

Implement Kubernetes Security at Scale



Carsten Duch
Senior Solution Engineer
HashiCorp



Claudio Serrano
Solution Architect
SUSE

Agenda

What is the challenge?

1

Solution Overview

2

Demo

3

Conclusion & QA

4



01



What is the challenge?

KUBERNETES



AND THIS HOW YOU DEPLOY A CONTAINER ON KUBERNETES



imgflip.com

production isn't
stressing me out
anymore.

— Mark, 22 years old



CONTAINER

WE USE
KUBERNETES NOW



BECAUSE IT SIMPLIFIES
OUR OPERATIONS, RIGHT?

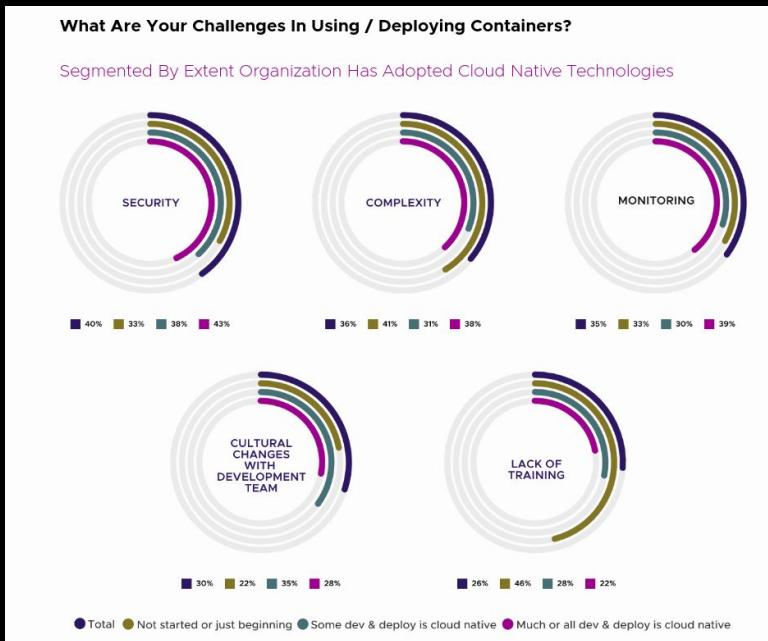


IT SIMPLIFIES
OUR OPERATIONS, RIGHT?

Kubernetes is a tool to
orchestrate containers,
it's not a turn-key platform.



Security is #1 challenge for production containers



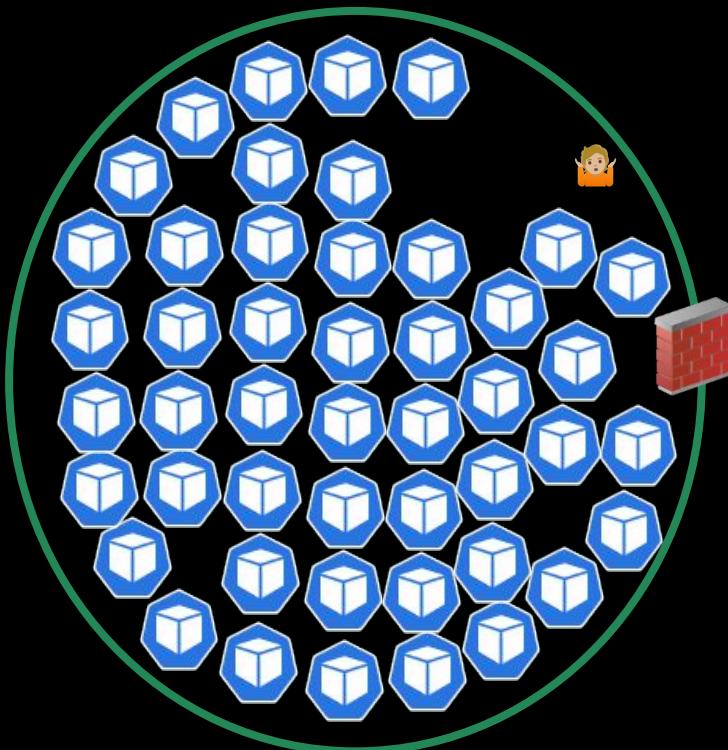
Extract from Cloud Native Computing Foundation Annual Survey 2023
(<https://www.cncf.io/reports/cncf-annual-survey-2023/>)

- Security is the top challenge for organizations running containers in production for two years in a row¹.
- Data breach average costs have been increasing 15% since 2020 reaching USD 4.45 million in 2023.²
- After the xz incident and others, securing the software supply chain continues to be one of the key challenges to prevent attacks.

