open source communities to tackle challenging problems with customers and partners
- Contributor to Azure KMS provider for k8s, open service broker for azure, Helm charts, acs-engine, virtual-kubelet etc...



ritazh



ritazzhang



# Kubernetes Database

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

<https://kubernetes.io/docs/concepts/overview/components/#etcd>

# The security footgun in etcd



Giovanni Collazo

March 16, 2018

security



*“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**. So I clicked a few and on the third try I saw what I was hoping not to see. CREDENTIALS, a lot of CREDENTIALS. Credentials for things like cms\_admin, mysql\_root, postgres, etc.

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

Then I performed a few basic searches to get a sense of what was there and found a bunch of credentials. Passwords for databases of all kinds, AWS secret keys, and API keys and secrets for a bunch of services.

password	8781
aws_secret_access_key	650
secret_key	23
private_key	8

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

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 secret's name, namespace, creation time, and type. The "Data" panel displays two entries: "aws-s3-access-key-id" and "aws-s3-secret-access-key", both of which are redacted with yellow bars.

Not Secure | [https://\[redacted\]/#/secret/default/aws-s3-credentials?namespace=default](https://[redacted]/#/secret/default/aws-s3-credentials?namespace=default)

kubernetes Search

Config and storage > Secrets > aws-s3-credentials

namespace: default

Overview

Workloads

- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

Discovery and Load Balancing

- Ingresses
- Services

Config and Storage

### Details

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

### Data

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





# GAME OVER

Don't give up now, Mario...



Hope is still underway...  
Do you want to try again?

▶ YES or NO

# 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 data at rest**

<https://kubernetes.io/docs/tasks/administer-cluster/securing-a-cluster/>

# Encryption at Rest

- Introduced in Kubernetes v1.7
- etcd v3 is required
- Only supports encryption using keys in config file
- Plain text, encoded with base64
- Set the `--experimental-encryption-provider-config` flag on the apiserver to point to the location of the config file
- Config file specifies resources to be encrypted and providers and keys to use for encryption and decryption

<https://kubernetes.io/docs/tasks/administer-cluster/encrypt-data/>

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



# KMS provider for Encryption at Rest

- Introduced in Kubernetes v1.10, alpha feature
- Separate key management from K8s cluster management
- use an external trusted Key Management Service (KMS) to manage the keys, e.g. Azure Key Vault
- etcd v3 is required
- Set the `--experimental-encryption-provider-config` flag on the apiserver to point to the location of the config file
- Config file specifies resources to be encrypted and a UNIX socket endpoint of the KMS plugin to use for encryption and decryption

