Zelf hosten met Helm

Zelf hosten met Helm

Dit artikel leidt je door de procedure om Bitwarden te installeren en te implementeren in verschillende Kubernetes-implementaties met behulp van een Helm-kaart.

Dit artikel beschrijft de algemene stappen voor het hosten van Bitwarden op Kubernetes. Er zijn aanbiedersspecifieke gidsen beschikbaar om te bekijken hoe je een implementatie kunt aanpassen op basis van het specifieke aanbod van elke aanbieder:

- Azure AKS implementatie
- OpenShift implementatie
- AWS EKS implementatie

Vereisten

Controleer voordat u verdergaat met de installatie of aan de volgende vereisten is voldaan:

- kubectl is geïnstalleerd.
- Helm 3 is geïnstalleerd.
- Je hebt een SSL-certificaat en -sleutel of toegang om er een te maken via een certificaatprovider.
- Je hebt een SMTP-server of toegang tot een cloud SMTP-provider.
- Een opslagklasse die ReadWriteMany ondersteunt.
- U hebt een installatie-id en -sleutel die u hebt opgehaald van https://bitwarden.com/host.

De kaart voorbereiden

De repo toevoegen aan Helm

Voeg de repo toe aan Helm met de volgende commando's:

```bash
helm repo add bitwarden https://charts.bitwarden.com/
helm repo update
```

Een naamruimte maken

Maak een naamruimte om Bitwarden in te implementeren. Onze documentatie gaat uit van een namespace met de naam `bitwarden`, dus pas de commando's aan als je een andere naam kiest.

```bash
kubectl create namespace bitwarden
```

Een configuratie maken

Maak een `my-values.yaml` configuratiebestand, dat je zult gebruiken om je implementatie aan te passen, met het volgende commando:

```Bash
helm show values bitwarden/self-host > my-values.yaml
```

Je moet op zijn minst de volgende waarden configureren in je `my-values.yaml` bestand:

Waarde	Beschrijving
`algemeen.domein:`	Het domein dat zal verwijzen naar het openbare IP-adres van je cluster.
`algemeen.ingress.ingeschakeld:`	Of de nginx ingress controller gedefinieerd in de grafiek gebruikt moet worden(zie een voorbeeld met een niet-ingebedde ingress controller).
`general.ingress.className:`	Bijvoorbeeld `"nginx"` of `"azure-application-gateway"`(zie een voorbeeld). Stel `general.ingress.enabled: false` in om andere ingresscontrollers te gebruiken.
`algemeen.ingress.annotaties:`	Annotaties om toe te voegen aan de ingangscontroller. Als je de meegeleverde nginx controller gebruikt, zijn er standaardinstellingen die je moet uncommenten en naar behoefte kunt aanpassen.
`algemeen.ingress.paden:`	Als je de standaard nginx controller gebruikt, zijn er standaardinstellingen die je naar behoefte kunt aanpassen.
`algemeen.ingress.cert.tls.naam:`	De naam van uw TLS-certificaat. We zullen later een voorbeeld bespreken, dus voer het nu in als je het hebt of kom later terug.
`algemeen.ingress.cert.tls.clusterIssuer:`	De naam van de uitgever van uw TLS-certificaat. We zullen later een voorbeeld bespreken, dus voer het nu in als je het hebt of kom later terug.
`algemeen.email.replyToEmail:`	E-mailadres dat wordt gebruikt voor uitnodigingen, meestal `no_reply@smtp_host`.

