



Secret Management für SUSE Rancher mit HCP Vault

Eine Geschichte vom nicht aufgeben wollen ...





Carsten Duch

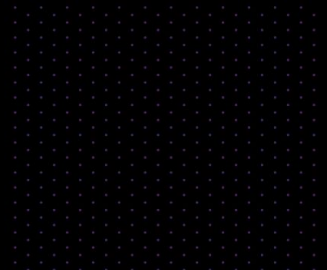
Sr. Solutions Engineer at HashiCorp
he/him

carsten@hashicorp.com



— 01

**Was wollen wir
erreichen?**



Secret Management mit HCP Vault



für Kubernetes Workloads in Rancher RKE Clustern

1. Workloads in Rancher RKE / K3S benötigen Secret Management
2. Applikationen sollen nicht extra “umprogrammiert” werden.
3. Für das Secret Management ist HCP Vault die ideale Lösung



HashiCorp
Vault

Wie kommen die Secrets in den Pod?



Vault Agent Sidecar Injector

- Secrets werden per “Agent Sidecar Injector” in den Pod gelegt
- Applikationen müssen nicht extra “umprogrammiert” werden
- Benötigt im Cluster lediglich einen Vault Injector Service
- Sonstige Konfiguration via Annotations

annotations:

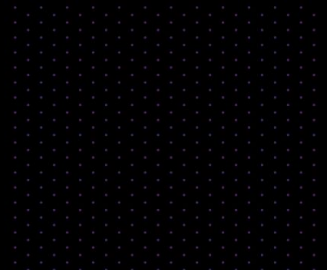
```
vault.hashicorp.com/agent-inject: "true"
```

```
vault.hashicorp.com/agent-inject-secret-hello: "secret/hello"
```

```
vault.hashicorp.com/role: "vault-app"
```

— 02

Was ist das Problem?





Learn-Guide

- **Was es benutzt:**
K8s via Minikube
& lokales Vault
- **Was wir wollen:**
Rancher RKE
& HCP Vault

The screenshot shows a web browser displaying the HashiCorp Learn page for the tutorial "Integrate a Kubernetes Cluster with an External Vault". The page has a sidebar on the left with a "Vault" header and a "Kubernetes" section containing a list of tutorials. The main content area has a title "Integrate a Kubernetes Cluster with an External Vault", a "15 MIN" duration indicator, and a "PRODUCTS USED: Vault" tag. The text explains that Vault can manage secrets for Kubernetes application pods from outside the cluster. It includes a highlighted section "Running Vault in Kubernetes" and a "Prerequisites" section. The footer of the page indicates the tutorial was last tested on 03 Jun 2021 on macOS 11.4.

Vault

Kubernetes

- Vault Installation to Minikube via Helm
- Vault Installation to Red Hat OpenShift via Helm
- Injecting Secrets into Kubernetes Pods via Vault Agent Containers
- **Integrate a Kubernetes Cluster with an External Vault**
- Vault Installation to Google Kubernetes Engine via Helm
- Vault Installation to Azure Kubernetes Service via Helm
- Vault Installation to Amazon Elastic Kubernetes Service via Helm
- Mount Vault Secrets through Container Storage Interface (CSI) Volume

Jump to section ▾ Docs Forum Bookmark

Integrate a Kubernetes Cluster with an External Vault

🕒 15 MIN PRODUCTS USED: ▼ Vault

Vault can manage secrets for Kubernetes application pods from outside the cluster. This could be [HashiCorp Cloud Platform \(HCP\) Vault](#) or another Vault service within your organization.

Running Vault in Kubernetes: Vault run in the cluster is explored in the [Vault Installation to Minikube via Helm](#) and [Injecting Secrets into Kubernetes Pods via Vault Helm Sidecar](#) tutorials.

In this tutorial, you will run Vault locally, start a Kubernetes cluster with Minikube, deploy an application that retrieves secrets directly from Vault, through a Kubernetes service, and through secret injection via Vault Agent Injector.

Prerequisites

This tutorial requires the [Kubernetes command-line interface \(CLI\)](#) and the [Helm CLI](#) installed, [Minikube](#), [Vault](#), and the [Vault Helm chart](#), the sample web application, and additional configuration to bring it all together.