¹ Cloud Native Computing Foundation Annual Survey 2023 - (<https://www.cncf.io/reports/cncf-annual-survey-2023/>)

² Cost of a Data Breach Report 2023 - (<https://www.ibm.com/security/data-breach>)

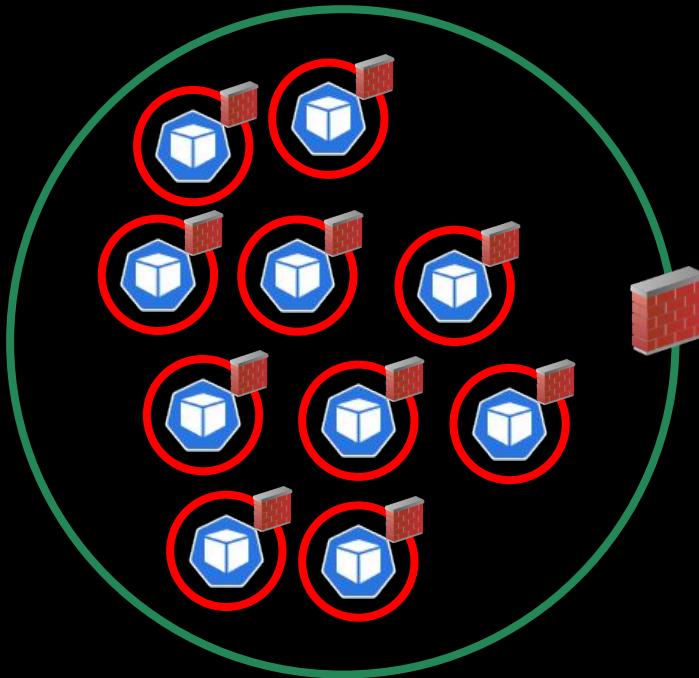
Modern day challenges with container security



- Increasing in Popularity- more attractive for attackers
- Vulnerability & Compliance Management -PCI /NIST /GDPR / HIPAA
- Runtime Security / Zero Trust
- Supply Chain Security - Are the images secure and up to date? - Necessitates an automated approach
- Internal (East-West) security as much (if not more) of a concern than North-South traffic (Perimeter security).
- What is happening in the cluster / Who can communicate with whom
- Container Sprawl - Ephemeral environments / clusters / workloads
- Network is essentially flat *and a bit mysterious*



Modern day challenges with container security



- Re-assign what we consider the security boundary
- Treat each container as its own security boundary
- Automate everything
- Zero-trust as default
- Layer 4 security is not enough
 - Layer 7 inspection/detection
- Falling back to a pure patching practice is not a strategy
 - Threat vs Risk Models
- Leverage abstractions
 - Grouping/Labelling/etc



Kubernetes is ...

... NOT a Secret Management System

- Secrets are stored **unencrypted** by default
- **No secret rotation**
- Secrets value must be base64-encoded
- Secrets are **cluster + namespace** scoped
- Adheres to K8s RBAC model (e.g if you can create a pod, you can see the value of the secret)
- **no dynamic / short lived secrets**
- Hard to audit secret access



Kubernetes is not a secrets management system. It doesn't dynamically generate generic secrets, nor rotate them, but it can securely store, distribute, and expose them to applications using POSIX interfaces such as files on disk, or environment variables.

