KUBE AND VAULT

JAKUB VEVERKA
(@VELKYK)

TABLE OF CONTENTS

WHAT IS THIS TALK ABOUT?

- Short PoC - don't try this at home (Production)
- Quick Vault overview
- Quick review of kube annotations feature
- How to generate Vault token for pod
- How to assign Vault policy for each pod based on its annotations.

VAULT

- Secure Secret Storage
- Dynamic Secrets
- Data Encryption
- Leasing and Renewal
- Revocation

VAULT - AUTHENTICATION

- authentication works by verifying your identity and then generating a tokens to associate with that identity,
- tokens map to information,
- the most important information mapped to a token is a set of one or more attached policies.
- authentication backends:
  - AppRole
  - Github
  - LDAP
  - ...

VAULT - AUTHORIZATION

Policy definition:

```
path "sys/*" {
  policy = "deny"
}

path "secret/*" {
  policy = "write"
}

path "secret/foo" {
  policy = "read"
  capabilities = ["create", "sudo"]
}

path "secret/super-secret" {
  capabilities = ["deny"]
}
```

VAULT - RESPONSE WRAPING

- Generate Vault token with required privileges and store them in chubby,
- generate wrapper token with TTL,
- return wrapper token that is passed to original client.

KUBERNETES

The orchestrator

ANNOTATIONS

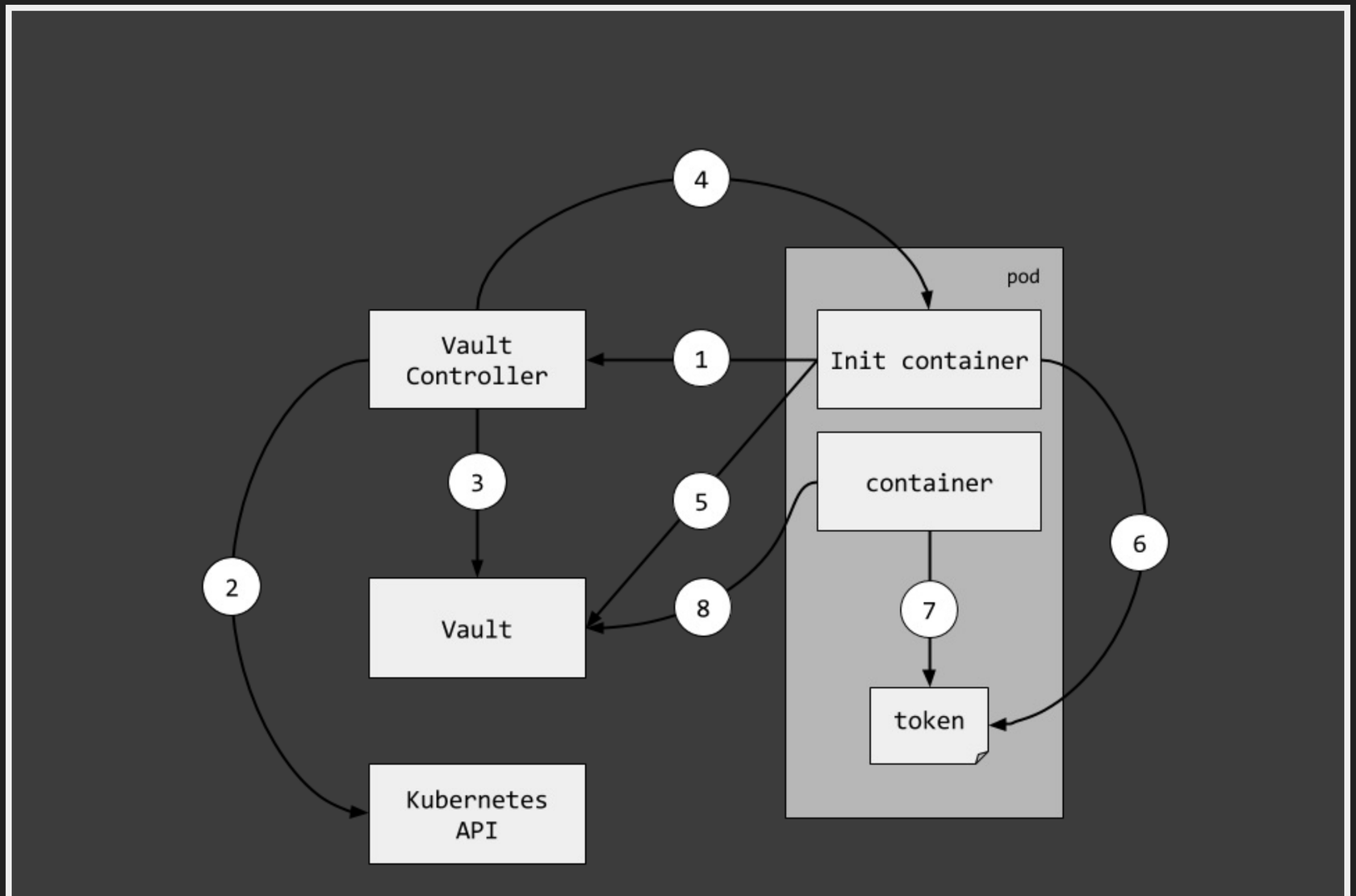
- Arbitrary non-identifying metadata, for retrieval by API clients such as tools, libraries, etc.
- Everything that doesn't belong to labels.

```
metadata:
  name: nginx
  annotations:
    key1: value1
    key2: value2
```