This tutorial was last tested 03 Jun 2021 on a macOS 11.4 using this configuration.

Vault version.

<https://learn.hashicorp.com/tutorials/vault/kubernetes-external-vault?in=vault/kubernetes>

Learn-Guide funktioniert nicht mit Rancher RKE und Vault HCP



- **Rancher RKE**

Rancher Serv

Verbindung g

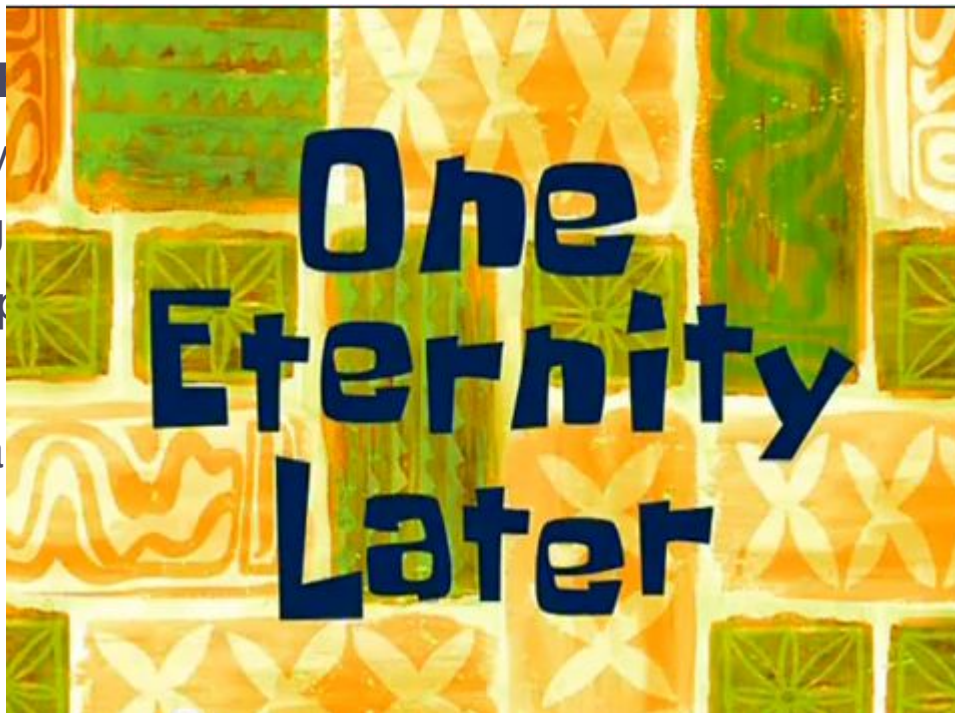
Ist nicht komp

- **HCP Vault**

Kein Zugriff a

- **Demo Apps**

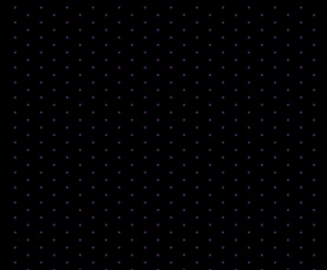
Unterstützen



d
en Cluster

— 03

Wie lösen wir es?



Rancher hat mehrere Auth-Methoden



für basierte RKE und K3S Cluster

1. **Via Rancher Server Authentication Proxy**

Verbindung geht über den Proxy, dann erst zum eigentlichen Cluster

2. **Direkt mit dem Cluster via Kubernetes API**

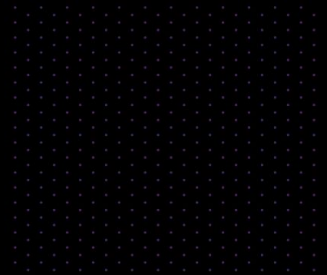
Funktioniert wie bei anderen Kubernetes Clustern