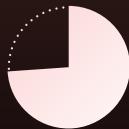
9:48 AM · Oct 28, 2022

Most breaches start with stolen credentials



380%

increase over the last
2 years of data theft
breaches



74%

of breaches involved
a human element



9/10

of web application
breaches consisted
of stolen credentials



62%

of system intrusions
involved a supply
chain

Source: Verizon 2023 Data Breach Investigations report

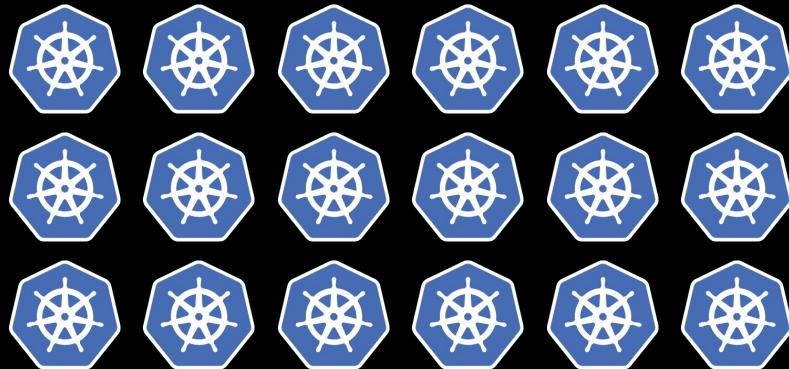
02



Solution Overview

Kubernetes clusters are great ...

... but hard to manage, especially many of them!



- Different environments
- Different teams
- Different hardware
- Different locations
- Edge devices

Rancher Prime

Open/ Interoperable Container Management

- Simple, consistent operations from data center to Edge
- Trusted, secure delivery
- Complete lifecycle management
- Industry leading container security, storage, VM management, and Linux OS
- Access to broad ecosystem of open source technologies

Everything you need to deploy, run, and manage all of your containerized workloads

Full-lifecycle cluster management



Virtual machine management



Distributed storage



Full-lifecycle container security



Operating system management



Infrastructure & Application Monitoring



Certified OS



What do you get with Rancher Prime?

Security

SUSE NeuVector

Vulnerability & Compliance Scanning

Pipeline Security Automation

Zero-Trust Controls & File Protection

Vulnerability (CVE) investigation, triage assistance

Best practices, hardening assistance

Run-time threat rules configuration, optimization

Platform Services

Rancher Prime Manager

App Catalog

Observability

Storage

Service Mesh

Policy Enforcement

Governance

Virtual Machine & OS Management

Cluster Provisioning & LCM

Rancher Prime Hosted

Developer Tools

App Deployment

Local Dev Tooling

Container Images



Infrastructure Services

Any Certified Kubernetes Distribution

Any Enterprise Linux & VMware

Servers

Containers

VM

On-prem

Full Cluster Lifecycle Support for EKS, AKS, & GKE

IaaS

CaaS

Cloud

Lightweight Kubernetes Distribution

Lightweight Linux OS & HCI

Servers

Devices

VM

Edge

Up to 24/7/365 Support & Professional Services

Why Rancher Prime

50,000+

Rancher active users

35,000+

Active Downloads Per Month

10 million+

K3s monthly downloads

Lead

Contributors to
3x CNCF Projects

Trusted by

15,000+
organization
worldwide

SUSE provided Kubernetes Distributions



RKE

**Rancher
Kubernetes Engine**

- 100% Upstream Kubernetes
- CNCF certified
- Easy installation
- Zero-downtime upgrades
- Backup & Disaster Recovery
- Air gapped installation support



RKE 2



K3S

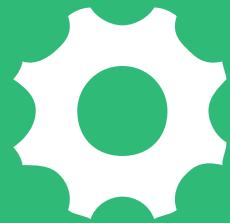


SUSE NeuVector Container Security



Copyright © SUSE 2022

Why SUSE NeuVector?



