Enabling HTTPS on a Cloud Run for Anthos on Google Cloud cluster

This page describes how to configure Cloud Run for Anthos on Google Cloud to use your SSL/TLS certificate for your domain to enable HTTPS connections.

Important: due to current limitations in Istio, Cloud Run for Anthos on Google Cloud supports only a single certificate per cluster. If you serve multiple domains in the same cluster, make sure the certificate is signed for all the domains.

Before you begin

These instructions assume that you already completed mapping your Cloud Run for Anthos on Google Cloud service (https://cloud.google.com/run/docs/mapping-custom-domains) to use your custom domain.

Obtaining an SSL/TLS certificate using `Let's Encrypt`

If you already have the SSL/TLS certificates you need, skip these instructions.

If you don't have an existing SSL/TLS certificate, you can use the `Let's Encrypt` certbot to obtain a certificate manually:

1. Install the certbot-auto script from the Certbot (https://certbot.eff.org/docs/install.html#id6) website on your local machine or in a Cloud Shell (https://cloud.google.com/shell/) machine in your project :

```
wget https://dl.eff.org/certbot-auto
chmod a+x ./certbot-auto
./certbot-auto --help
```

2. Use the certbot to request a certificate, using DNS validation. The certbot tool walks you through validating your domain ownership by creating TXT records in your domain:

```
./certbot-auto certonly --manual --preferred-challenges dns -d '*.default.yourd
```

3. After certbot completes, you have two output files, `privkey.pem` and `fullchain.pem`. These files are used when you import a TLS certificate/private key into a Kubernetes Secret (#secret).

Warning: Certificates issued by `Let's Encrypt` are only valid for 90 days. You must renew your certificate with the certbot tool again every 90 days.

Importing TLS certificate/private key into a Kubernetes Secret

To import the certificates into a Secret:

1. Copy the certificates into your current directory.
2. Use the following command to create a Secret that stores the certificates, where `privkey.pem` contains your certificate private key and `fullchain.pem` contains the public certificate"

```
kubectl create --namespace NAMESPACE secret tls ISTIO-GATEWAY-certs \
--key privkey.pem \
--cert fullchain.pem
```

Replace ***ISTIO-GATEWAY*** and ***NAMESPACE*** as follows:

Cluster version	ISTIO-GATEWAY	NAMESPACE
`1.15.3-gke.19` and greater `1.14.3-gke.12` and greater `1.13.10-gke.8` and greater	`istio-ingress`	`gke-system`
All other versions	`istio-ingressgateway`	`istio-system`

You must use `ISTIO-GATEWAY-certs` as the Secret name, as shown in the example.

Configuring the gateway

You must configure the ingress gateway spec to use the new secret that contains the certificate.

To configure the gateway:

1. Open the shared gateway spec for editing:

   ```
   kubectl edit gateway GATEWAY --namespace knative-serving
   ```

   Replace ***GATEWAY*** as follows:

Cluster version	GATEWAY
1.15.3-gke.19 and greater **1.14.3-gke.12** and greater **1.13.10-gke.8** and greater.	**gke-system-gateway**
All other versions	**knative-ingress-gateway**

2. Change the gateway spec to include the `tls:` section as shown below:

   ```
   # Edit the object below. Lines beginning with a # will be ignored.
   # and an empty file will abort the edit. If an error occurs while saving this f
   # reopened with the relevant failures.
   apiVersion: networking.istio.io/v1alpha3
   kind: Gateway
   metadata:
   # ... skipped ...
   spec:
     selector:
       istio: ingressgateway
     servers:
       - hosts:
         - "*"
         port:
           name: http
           number: 80
           protocol: HTTP
       - hosts:
         - "*"
         port:
           name: https
   ```

```
        number: 443
        protocol: HTTPS
      tls:
        mode: SIMPLE
        privateKey: /etc/istio/ingressgateway-certs/tls.key
        serverCertificate: /etc/istio/ingressgateway-certs/tls.crt
```

3. Save your changes.

After this change, you can use the HTTPS protocol to access your deployed services.