Muss ggf. aber erst aktiviert werden

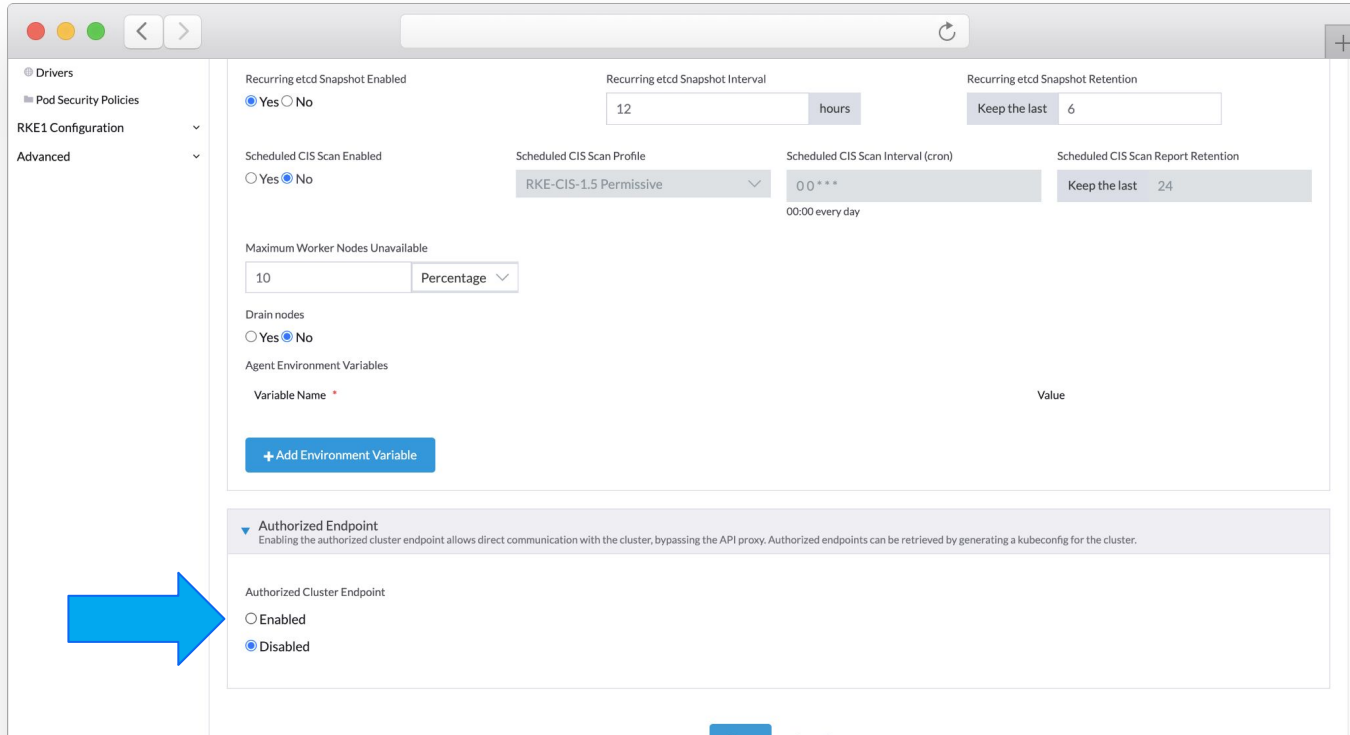
-> **Sicherstellen das via Kubernetes API authentifiziert wird.
UND den richtigen Namespace ansprechen :-)**