Automate
Security Policies



Network Visibility
in Production

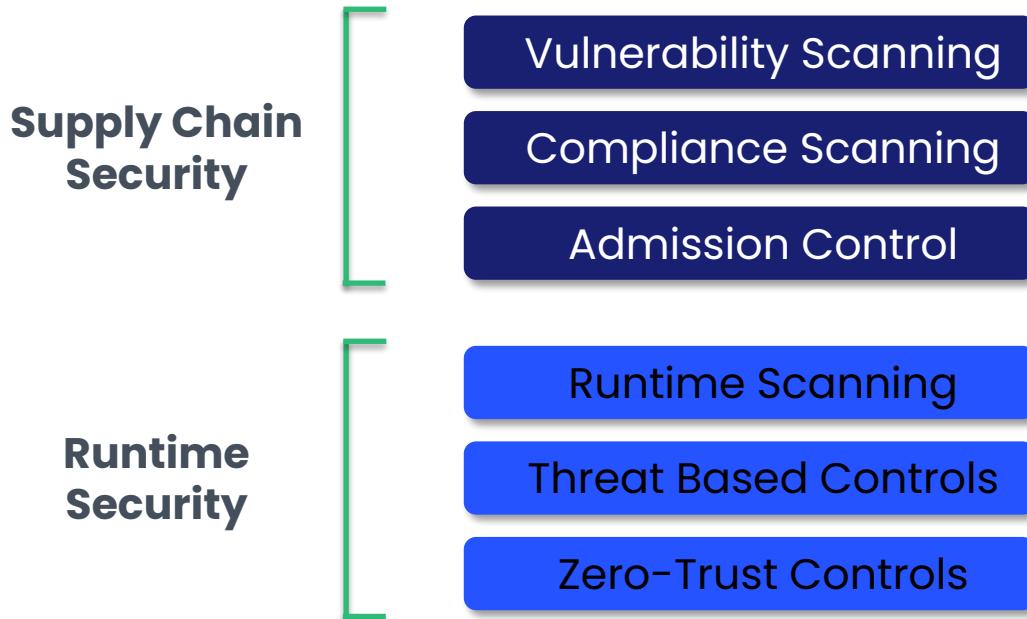


Zero-Trust
Protections –
Network, Process
& File Access



Data Loss
Prevention for
Compliance

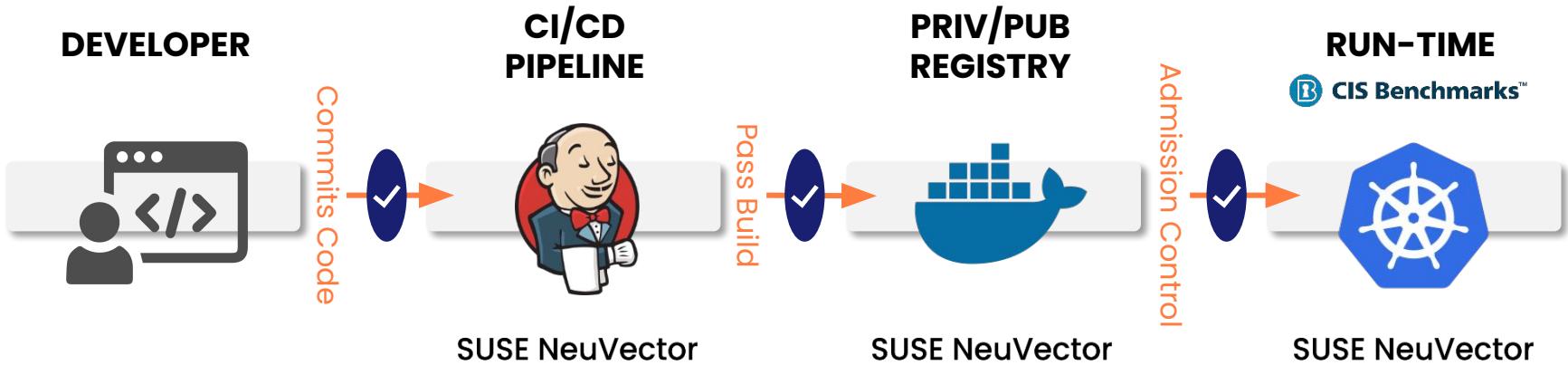
LAYERED SECURITY: DEFENSE IN DEPTH



CVE Database Sources

Source	URL
 nvd and Mitre	https://nvd.nist.gov/feeds/
 SUSE SLE/SLES	https://ftp.suse.com/pub/projects/security/oval/
 Rancher OS	https://rancher.com/docs/os/v1.x/en/about/security/
 Alpine	https://github.com/alpinelinux/alpine-secdb
 Ubuntu	https://launchpad.net/ubuntu-cve-tracker
 Debian	https://security-tracker.debian.org/tracker/data/json
 RedHat	https://www.redhat.com/security/data/oval/
 Amazon	https://alas.aws.amazon.com/
 Busybox	https://www.cvedetails.com/vulnerability-list/
 NGINX	http://nginx.org/en/security_advisories.html
 NodeJS	https://www.npmjs.com/advisories/
 Ruby	https://github.com/rubysec/ruby-advisory-db
 OpenSSL	https://www.openssl.org/news/vulnerabilities.html
 Apache	https://www.cvedetails.com/vendor/45/Apache.html
 Java	https://openjdk.java.net/groups/vulnerability/advisories/
 Python	https://github.com/pyupio/safety-db
 Microsoft Mariner	https://github.com/microsoft/CBL-MarinerVulnerabilityData

