MIRANTIS | LABS

Tech Talks

Secret Management

Using HashiCorp's Vault and Their New Secrets Operator

Disclaimer

The content contained herein is for informational purposes only, may not be referenced or added to any contract, and should not be relied upon to make purchasing decisions. It is not a commitment, promise, or legal obligation to provide any features, functionality, capabilities, code, etc. or to provide anything within any schedule, date, time, etc. All Mirantis product and service decisions remain at Mirantis sole and exclusive discretion.

Whoami

Martin Nirtl

Solutions Architect

I am an IT engineer 👨‍💻 with strong backgrounds in software, DevOps/platform and electronic engineering working for Mirantis as a pre-sales solution architect. Next to my job, my main side-hustles are all around Kubernetes ⛵, IaC and automating things. From time to time, I even build little apps in Go or other languages.

martinnirtl martinnirtl

martinnirtl

Slides, code and stuff

What we will cover today

- Secret Management
- Kubernetes Secrets
- Challenges & Considerations
- HashiCorp Vault
- Demo
  - Exploring Vault
  - Installing Vault Secret Operator & Retrieving a Secret
- Q&A

Secret Management

Let’s frame it!

Secret Management

Definition

- Securely store, access and manage sensitive information
  - Passwords
  - API keys
  - Cryptographic keys
  - DB connection URLs
  - …
- Protect from unauthorized access, misuse or exposure

Key Points

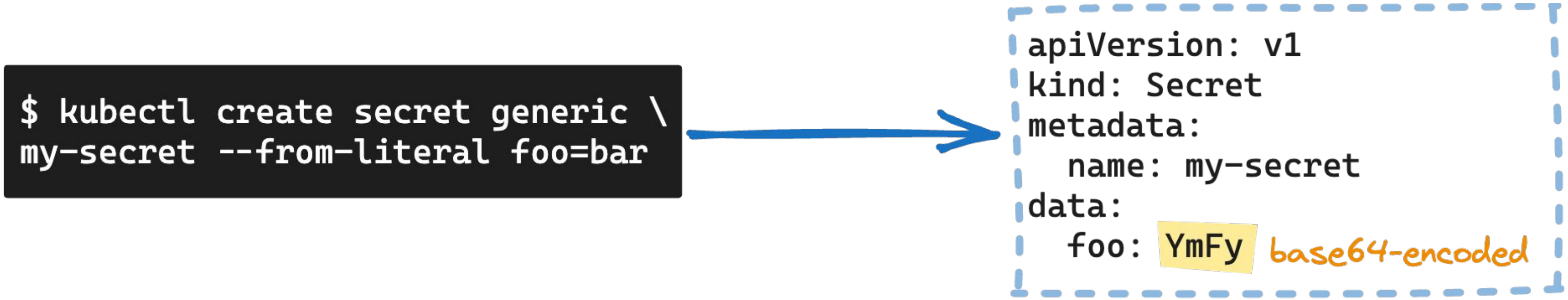
1. Secret Storage
2. Access Control
3. Secure Transmission
4. Rotation and Expiration
5. Audit and Monitoring
6. Automation

Kubernetes Secrets

kubectl explain secrets 🤓

Kubernetes Secrets

- Kubernetes object meant to contain sensitive information
  - Persisted in Kubernetes API
  - Access control via RBAC (namespaced)
- Workloads consume Kubernetes secrets via
  - Environment Variables or Volume Mounts
- Pitfalls like encryption at rest, RBAC, etc.



```
$ kubectl create secret generic \
my-secret --from-literal foo=bar
```

