Traffic Routing

Afficher dans le centre d'aide https://bitwarden.com/help/traffic-routing/

Traffic Routing

> **Note**
>
> Ingress NGINX has reached EOL and will no longer receive support. Bitwarden has included configurations for Gateway API in `my-values.yaml`. Please see Kubernetes' official statement on the deprecation of Ingress NGINX.

This article provides sample traffic routing configurations for Kubernetes based Bitwarden deployments, and should be used alongside Self-host with Helm. This article covers a standard Gateway API setup, as well as migration steps if you are running the deprecated Ingress NGINX configuration.

Prerequisites

Before proceeding with the Gateway API setup, ensure the following requirements and initial setup have been completed in Self-host with Helm:

1. Bitwarden namespace created
2. Create Secrets
3. Create and install certificates

NGINX Gateway Fabric

The following sections include instructions to setup or migrate from Ingress NGINX to NGINX Gateway Fabric. Configure Gateway API for your Bitwarden Helm deployment using the steps below:

- **New Deployment**: Follow the steps in order.
- **Migrating from Ingress NGINX:** Complete the steps up to Create the Gateway resource, then skip to Migrating from Ingress NGINX to NGINX Gateway Fabric.

Install the Gateway API custom resource definition

Install the Gateway API custom resource definitions before deploying a Gateway controller. The following example is using NGINX Gateway Fabric v2.4.2:

```Bash
kubectl kustomize "https://github.com/nginx/nginx-gateway-fabric/config/crd/gateway-api/standard?ref=v2.4.2" | kubectl apply -f -
```

Note

CRD versions are tied to the Gateway controller version. Check your controller's documentation to confirm the compatible CRD version before running this command. In this example, the CRD version is tied to NGINX Gateway Fabric.

Additional implementation options can be found in Gateway API's documentation.

Install a Gateway controller

A Gateway controller handles the traffic routing. To install NGINX Gateway Fabric using Helm:

```Bash
helm install ngf oci://ghcr.io/nginx/charts/nginx-gateway-fabric \
  --create-namespace \
  -n nginx-gateway \
  --set nginx.service.type=LoadBalancer
```

Create the Gateway resource

Create a `Gateway` resource in the `bitwarden` namespace. The Gateway terminates TLS using the `self-signed-cert` secret created during the prerequisite setup, and restricts attached routes to the same namespace.

YAML

```yaml
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: bw-gateway
  namespace: bitwarden
spec:
  # Change this if your controller uses a different GatewayClass.
  # For example, many environments will NOT use "nginx" here.
  gatewayClassName: nginx
  listeners:
    - name: http
      hostname: bw.localtest.me       # change this value
      port: 80
      protocol: HTTP
      allowedRoutes:
        namespaces:
          from: Same
    - name: https
      hostname: bw.localtest.me    # change this value
      port: 443
      protocol: HTTPS
      tls:
        mode: Terminate
        certificateRefs:
          - kind: Secret
            name: self-signed-cert
      allowedRoutes:
        namespaces:
          from: Same
```

If your setup requires HTTP to HTTPS redirect, you can use the following additional route to a `redirect.yaml` file:

YAML

```yaml
apiVersion: gateway.networking.k8s.io/v1
kind: HTTPRoute
metadata:
  name: bw-gateway-http-redirect
  namespace: bitwarden
spec:
  hostnames:
    - bw.localtest.me
  parentRefs:
    - group: gateway.networking.k8s.io
      kind: Gateway
      name: bw-gateway
      sectionName: http
  rules:
    - filters:
        - type: RequestRedirect
          requestRedirect:
            scheme: https
            statusCode: 301
```

To apply HTTP to HTTPS:

Bash

```
kubectl apply -f <redirectFileName>
```

Apply the manifest:

Bash

```
kubectl apply -f gateway.yaml
```

To list the `GatewayClass` resources installed by your controller, run:

```Bash
kubectl get gatewayclass
```

The `gatewayClassName` value above must match one of these.

> **Note**
>
> At this point, if you are migrating from an Ingress NGINX setup, continue to Migrating from Ingress NGINX to Gateway API.

Configure the Helm chart for Gateway API

Update the `my-values.yaml` file to disable the deprecated Ingress, and enable the Gateway API HTTPRoute. Set `parentRefs` to point to the Gateway:

```YAML
general:
  ingress:
    # Ingress is deprecated. Disable it when using Gateway API.
    enabled: false
  gateway:
    # Set to true to create an HTTPRoute resource managed by the Helm chart.
    enabled: true
    # parentRefs attach the HTTPRoute to the Gateway.
    parentRefs:
      - name: bw-gateway        # Must match the Gateway metadata.name
        namespace: bitwarden    # Must match the Gateway metadata.namespace
        sectionName: https      # Must match the listener name in the Gateway spec
```

Apply the changes:

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

Note

The provided `my-values.yaml` file includes configurations for both Ingress and Gateway API setups. Both of these methods can be utilized at the same time depending on your specific infrastructure needs.

HTTPRoute functionality

When `general.gateway.enabled` is `true`, the Helm chart creates an `HTTPRoute` resource in the `bitwarden` namespace. The `HTTPRoute` attaches to the Gateway defined in `parentRefs` and routes traffic to each Bitwarden service by path prefix. The HTTPRoute chart produced by the configuration can be reviewed here.

Note

TLS is handled at the Gateway level, not the HTTPRoute. Do not add TLS configuration to the HTTPRoute resource.

Migrating from Ingress NGINX to Gateway API

If you have an existing Bitwarden Helm deployment using the deprecated `general.ingress` configuration, you may migrate to Gateway API. If you have not completed Install NGINX Gateway Fabric, and Create Gateway resource from the steps above, please do so before returning to this section.

1. Next, update your values file to disable Ingress and enable the Gateway:

YAML

```yaml
general:
  ingress:
    enabled: false
  gateway:
    enabled: true
    parentRefs:
      - name: bw-gateway
        namespace: bitwarden
        sectionName: https
```

2. Apply the changes:

Bash

```bash
helm upgrade bitwarden bitwarden/self-host \
  --namespace bitwarden \
  --values my-values.yaml
```

The chart will delete the old `Ingress` resource and create the `HTTPRoute` in its place.