VULNERABILITY & COMPLIANCE MANAGEMENT



Run Time Security: Defense in Depth

Threat (signature) Based Controls

CVEs

DLP

Network Attacks

OWASP Top 10

Admission Control

NeuVector Zero-Trust Controls

Automated Learning

Network

Process

File Access

Security as Code



HashiCorp
Vault

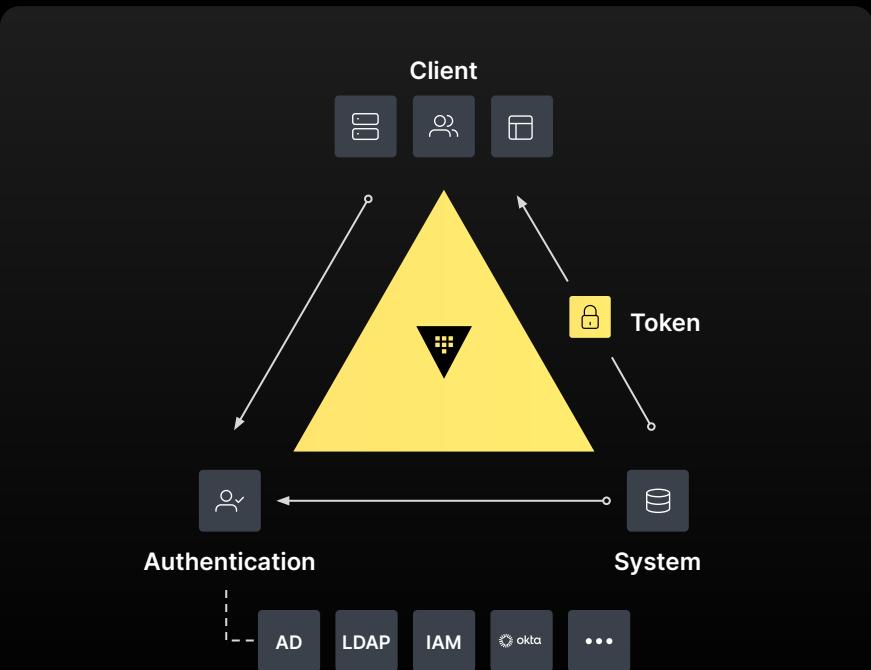
Authenticate and authorize every access request

In dynamic cloud infrastructure, security starts with identity.

Identity-based security uses trusted identities to automate access to secrets, data, and applications

Security system of record to centrally store and protect secrets across clouds and applications

Lifecycle management of your credentials to ensure proper oversight, rotation, and expiry





Automate **secure access** and **lifecycle management** for credentials and sensitive data

Identity-based security

Secrets	Certificates	Keys	Data Protection
Static	PKI	KMS	Encryption
Rotated	Managed Keys	KMIP	Signatures
Dynamic		HSM	Tokenization
Database			



Choose your delivery model



HCP Vault

For small and medium size businesses with cloud-native architectures

- ✓ **HCP Vault Secrets** for secret lifecycle management
- ✓ **HCP Vault Radar** for secret scanning

Cloud

Managed

Self-managed

Choose your delivery model



HCP Vault Dedicated

For large businesses that require a single tenant cloud environment

- 🕒 Secrets Management
- 🕒 Key Management
- 🕒 Certificate Management
- 🕒 Advanced Data Protection

Cloud

Managed

Self-managed

Choose your delivery model



Vault Enterprise

For large businesses with strict compliance and security requirements

- 🕒 Secrets Management
- 🕒 Key Management
- 🕒 Certificate Management
- 🕒 Advanced Data Protection

Cloud

Managed

Self-managed



Kubernetes secrets

[Challenge](#) [Solution](#) [Results](#)

Kubernetes

In the DevOps world, some CI/CD pipelines require the ability to manage applications running on a Kubernetes cluster.

Kubernetes Secrets are, by default, stored unencrypted in the API server's underlying data store (etcd). Anyone with API access can retrieve or modify a Secret, and so can anyone with access to etcd.