— 04

DEMO



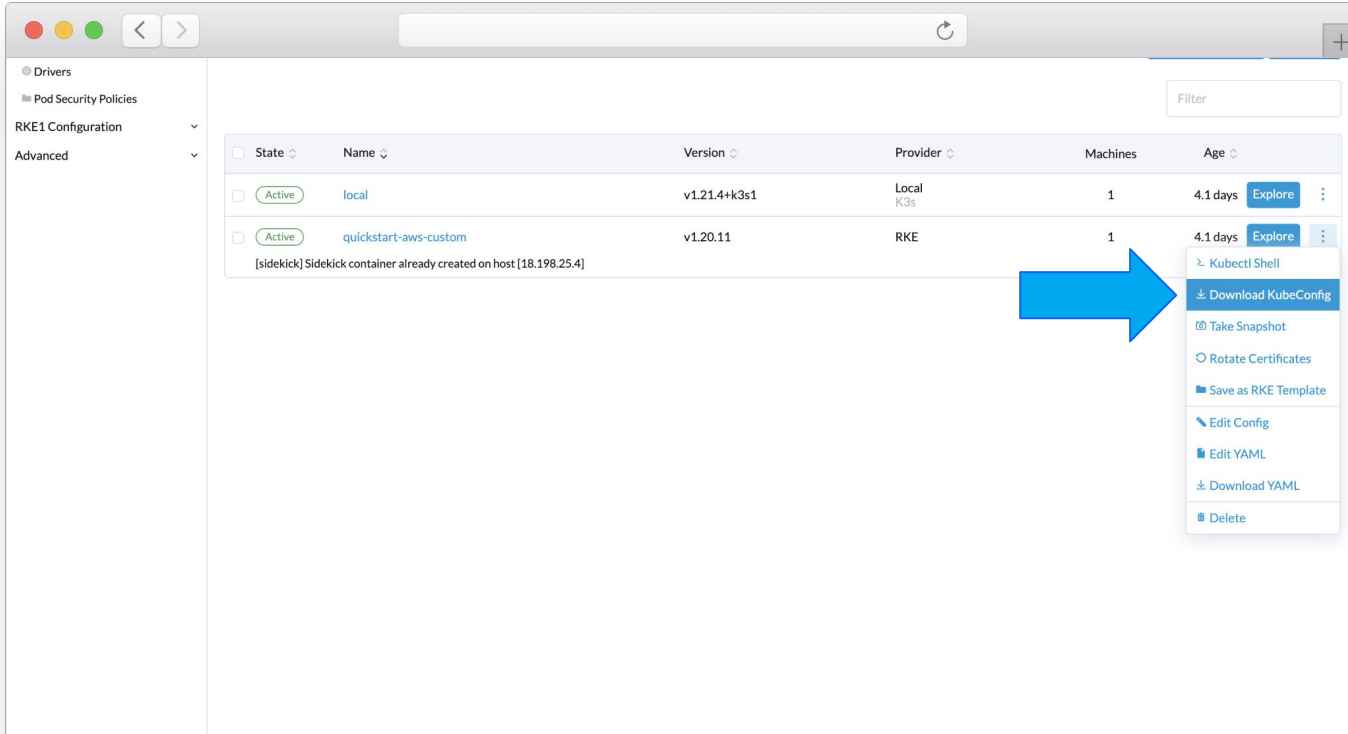
1. Enable Authorized Cluster Endpoint



The screenshot shows the Rancher UI configuration page for RKE1. The left sidebar contains a menu with 'Drivers', 'Pod Security Policies', 'RKE1 Configuration', and 'Advanced'. The main content area is divided into several sections:

- Recurring etcd Snapshot Enabled:** ☒ Yes ☐ No
- Recurring etcd Snapshot Interval:** 12 hours
- Recurring etcd Snapshot Retention:** Keep the last 6
- Scheduled CIS Scan Enabled:** ☐ Yes ☒ No
- Scheduled CIS Scan Profile:** RKE-CIS-1.5 Permissive
- Scheduled CIS Scan Interval (cron):** 0 0 * * * 00:00 every day
- Scheduled CIS Scan Report Retention:** Keep the last 24
- Maximum Worker Nodes Unavailable:** 10 Percentage
- Drain nodes:** ☐ Yes ☒ No
- Agent Environment Variables:** A table with columns 'Variable Name' and 'Value'. A blue button '+ Add Environment Variable' is located below the table.
- Authorized Endpoint:** A section with a description: 'Enabling the authorized cluster endpoint allows direct communication with the cluster, bypassing the API proxy. Authorized endpoints can be retrieved by generating a kubeconfig for the cluster.' Below this, there are two radio buttons: ☐ Enabled and ☒ Disabled. A large blue arrow points to the 'Disabled' radio button.

2. Download KubeConfig



The screenshot shows the Rancher UI interface. On the left, the sidebar is expanded to 'Drivers' > 'Pod Security Policies' > 'RKE1 Configuration' > 'Advanced'. The main area displays a table of RKE1 configurations. A context menu is open for the 'quickstart-aws-custom' entry, and a blue arrow points to the 'Download KubeConfig' option.