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
data:
  foo: YmFy
```

Secret Management

Challenges & Considerations

Kubernetes Secrets vs. Alternatives

Challenges & Considerations

- Secret management is generally a complex topic
  - Security-related things are always hard!
- Kubernetes secrets are simple
  - But are they an holistic solution? Depends!
  - Check Sealed Secrets
- Alternative solutions like HashiCorp Vault
  - Add functionality (e.g. UI, secret distribution, etc.)
  - Add complexity in terms of secret usage and security
- All solutions have their trade-offs
  - We need to know our **requirements** and understand potential **threats**!

How can secrets be stolen? READ THIS!

Kubernetes Secrets

Read via Kubernetes API
- Requires respective RBAC (SA)
- Via Kubelet config (Kubeconfig)

Read from ETCD directly or its memory
- Requires control-plane node access

Read from memory
- Requires node access

Alternatives

Read from external store (e.g. Vault)
- Imitate Pod w/ right annotations
- Requires to know auth method

~~Read from ETCD directly or its memory~~
- ~~Requires control-plane node access~~

Read from memory
- Requires node access

Caution! These are not complete lists!

“Martin, that’s scary! What should I do?”

Someone

Somewhere

We (or you) need a Strategy!

- Identify risks - What could potentially happen?
  - Improper RBAC (e.g. using Kubernetes Secrets)
  - Hacking attack via CVE
  - …
- Mitigate - How can I prevent/reduce the risk?
  - Policy engines (e.g. OPA Gatekeeper)
  - Audit logs
  - Intrusion detection systems (e.g. Falco)
  - …
- Constantly improve strategy
  - Have easy to follow processes for specific situations (e.g. What to to when …)

HashiCorp Vault

Cloud-agnostic, Open Source Secret Management Solution

HashiCorp Vault

General

- Holistic solution
    - Checks all the key points ✅
- Can be operated on Kubernetes
- Various static & dynamic secret engines
- Supports sophisticated auth methods
- Kubernetes Integrations
    - Agent Sidecar
    - CSI Provider

NEW! Secret Operator

- Public Beta
    - Kubernetes auth only
- Closes the gap towards Kubernetes secrets
- Support for static & dynamic secret engines
- Installation via Helm or Kustomize
- Works with Custom Resources

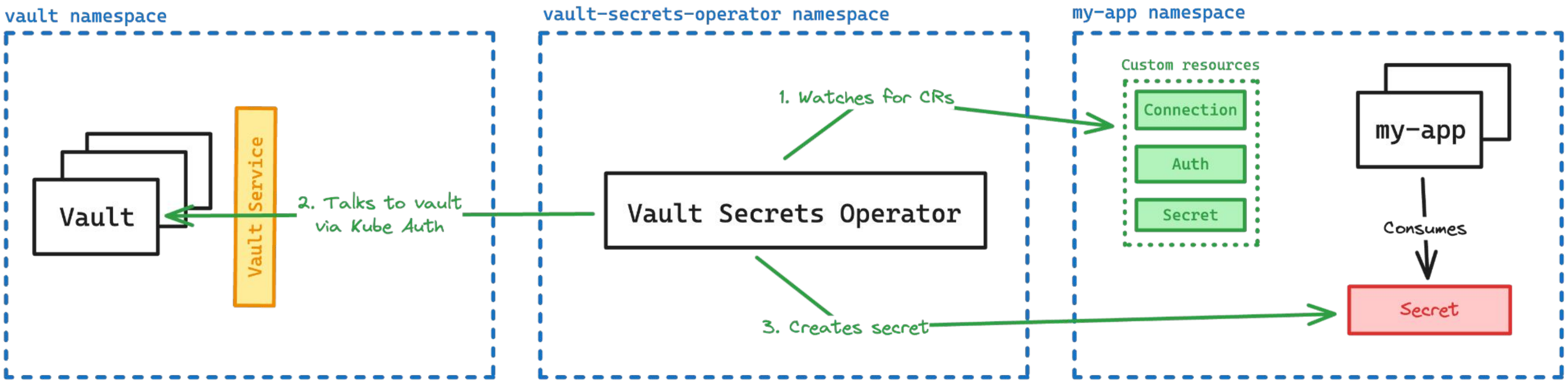
Running Vault in Production on Kubernetes Read more

- Operate Vault in its own cluster or (at least) node-pool
  - Spread Vault Pods using topology spread constraints
- Configure HA storage backend like Consul or Raft (integrated storage)
- Use network attached storage volumes
  - Rebind volumes across nodes in case of node failure
- Use sophisticated auth methods like OIDC/JWT tokens
- Enable TLS

Demo Time ⏰

Vault Secrets Operator

Architecture / Setup


vault namespace
vault-secrets-operator namespace
my-app namespace
Vault
Vault Service
2. Talks to vault via Kube Auth
Vault Secrets Operator
1. Watches for CRs
3. Creates secret
Custom resources
Connection
Auth
Secret
my-app
Consumes
Secret