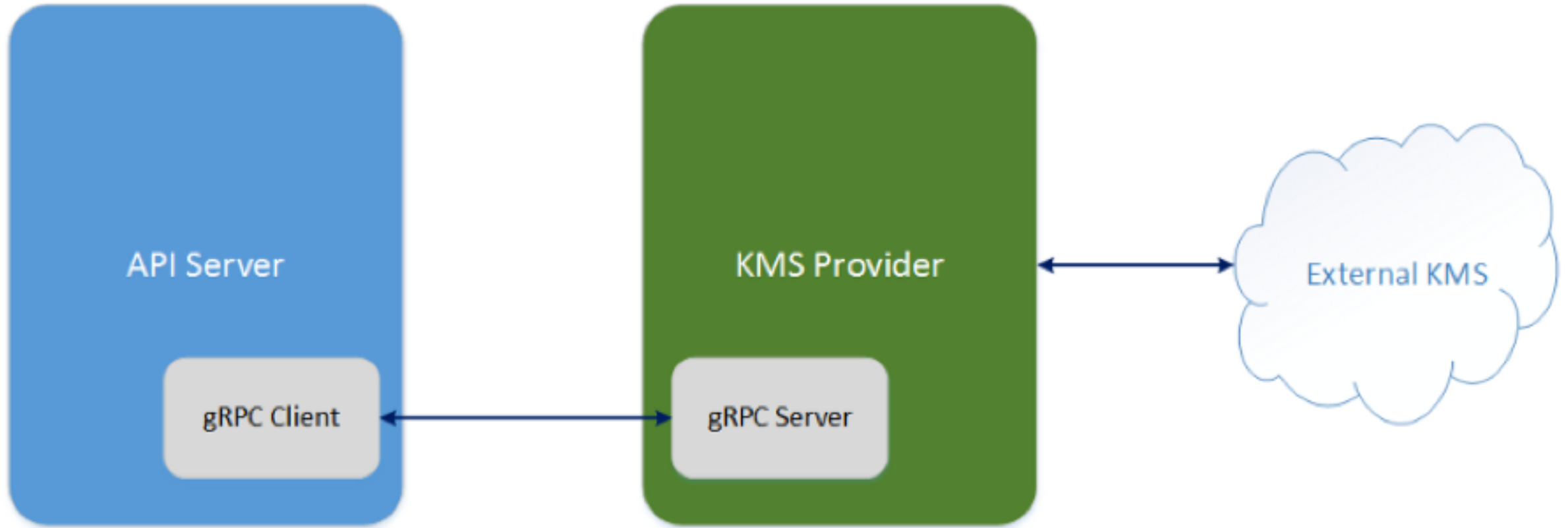
<https://kubernetes.io/docs/tasks/administer-cluster/kms-provider/>

```
kind: EncryptionConfig
apiVersion: v1
resources:
  - resources:
    - secrets
  providers:
    - kms:
        name: myKmsPlugin
```

# How Do I Use This?

- Kubernetes cluster v1.10+
-  acs-engine
  - [Guide](#)
  - [api model](#)
-  SOON Azure Kubernetes Service (AKS)

# High Level Design





# Components

- The Contract - Public APIs for the the gRPC service:  
[service.proto](#)
- The Server – gRPC server
- The Client – Kubernetes API Server

# A new KMS Provider

- gRPC server
  - how to connect to the external KMS
  - how to authenticate with the external KMS
  - which keys to use for encryption and decryption
  - exposed as a UNIX domain socket connect  
(e.g. `unix:///opt/azurekms.sock`)

<https://github.com/Azure/kubernetes-kms/blob/master/server.go>

With the *Azure Key Vault KMS provider plugin*, we can encrypt Kubernetes data stored in etcd at rest with an Azure Key Vault managed key.

Is it possible to store application secrets in a Key Management Service and have Kubernetes query the secret value and use it in the application?

# Kubernetes Key Vault FlexVolume

- Flexvolume enables users to mount vendor volumes into kubernetes. It expects vendor drivers to be installed in the volume plugin path on every kubelet node.
- Driver makes a request to Azure Key Vault and mounts secret and secret value as a volume to containers

<https://github.com/Azure/kubernetes-keyvault-flexvol>

With the Kubernetes Key Vault FlexVolume driver, we can store and retrieve secrets from an Azure Key Vault instance and mount the values as a volume to containers.

# Resources

- Blog post: <https://ritazh.com/using-azure-key-vault-for-kubernetes-data-encryption-d5eac8daee71>
- Kubernetes doc for Using a KMS provider for data encryption: <https://kubernetes.io/docs/tasks/administer-cluster/kms-provider/>
- acs-engine doc: Enabling Azure Key Vault Encryption: <https://github.com/Azure/acs-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 repo: <https://github.com/Azure/kubernetes-kms>
- Kubernetes KMS Plugin for Google CloudKMS repo: <https://github.com/GoogleCloudPlatform/k8s-cloudkms-plugin/>
- Kubernetes Key Vault FlexVolume repo: <https://github.com/Azure/kubernetes-keyvault-flexvol>



**S** OPEN SOURCE SUMMIT  
JAPAN

THE LINUX FOUNDATION



AUTOMOTIVE  
LINUX SUMMIT

