SecureVault Kubernetes Integration Guide

Document ID: PRD-SV-035 **Last Updated:** 2024-02-25 **Owner:** SecureVault Product Team **Classification:** Public

---

Overview

SecureVault integrates with Kubernetes to provide secure secrets management for containerized workloads. This guide covers authentication methods, secrets injection, and best practices for Kubernetes environments.

Integration Methods

Method Comparison

Method	Use Case	Complexity	Security
Sidecar Injector	Legacy apps, no code changes	Low	High
CSI Driver	Native K8s secrets	Medium	High
External Secrets	GitOps workflows	Medium	High
Direct SDK	Custom integrations	High	Highest

Kubernetes Authentication

Service Account Authentication

Configure Kubernetes auth method:

```
# Enable Kubernetes auth
securevault auth enable kubernetes

# Configure Kubernetes auth
securevault write auth/kubernetes/config \
  kubernetes_host="https://kubernetes.default.svc" \
  kubernetes_ca_cert=@/var/run/secrets/kubernetes.io/serviceaccount/ca.crt
```

Create Roles

```
# Create role for application
securevault write auth/kubernetes/role/my-app \
  bound_service_account_names=my-app-sa \
  bound_service_account_namespaces=default \
  policies=my-app-policy \
  ttl=24h
```

Service Account Setup

```
# service-account.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-app-sa
  namespace: default
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: my-app-tokenreview
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: system:auth-delegator
subjects:
  - kind: ServiceAccount
    name: my-app-sa
    namespace: default
```

Sidecar Injector

Installation

```
# Add Helm repository
helm repo add securevault https://charts.novatech.com

# Install injector
helm install securevault-injector securevault/vault-sidecar-injector \
  --namespace securevault \
  --set vault.address=https://securevault.novatech.com \
  --set vault.authPath=auth/kubernetes
```

Pod Annotations

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-app
  annotations:
    securevault.novatech.com/inject: "true"
    securevault.novatech.com/role: "my-app"
    securevault.novatech.com/secrets: |
      secret/data/my-app/config
    securevault.novatech.com/secret-volume-path: "/vault/secrets"
spec:
  serviceAccountName: my-app-sa
  containers:
    - name: my-app
      image: my-app:latest
      volumeMounts:
        - name: secrets
          mountPath: /vault/secrets
          readOnly: true
```

Annotation Reference

Annotation	Description	Default
`inject`	Enable injection	false
`role`	SecureVault role	required
`secrets`	Secret paths	required
`secret-volume-path`	Mount path	/vault/secrets
`template`	Custom template	-
`agent-inject-token`	Include token	false

Template Support

```yaml
annotations:
  securevault.novatech.com/template: |
    {{- with secret "secret/data/my-app/db" -}}
    DATABASE_URL=postgres://{{ .Data.data.username }}:{{ .Data.data.password }}@db:5432/mydb
    {{- end }}
```

CSI Driver

Installation

```
# Install SecureVault CSI driver
helm install securevault-csi securevault/vault-csi-provider \
  --namespace securevault \
  --set vault.address=https://securevault.novatech.com
```

SecretProviderClass

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: my-app-secrets
spec:
  provider: vault
  parameters:
    vaultAddress: "https://securevault.novatech.com"
    roleName: "my-app"
    objects: |
      - objectName: "db-password"
        secretPath: "secret/data/my-app/db"
        secretKey: "password"
      - objectName: "api-key"
        secretPath: "secret/data/my-app/api"
        secretKey: "key"
  secretObjects:
    - secretName: my-app-secrets
      type: Opaque
      data:
        - objectName: db-password
          key: DB_PASSWORD
        - objectName: api-key
          key: API_KEY
```

Pod Configuration

```
apiVersion: v1
kind: Pod
metadata:
  name: my-app
spec:
  serviceAccountName: my-app-sa
```

```yaml
  containers:
    - name: my-app
      image: my-app:latest
      envFrom:
        - secretRef:
            name: my-app-secrets
      volumeMounts:
        - name: secrets-store
          mountPath: "/mnt/secrets"
          readOnly: true
  volumes:
    - name: secrets-store
      csi:
        driver: secrets-store.csi.k8s.io
        readOnly: true
        volumeAttributes:
          secretProviderClass: my-app-secrets
```

External Secrets Operator

Installation

```bash
# Install External Secrets Operator
helm install external-secrets external-secrets/external-secrets \
  --namespace external-secrets \
  --create-namespace
```

ClusterSecretStore

```yaml
apiVersion: external-secrets.io/v1beta1
kind: ClusterSecretStore
metadata:
  name: securevault
spec:
  provider:
    vault:
      server: "https://securevault.novatech.com"
      path: "secret"
      version: "v2"
      auth:
        kubernetes:
          mountPath: "kubernetes"
```

```
          role: "external-secrets"
          serviceAccountRef:
            name: external-secrets
            namespace: external-secrets
```

ExternalSecret

```yaml
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: my-app-secrets
  namespace: default
spec:
  refreshInterval: 1h
  secretStoreRef:
    name: securevault
    kind: ClusterSecretStore
  target:
    name: my-app-secrets
    creationPolicy: Owner
  data:
    - secretKey: DB_PASSWORD
      remoteRef:
        key: secret/data/my-app/db
        property: password
    - secretKey: API_KEY
      remoteRef:
        key: secret/data/my-app/api
        property: key
```