Waarde	Beschrijving
`algemeen.email.smtpHost:`	De hostnaam of het IP-adres van uw SMTP-server.
`general.email.smtpPort:`	De SMTP-poort die wordt gebruikt door de SMTP-server.
`algemeen.email.smtpSsl:`	Of uw SMTP-server een coderingsprotocol gebruikt(`true` = SSL, `false` = TLS).
`CloudCommunication inschakelen:`	Stel in op `waar` om communicatie tussen je server en ons cloudsysteem toe te staan. Dit maakt facturering en licentiesynchronisatie mogelijk.
`cloudRegio:`	Standaard VS. Stel in op EU als je organisatie is gestart via de EU cloudserver.
`gedeeldeStorageClassName:`	De naam van de gedeelde opslagklasse, die je moet opgeven en ReadWriteMany moet ondersteunen(zie een voorbeeld met Azure File Storage) tenzij het een single-node cluster is.
`secrets.secretName:`	De naam van je Kubernetes geheim object. Je zult dit object in de volgende stap maken, dus beslis nu over een naam of kom terug op deze waarde.
`database.ingeschakeld:`	Of de SQL-pod in de grafiek moet worden gebruikt. Alleen op `false` zetten als je een externe SQL-server gebruikt.
`component.scim.ingeschakeld`	De SCIM pod is standaard uitgeschakeld. Stel waarde = `waar` in om de SCIM pod in te schakelen.
`component.volume.logs.enabled:`	Hoewel dit niet vereist is, raden we aan dit op `true` te zetten om problemen op te lossen.

Een geheim object maken

Maak een Kubernetes secret object om minimaal de volgende waarden in te stellen:

Waarde	Beschrijving
`globalSettings__installation__id`	Een geldig installatie-id opgehaald van https://bitwarden.com/host. Zie voor meer informatie Waarvoor worden mijn installatie-id en installatiesleutel gebruikt?
`globalSettings__installation__key`	Een geldige installatiesleutel die is opgehaald van https://bitwarden.com/host. Zie voor meer informatie Waarvoor worden mijn installatie-id en installatiesleutel gebruikt?
`globalSettings__mail__smtp__gebruikersnaam`	Een geldige gebruikersnaam voor uw SMTP-server.
`globaalInstellingen__mail__smtp__wachtwoord`	Een geldig wachtwoord voor de ingevoerde gebruikersnaam van de SMTP-server.
`globaleInstellingen__yubico__clientId`	Client-ID voor YubiCloud Validation Service of zelf gehoste Yubico Validation Server. Als u YubiCloud gebruikt, kunt u hier uw klant-ID en geheime sleutel ophalen.
`globale instellingen__yubico__sleutel`	Geheime sleutel voor YubiCloud Validation Service of zelf gehoste Yubico Validation Server. Als u YubiCloud gebruikt, kunt u hier uw klant-ID en geheime sleutel ophalen.
`globaleInstellingen__hibpApiKey`	Je HaveIBeenPwned (HIBP) API-sleutel, hier beschikbaar. Met deze sleutel kunnen gebruikers het rapport over gegevensinbreuken uitvoeren en hun hoofdwachtwoord controleren op aanwezigheid in inbreuken wanneer ze een account aanmaken.
Als u de Bitwarden SQL-pod gebruikt, is `SA_PASSWORD` Als je je eigen SQL-server gebruikt, moet `globalSettings__sqlServer__connectionString`	Credentials voor de database die is verbonden met uw Bitwarden-instantie. Wat nodig is, hangt af van of u de meegeleverde SQL-pod of een externe SQL-server gebruikt.

Als je bijvoorbeeld het commando `kubectl create secret` gebruikt om deze waarden in te stellen, zou het er als volgt uitzien:

> ⚠ **Warning**
> Dit voorbeeld zal commando's opnemen in je shell geschiedenis. Er kunnen andere methoden worden overwogen om een geheim veilig in te stellen.

```Bash
kubectl create secret generic custom-secret -n bitwarden \
    --from-literal=globalSettings__installation__id="REPLACE" \
    --from-literal=globalSettings__installation__key="REPLACE" \
    --from-literal=globalSettings__mail__smtp__username="REPLACE" \
    --from-literal=globalSettings__mail__smtp__password="REPLACE" \
    --from-literal=globalSettings__yubico__clientId="REPLACE" \
    --from-literal=globalSettings__yubico__key="REPLACE" \
    --from-literal=globalSettings__hibpApiKey="REPLACE" \
    --from-literal=SA_PASSWORD="REPLACE"
```

Vergeet niet om de `secrets.secretName:` waarde in `my-values.yaml` in te stellen op de naam van het aangemaakte geheim, in dit geval `custom-secret`.