<input type="checkbox"/>	State	Name	Version	Provider	Machines	Age	Actions
<input type="checkbox"/>	Active	local	v1.21.4+k3s1	Local K3s	1	4.1 days	Explore
<input type="checkbox"/>	Active	quickstart-aws-custom	v1.20.11	RKE	1	4.1 days	Explore

[sidekick] Sidekick container already created on host [18.198.25.4]

- Kubectl Shell
- Download KubeConfig**
- Take Snapshot
- Rotate Certificates
- Save as RKE Template
- Edit Config
- Edit YAML
- Download YAML
- Delete

3. Set context to Kubernetes API Auth



```
$ kubectl config get-contexts
```

CURRENT	NAME	CLUSTER
*	quickstart-aws-custom	quickstart-aws-custom
	quickstart-aws-custom-ip-172-31-13-222	quickstart-aws-custom-ip..

```
$ kubectl config use-context quickstart-aws-custom-ip-172-31-13-222
```

```
Switched to context "quickstart-aws-custom-ip-172-31-13-222".
```

```
$ kubectl config get-contexts
```

CURRENT	NAME	CLUSTER
	quickstart-aws-custom	quickstart-aws-custom
*	quickstart-aws-custom-ip-172-31-13-222	quickstart-aws-custom-ip..

4. Prepare exports



```

$ export KUBE_HOST="https://1.2.3.4:6443"
$ export EXTERNAL_VAULT_ADDRESS='https://vault-cluster.vault.XXX.aws.hashicorp.cloud:8200'
$ export VAULT_ADDR='https://vault-cluster.vault.XXX.aws.hashicorp.cloud:8200'

```


5. Create Namespace & Service Account



```
$ kubectl create ns vault-test
```

```
$ kubectl apply -f sa.yaml
```

6. Enable Kubernetes Auth in Vault



```
$ vault auth enable kubernetes
```

```
$ export TOKEN_REVIEW_JWT="$(kubectl get secret vault-auth -n vault-test -o  
go-template='{{ .data.token }}' | base64 --decode)"
```

```
$ kubectl get secret vault-auth -n vault-test -o go-template='{{ index  
.data "ca.crt" }}' | base64 --decode > vault-ca.crt
```

```
$ vault write auth/kubernetes/config \  
    token_reviewer_jwt="$TOKEN_REVIEW_JWT" \  
    kubernetes_host="$KUBE_HOST" \  
    kubernetes_ca_cert=@vault-ca.crt
```

7. Create an App Policy in Vault



```

$ vault policy write internal-app-pol internal-app-policy.hcl

$ vault write auth/kubernetes/role/vault-app \
    bound_service_account_names=internal-app \
    bound_service_account_namespaces=vault-test \
    policies=internal-app-pol \
    ttl=24h

```

8. Install the Vault Sidecar Injector



```

$ helm repo add hashicorp https://helm.releases.hashicorp.com
$ helm repo update
$ helm upgrade --install vault hashicorp/vault \
  -n vault-test \
  --version 0.9.1 \
  --set "injector.externalVaultAddr=${EXTERNAL_VAULT_ADDRESS}" \
  --set "injector.authPath=auth/kubernetes" \
  --set "server.serviceAccount.name=vault-auth" \
  --set "server.serviceAccount.create=false" \
  --wait
```