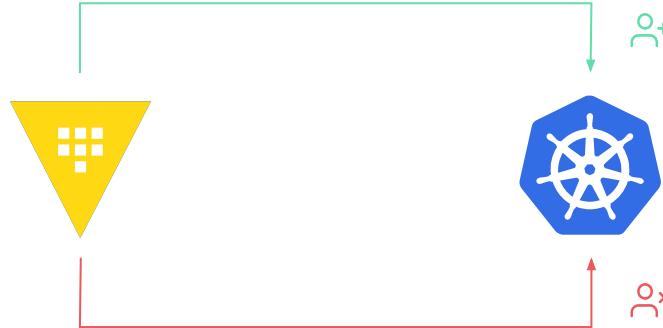


Kubernetes secrets

Challenge **Solution** Results

Kubernetes

The Kubernetes Secrets Engine for Vault generates Kubernetes service account tokens, and optionally service accounts, role bindings, and roles. The created service account tokens have a configurable TTL and any objects created are automatically deleted when the Vault lease expires.



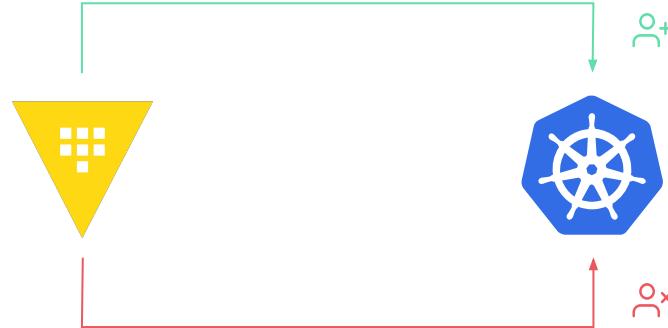
Kubernetes secrets

Challenge **Solution** Results

Sidecar containers

Enable access to Vault secrets by Kubernetes applications that don't have native Vault logic built-in.

- Allow automatic injection of secrets into the pod file system
- Allow applications to only concern themselves with finding a secret at a filesystem path, rather than managing the auth tokens and other mechanisms



Kubernetes secrets

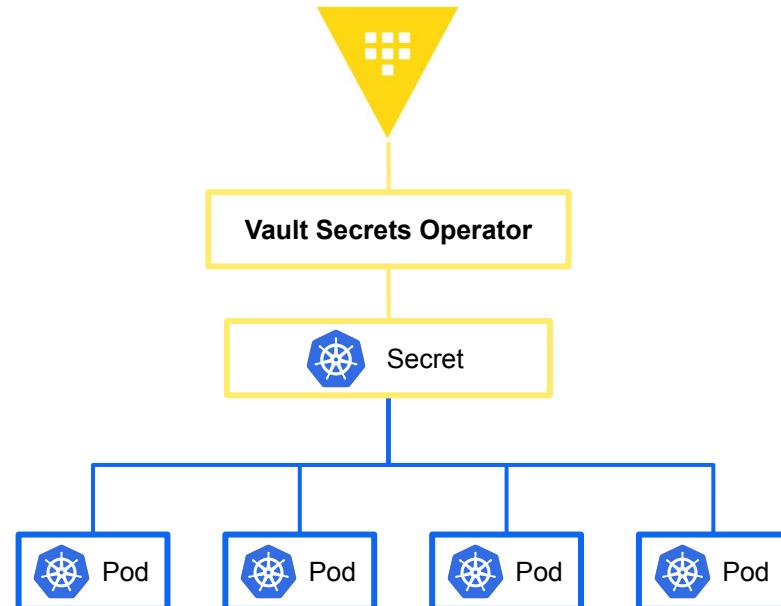
Challenge **Solution** Results

Vault Secrets Operator for Kubernetes

Using controller-runtime and custom resource definitions to allow application developers to consume Vault Secrets directly from Kubernetes. The Operator ensures Vault Secrets remain synchronized in the cluster.

Supports synchronizing all classes of Vault secrets:

- Dynamic: database credentials, cloud credentials, etc.
- PKI: TLS certificates
- Static: Key/Value
- Google auth for Workload Identity on GKE
- OpenShift OLM



How to Consume Vault Secrets in Kubernetes

- Vault Secrets Operator (VSO): Sync secrets from Vault into k8s “secret”-ressource
 - <https://developer.hashicorp.com/vault/tutorials/kubernetes/vault-secrets-operator>
- Vault Agent Injector (VAI): Inject secret from Vault directly into a Pod
 - <https://developer.hashicorp.com/vault/tutorials/kubernetes/kubernetes-sidecar>

Alternative Options:

- CSI: Mount Secret through Container Storage Interface (CSI)
 - <https://developer.hashicorp.com/vault/tutorials/kubernetes/kubernetes-secret-store-driver>
- API: Applications can use Vault API Library's to consume Secrets from Vault
 - <https://developer.hashicorp.com/vault/api-docs/libraries>