ANNOTATIONS - INITIALIZATION

```
metadata:
  name: nginx
  annotations:
    pod.beta.kubernetes.io/init-containers: '[
        {
            "name": "install",
            "image": "busybox",
            "command": ["wget", "-O", "/work-dir/index.html", "http://kubernetes.io/index.html"],
            "volumeMounts": [
                {
                    "name": "workdir",
                    "mountPath": "/work-dir"
                }
            ]
        }
    ]'
```

POC DEMO


4
pod
Vault
Controller
1
Init container
3
5
container
6
2
Vault
8
7
token
Kubernetes
API

VAULT CONTROLLER

- Listens to token request from pods (pod namespace and name as input),
- verify pod at kube apiserver and retrieve policies and ttl (pod info is trusted),
- generate wrapped token request,
- post token of wrapped request to Init container.

VAULT CONTROLLER REQUEST

```
&api.TokenCreateRequest{
  Policies: strings.Split(policies, ","),
  Metadata: map[string]string{
    "host_ip":    pod.Status.HostIP,
    "namespace": pod.Metadata.Namespace,
    "pod_ip":     pod.Status.PodIP,
    "pod_name":   pod.Metadata.Name,
    "pod_uid":    pod.Metadata.Uid,
  },
  DisplayName: pod.Metadata.Name,
  Period:       ttl,
  NoParent:     true,
  TTL:          ttl,
}
```

INIT CONTAINER

POD DEFINITION

```
spec:
  replicas: 1
  template:
    metadata:
      annotations:
        vaultproject.io/policies: default
        vaultproject.io/ttl: "24h"
        pod.alpha.kubernetes.io/init-containers: '[{
            "name": "vault-init",
            "image": "kelseyhightower/vault-init:0.0.1",
            "env": [
              {
                "name": "POD_NAME",
                "valueFrom": {"fieldRef": {"fieldPath": "metadata.name"}}
              },
              {
                "name": "POD_NAMESPACE",
                "valueFrom": {"fieldRef": {"fieldPath": "metadata.namespace"}}
              }            ],
...
            "volumeMounts": [
              {
                "name": "vault-token",
                "mountPath": "/var/run/secrets/vaultproject.io"}]}]'
    spec:
      containers:
        - name: vault-example
          image: "kelseyhightower/vault-example:0.0.1"
```

POD INITIALIZATION

- Request wrapped token from vault-controller,
- listen for wrapped token,
- save token to local file and exit.
- Main container starts

SOURCES

- Kelsey Hightower - kube-vault PoC
- Vault wrapping