9. Test it



TERMINAL

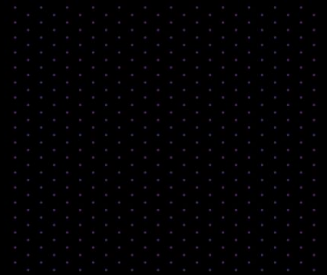
```
# create a test secret
$ vault kv put secret/hello foo=world

# deploy a demo app
$ kubectl apply -f deploy.yaml -n vault-test

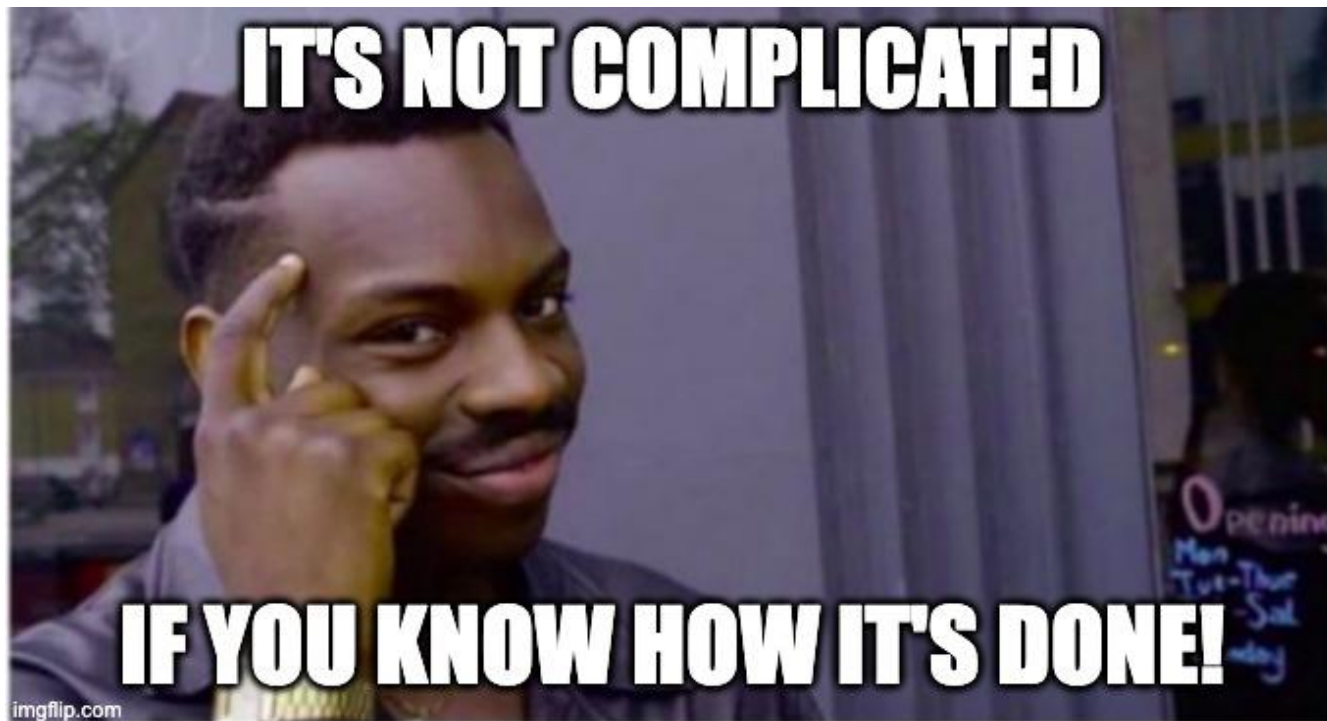
# get the pod id
$ kubectl get pods -n vault-test

# check for the secret in the pod
$ kubectl exec -it nginx-58dddb9876-f2kgp cat /vault/secrets/hello -n
vault-test
```

Fazit



Rancher RKE & HCP Vault



Referenzen



- **Slides und Infos im Git Repo zum Vortrag**

<https://github.com/cduch/hcpvault-rancher-demo> oder <https://bit.ly/31Tcbsr>

- **Learn Guides**

<https://learn.hashicorp.com/tutorials/vault/kubernetes-sidecar?in=vault/kubernetes>

<https://learn.hashicorp.com/tutorials/vault/kubernetes-external-vault?in=vault/kubernetes>

- **Rancher Authorized Cluster Endpoint (ACE)**

<https://rancher.com/docs/rancher/v2.6/en/cluster-admin/cluster-access/ace/>

<https://rancher.com/docs/rancher/v2.6/en/cluster-admin/editing-clusters/rke-config-reference/#authorized-cluster-endpoint>



Thank You

hugs@hashicorp.com | learn.hashicorp.com | discuss.hashicorp.com



HashiTalks

hugs@hashicorp.com | learn.hashicorp.com | discuss.hashicorp.com