Dynamic Secrets

Database Credentials

```yaml
# SecretProviderClass for dynamic DB credentials
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: db-creds
spec:
  provider: vault
  parameters:
```

```
    vaultAddress: "https://securevault.novatech.com"
    roleName: "my-app"
    objects: |
      - objectName: "db-creds"
        secretPath: "database/creds/my-app-role"
        secretKey: "username"
      - objectName: "db-password"
        secretPath: "database/creds/my-app-role"
        secretKey: "password"
```

AWS Credentials

```
# Dynamic AWS credentials
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: aws-creds
spec:
  refreshInterval: 30m
  secretStoreRef:
    name: securevault
    kind: ClusterSecretStore
  target:
    name: aws-creds
  data:
    - secretKey: AWS_ACCESS_KEY_ID
      remoteRef:
        key: aws/creds/my-app-role
        property: access_key
    - secretKey: AWS_SECRET_ACCESS_KEY
      remoteRef:
        key: aws/creds/my-app-role
        property: secret_key
```

Best Practices

Pod Security

```
apiVersion: v1
kind: Pod
metadata:
  name: my-app
```

```yaml
spec:
  serviceAccountName: my-app-sa
  securityContext:
    runAsNonRoot: true
    runAsUser: 1000
    fsGroup: 1000
  containers:
    - name: my-app
      image: my-app:latest
      securityContext:
        allowPrivilegeEscalation: false
        readOnlyRootFilesystem: true
        capabilities:
          drop:
            - ALL
      volumeMounts:
        - name: secrets
          mountPath: /secrets
          readOnly: true
```

Namespace Isolation

```
# Create role per namespace
securevault write auth/kubernetes/role/app-production \
  bound_service_account_names=* \
  bound_service_account_namespaces=production \
  policies=production-secrets \
  ttl=1h

securevault write auth/kubernetes/role/app-staging \
  bound_service_account_names=* \
  bound_service_account_namespaces=staging \
  policies=staging-secrets \
  ttl=1h
```

Secret Rotation

```yaml
# External Secrets with frequent refresh
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: rotating-secrets
spec:
  refreshInterval: 15m  # Refresh every 15 minutes
```

```yaml
  secretStoreRef:
    name: securevault
    kind: ClusterSecretStore
  target:
    name: app-secrets
    template:
      metadata:
        annotations:
          reloader.stakater.com/auto: "true"  # Auto-reload pods
```

Monitoring

```yaml
# Monitor secret access
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: securevault-alerts
spec:
  groups:
    - name: securevault
      rules:
        - alert: SecretAccessFailure
          expr: increase(vault_secret_access_errors_total[5m]) > 0
          for: 1m
          labels:
            severity: warning
          annotations:
            summary: "Secret access failures detected"
```

Troubleshooting

Common Issues

Authentication Failed:

```bash
# Check service account token
kubectl exec -it $POD -- cat /var/run/secrets/kubernetes.io/serviceaccount/token | jwt decoc

# Verify Kubernetes auth config
securevault read auth/kubernetes/config

# Test authentication
```

```
securevault write auth/kubernetes/login \
  role=my-app \
  jwt=$(kubectl exec -it $POD -- cat /var/run/secrets/kubernetes.io/serviceaccount/token)
```

Secrets Not Injected:

```
# Check injector logs
kubectl logs -n securevault -l app=vault-sidecar-injector

# Verify pod annotations
kubectl get pod $POD -o yaml | grep securevault

# Check init container logs
kubectl logs $POD -c vault-agent-init
```

Permission Denied:

```
# Test policy
securevault token capabilities secret/data/my-app/config

# Check role configuration
securevault read auth/kubernetes/role/my-app
```

Debug Mode

```
annotations:
  securevault.novatech.com/inject: "true"
  securevault.novatech.com/log-level: "debug"
  securevault.novatech.com/preserve-secret-case: "true"
```

Helm Chart Configuration

Values Example

```
# values.yaml
global:
  vault:
    address: https://securevault.novatech.com
    authPath: auth/kubernetes

serviceAccount:
```

```
  create: true
  name: my-app

secrets:
  - path: secret/data/my-app/config
    template: |
      {{- with secret "secret/data/my-app/config" -}}
      export DATABASE_URL="{{ .Data.data.database_url }}"
      export API_KEY="{{ .Data.data.api_key }}"
      {{- end }}

csi:
  enabled: true
  secretProviderClass:
    name: my-app-secrets
    secrets:
      - secretKey: db-password
        remotePath: secret/data/my-app/db
        remoteKey: password
```

Migration Guide

From Kubernetes Secrets

1. **Inventory existing secrets**

```
kubectl get secrets -A -o jsonpath='{range .items[*]}{.metadata.namespace}/{.metadata.name}+
```

2. **Import to SecureVault**

```
#!/bin/bash
for secret in $(kubectl get secrets -o name); do
  name=$(echo $secret | cut -d'/' -f2)
  data=$(kubectl get $secret -o jsonpath='{.data}')
  securevault kv put secret/k8s-migrated/$name "$data"
done
```

3. **Update deployments**

- Add annotations or CSI configuration
- Test in staging
- Roll out gradually

API Reference

Kubernetes Auth Endpoints

Endpoint	Method	Description
`/auth/kubernetes/config`	POST	Configure auth
`/auth/kubernetes/role/:name`	POST	Create role
`/auth/kubernetes/login`	POST	Authenticate

Related Documents: Getting Started (PRD-SV-001), Dynamic Secrets (PRD-SV-020), Authentication (PRD-SV-010)