Blogpost:

<https://www.hashicorp.com/blog/kubernetes-vault-integration-via-sidecar-agent-injector-vs-csi-provider>



03

Demo





HashiCorp
Vault

Setup Static Secret in Vault

The screenshot shows the HashiCorp Vault web interface. On the left, a dark sidebar lists various navigation items: Vault, Dashboard, Secrets Engines (selected), Secrets Sync, Enterprise, Access, Policies, Tools, Monitoring, Client Count, and Seal Vault. The main content area displays a path: secrets / kvv2 / webapp / config. Below this, the title "webapp/config" is shown above a table with tabs for Secret, Metadata, Paths, and Version History. The "Secret" tab is selected. A JSON toggle switch is on, and the table has columns for Key and Value. There are two rows: "password" and "username". Each row contains a key, a value field with a redacted password/username, and three icons: a copy icon, a download icon, and an eye icon. A timestamp at the bottom right indicates "Version 3 created Jun 11, 2024 02:03 PM".

Key	Value	Actions		
password	██████████			
username	██████████			

Version 3 created Jun 11, 2024 02:03 PM

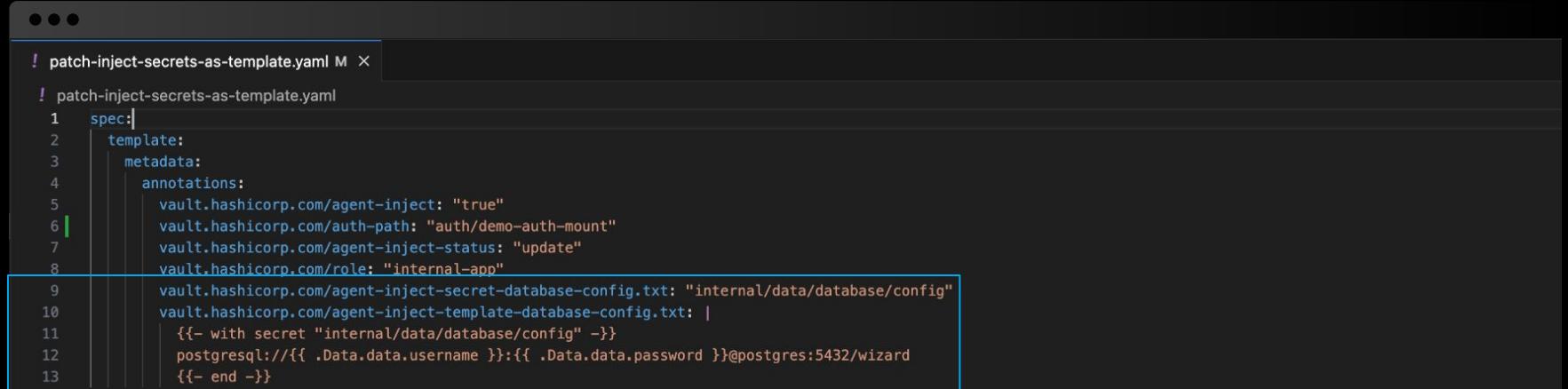
VSO: Configure Secret Sync via CRD

```
! static-secret.yaml •
Users > cduch > Documents > git > hashicorp-education > learn-vault-secrets-operator > vault > ! static-secret.yaml
1  apiVersion: secrets.hashicorp.com/v1beta1
2  kind: VaultStaticSecret
3  metadata:
4    name: vault-kv-app
5    namespace: app
6  spec:
7    type: kv-v2
8
9    # mount path
10   mount: kvv2
11
12   # path of the secret
13   path: webapp/config
14
15   # dest k8s secret
16   destination:
17     name: secretkv
18     create: true
19
20   # static secret refresh interval
21   refreshAfter: 30s
22
23   # Name of the CRD to authenticate to Vault
24   vaultAuthRef: static-auth
25
```