Voorbeeld certificaat instellen

Implementatie vereist een TLS-certificaat en -sleutel, of toegang tot het aanmaken ervan via een certificaatprovider. Het volgende voorbeeld leidt je door het gebruik van cert-manager om een certificaat te genereren met Let's Encrypt:

1. Installeer cert-manager op het cluster met het volgende commando:

   ```Bash
   kubectl apply -f https://github.com/cert-manager/cert-manager/releases/download/v1.11.0/cert-manager.yaml
   ```

2. Een uitgever van certificaten definiëren. Bitwarden raadt aan om de **Staging-configuratie** in dit voorbeeld te gebruiken totdat uw DNS-records naar uw cluster zijn verwezen. Zorg ervoor dat u de placeholder `e-mail:` vervangt door een geldige waarde:

⇒Staging

```Bash
cat <<EOF | kubectl apply -n bitwarden -f -
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-staging
spec:
  acme:
    server: https://acme-staging-v02.api.letsencrypt.org/directory
    email: me@example.com
    privateKeySecretRef:
      name: tls-secret
    solvers:
      - http01:
          ingress:
            class: nginx #use "azure/application-gateway" for Application Gateway ingress
EOF
```

⇒Productie

```Bash
cat <<EOF | kubectl apply -n bitwarden -f -
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-production
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: me@example.com
    privateKeySecretRef:
      name: tls-secret
    solvers:
      - http01:
          ingress:
            class: nginx #use "azure/application-gateway" for Application Gateway ingress
EOF
```

3. Als je dit nog niet hebt gedaan, zorg er dan voor dat je de waarden `general.ingress.cert.tls.name:` en `general.ingress.cert.tls.clusterIssuer:` instelt in `my-values.yaml`. In dit voorbeeld stel je in:
   - `general.ingress.cert.tls.name: tls-secret`
   - `general.ingress.cert.tls.clusterIssuer: letsencrypt-staging`

RawManifest bestanden toevoegen

Met de Bitwarden self-host Helm Chart kun je andere Kubernetes manifest-bestanden voor of na de installatie toevoegen. Om dit te doen, moet je de `rawManifests` sectie van de kaart bijwerken(meer informatie). Dit is bijvoorbeeld handig in scenario's waar je een andere ingress controller wilt gebruiken dan de nginx controller die standaard is gedefinieerd.

Installeer de kaart

Voer het volgende commando uit om Bitwarden te installeren met de configuratie in `my-values.yaml`:

```Bash
helm upgrade bitwarden bitwarden/self-host --install --namespace bitwarden --values my-values.yaml
```

Gefeliciteerd! Bitwarden is nu actief op `https://your.domain.com`, zoals gedefinieerd in `my-values.yaml`. Bezoek de webkluis in je webbrowser om te controleren of deze werkt. Je kunt nu een nieuwe account registreren en inloggen.

Je moet een SMTP-configuratie en bijbehorende geheimen hebben ingesteld om de e-mail voor je nieuwe account te kunnen verifiëren.

Volgende stappen

Back-up en herstel van database

In dit archief hebben we twee illustratieve voorbeeldtaken gegeven voor het back-uppen en herstellen van de database in de Bitwarden databasepod. Als u uw eigen SQL Server-instantie gebruikt die niet wordt ingezet als onderdeel van deze Helm-kaart, volg dan het back-up- en herstelbeleid van uw bedrijf.

Database back-ups en back-up beleid zijn uiteindelijk aan de uitvoerder. De back-up kan buiten het cluster worden ingepland om op regelmatige intervallen te worden uitgevoerd, of het kan worden aangepast om een CronJob-object binnen Kubernetes te maken voor planningsdoeleinden.

De back-uptaak maakt tijdstempelversies van de vorige back-ups. De huidige back-up heet gewoon `vault.bak`. Deze bestanden worden in het persistente volume van MS SQL-back-ups geplaatst. De hersteltaak zoekt naar `vault.bak` in hetzelfde permanente volume.