VSO: Access Synced Secret in Kubernetes

The screenshot shows the HashiCorp Vault UI interface. On the left is a sidebar with various navigation items like Cluster, Workloads, Apps, Service Discovery, Storage, PersistentVolumes, StorageClasses, ConfigMaps, PersistentVolumeClaims, Secrets, Policy, Monitoring, Logging, CIS Benchmark, Istio, Kubewarden, Longhorn, NeuVector, and More Resources. The 'Secrets' item is currently selected and highlighted in blue. The main panel displays a secret named 'secretkv' which is active. It shows the namespace as 'app' and an age of '1.1 days'. The secret type is 'Secret' and it has labels: app.kubernetes.io/component:secret-sync, app.kubernetes.io/managed-by:hashicorp-vso, app.kubernetes.io/name:vault-secrets-operator, and secrets.hashicorp.com/vso-ownerRefUID:32d45d90-bde1-4531-b9a9-696ba5fc0588. Below this, there are tabs for Data, Recent Events, and Related Resources, with 'Data' being the active tab. The data section shows three fields: '_raw', 'password', and 'username'. The '_raw' field contains a large amount of sensitive data represented by a grid of orange and purple dots. The 'password' field shows a redacted password value: '.....'. The 'username' field also shows a redacted value: '.....'. Each of these fields has a 'Copy' button to its right.

VAI: Inject Secret with Template into Pod



```
patch-inject-secrets-as-template.yaml M X
patch-inject-secrets-as-template.yaml
1 spec:
2   template:
3     metadata:
4       annotations:
5         vault.hashicorp.com/agent-inject: "true"
6         vault.hashicorp.com/auth-path: "auth/demo-auth-mount"
7         vault.hashicorp.com/agent-inject-status: "update"
8         vault.hashicorp.com/role: "internal-app"
9         vault.hashicorp.com/agent-inject-secret-database-config.txt: "internal/data/database/config"
10        vault.hashicorp.com/agent-inject-template-database-config.txt: |
11          {{- with secret "internal/data/database/config" -}}
12            postgresql://{{ .Data.data.username }}:{{ .Data.data.password }}@postgres:5432/wizard
13          {{- end -}}
```

VAI: Show Injected Secret In Pod

The screenshot shows the HashiCorp Vault UI interface. On the left, there's a sidebar with navigation buttons: Home, CEO (which is selected), DTE, DOR, DNS, LK8, and NJT. Below these are sections for Cluster, Workloads, CronJobs, DaemonSets, Deployments (selected, showing 3 items), Jobs, StatefulSets, and Pods (showing 4 items). A 'Cluster Tools' button is also present. The main content area shows a deployment named 'orgchart' in the 'vault' namespace, which is active and has been running for 13 days with 0 pod restarts. It uses the image 'jweissig/app:0.0.1' and has 1 ready pod, 1 up-to-date pod, and 1 available pod. Labels include 'app: orgchart'. Annotations link to 'Show 2 annotations'. A 'Pods by State' section shows 1 pod running. At the bottom, a terminal window displays the command 'cat /vault/secrets/database-config.txt' and its output: 'postgresql://db_READONLY_USERNAME:db_SECRET_PASSWORD@postgres:5432/wizard'. The UI version is v2.8.2.

Deployment: orgchart (Active)

Namespace: vault Age: 13 days Pod Restarts: 0

Image: jweissig/app:0.0.1 Ready: 1/1 Up-to-date: 1 Available: 1

Labels: app:orgchart

Annotations: Show 2 annotations

Pods by State

1 Running

v2.8.2

↳ orgchart-7d598cf79f-9dpmw [X] ⌂

```
/app # cat /vault/secrets/database-config.txt
postgresql://db_READONLY_USERNAME:db_SECRET_PASSWORD@postgres:5432/wizard/app #
```

04



Conclusion & QA

Conclusion - SUSE & HashiCorp: Better Together!

SUSE Rancher Prime

Streamlines cluster deployment, offering centralized authentication, access control and observability across your deployments anywhere.

SUSE Neuvector Prime

Providing end-to-end vulnerability management for the full container lifecycle for your Kubernetes environments.

HashiCorp Vault

Providing identity-based security to automatically authenticate and authorize access to secrets and other sensitive data for Kubernetes Applications.



Events

HashiConf 2024

October 14-16, 2024 | Boston

HashiConf is HashiCorp's global cloud conference. Join us for 2+ days of conversations on the future of cloud automation with product announcements, technical sessions, hands-on labs, certifications, social events, and more.

<https://www.hashicorp.com/conferences/hashiconf>

SUSE Rancher and SUSE Neuvvector Rodeos:

https://www.suse.com/events/?event_type=rodeos





Thank you

hello@hashicorp.com