| Master in Cybersecurity  Course 2022-2023 |
| --- |
| Thesis App Setup |
| **Thesis** |



***Clem O'Regan, 100480776***

# 

# TLS Full Setup (via Terraform)

First try the managed cert setup:

1. Follow: <https://cloud.google.com/kubernetes-engine/docs/how-to/managed-certs#console>
   1. gcloud compute addresses create tftst --project=tls-terraform --global (did this via the GCP console)
   2. kubectl apply -f managed-cert.yaml (DNS entries to cert will be stored here)
   3. kubectl apply -f mc-service.yaml
   4. kubectl apply -f managed-cert-ingress.yaml
2. Check status: “kubectl get ingress” - If ADDRESS field is not filled, give it time.. May have to wait a bit for ingress to register (check events to make sure all ok)
3. Then create the DNS domains specified in cert above in google domain, and put in IP of create load balancer!
4. Have to then wait again (up to 60m), in meantime check:
   1. “watch kubectl describe managedcertificate managed-cert”
   2. Also watch events
   3. After that time cert should go to ACTIVE
5. You can then verify cert is secure by going to <https://domain/> in browser

Deleting (if cluster created by TF):

* kubectl delete -f managed-cert.yaml
* gcloud compute addresses delete tftst --project=tls-terraform –global
* terraform destroy
* After delete certificates (certificate manager)
* Delete LB that was created as well!

## Integrating with traefik

* Ensure create static IP and create cert 1a & traefikcert.yaml created (will be created in traefik namespace)
* Install traefik via helm (using the values file above first?) into traefik namesapce
* Apply traefik-ingress.yaml
  + (on port 443?? but kubectl ingress on port 80?)
  + So certificate not auto recognized in LB
  + Change LB config - either via console or some config file way
* Apply traefik-ingressroute.yaml
* May need to update LB background as well - not healthy! But can do that after I think!

## Create additional load balancer for Full E2E TLS

After considering many options, for more consistent test results I opted to go for an additional secure route through the same cluster. Isolating tests (not running in parallel), on the same base resources.

Therefore necessary to create additionally:

1. Load balancer for full E2E TLS
2. Managed certificate for full E2E TLS
3. Route to traefik ingress controller on websecure port (TLS)
4. Separate namespace for each TLS application depending on cipher suite

| **NOTE**: Very important here - LB is exposed on 443 port, but traefik is exposed on different port (8443 websecure) - so when configuring backend, need to ensure its the 8443 port that is used, or specify websecure in traefik annotations! |
| --- |

To perform steps 1-4 above, need to run scripts in ../tfm/tlson/kub\_lb\_additional/ folder 0-3.

Have to wait after step 2 checking:

* k get ingress
* k get events
* k describe ingress
* watch kubectl describe managedcertificate e2e-cert -n tlsoff
* Make sure google domain updated with LB IP once ingress is synched
* Then once managed cert is active (30 mins or so later), run 3.yaml to create traefik ingress route
* Aside to try & get certificate to store later in a secret use:

| gcloud beta compute ssl-certificates describe mcrt-f2d40619-0cc2-4d5b-a9b5-335e78e67875 --project=tfm-jun18 --format='get(privateKey)' > privatekey.pem  &  gcloud beta compute ssl-certificates describe mcrt-f2d40619-0cc2-4d5b-a9b5-335e78e67875 --project=tfm-jun18 --format='get(certificate)' > certificate.pem |
| --- |

* But private key for certificate is stored in google and you cannot get it! So you can’t use kubernetes secrets for the HTTPS classic load balancer. Also appears that google cloud does not support fully TLS passthrough with HTTPS load balancer. Google keeps synching the load balancer and reverting back to HTTP. **So need to use a TCP network load balancer to do this!**

Hit a problem with quotas (IAM - quotas) on IN USE IP ADDRESSES. Can view all IP addresses in use via searching for IP addresses (VPC)

* Should have created cluster with private IPs on nodes! As can’t delete them
* Could still sync LB by manual update via GCP web console
* Managed to release an old static IP, then LB synched correctly in kubectl events
* Create healthcheck ingress route on websecure
* Ensure backend “ENDPOINT Protocol” is configured to HTTPS, if not change this

Then ensure LB backend is synched:

* I created a new backend config with the GCP console
* And changed the backend config - HTTPS & also changed the healthcheck to HTTPS, changing the port to 8443 (as configured in traefik deployment - websecure)
* Healthcheck status then changed to green 🙂.. With healthcheck green - could then access the traefike2e dashboard via full TLS!

## Try with creating a second cluster

For various problems, and cause ingress we are creating is a second on on top of existing traefik.. try creating a new cluster and running fresh install of traefik there in HTTPS.

Steps:

1. Create cluster via Terraform - 1 node.
2. In folder /Users/clemoregan/Desktop/tfm/tlson/kub\_lb\_additional/second\_cluster run:
   1. Connect to new cluster
   2. k create ns traefik
   3. k apply -f 0-static (If new GCP project)
   4. k apply -f 1-e2etlscert.yaml
   5. **Deploy traefik**: helm upgrade --install traefik traefik/traefik --values traefik\_values.yaml -n traefik
   6. Now create ingress for traefik! Big moment!! **k apply -f 2-traefike2e-ingress.yaml**
      1. Do the usual monitoring for 30 mins until cert & ing are synched
   7. In meantime Create ingress route to healthcheck: k apply -f 3-...
   8. And update the healthcheck for newly created LB:

| gcloud compute health-checks update http --request-path "/healthcheck" --project=tls-terraform k8s-be-32080--594371466df8a851 |
| --- |

* 1. So LB healthcheck should become green
  2. FE config may still not work, as need to perform HTTPS offloading (will need to update FE to configure static IP address to reconfigure to HTTPS offloading)

| Note: Forums seem to think that google managed certs are for front end only:  <https://stackoverflow.com/questions/75046959/using-http2-with-gke-and-google-managed-certificates>  HTTPS between LB and apps:  <https://cloud.google.com/kubernetes-engine/docs/concepts/ingress-xlb#setting_up_https_tls_between_client_and_load_balancer>  May not need backend-config TLS, as google apparently encrypts it on network layer anyway:  <https://stackoverflow.com/questions/70782206/gke-ingress-configuration-for-https-enabled-applications-leads-to-failed-to-conn> |
| --- |

Maybe best option is traefik - select cipher suite.. As google encrypt traffic on network layer underneath, and google managed certificates cannot be included in kubernetes secrets.. could be little benefit in performance to measure full TLS!

Otherwise will probably need to use lets encrypt, and self managed certs

Justify..

## Setup metrics & monitoring

Follow practical here:

<https://www.youtube.com/watch?v=CL5Cxxz-yHo&t=4320sz>

<https://github.com/traefik-workshops/traefik-workshop>

Exercise 15:

But with following changes

| helm repo add prometheus-community <https://prometheus-community.github.io/helm-charts>  helm repo update |
| --- |

Don’t run the kubectl apply -f traefik-dashboard.yaml

* As they don’t seem to set the metrics correctly! for k get endpoints

Instead deploy the new helm values, with traefik metrics enabled

| metrics:  prometheus:  serviceMonitor:  service:  enabled: true |
| --- |

Deploy the prometheus stack from helm (will need at least 2 worker nodes for resources!)

When testing for internal “k get svc” & “k get endpoints” run the alpine pod:

k run -it --rm --image alpine -- sh

& run the curl against the svc (specifying namespace where traefik-metrics service is deployed:

| curl traefik-metrics.tlsoff.svc.cluster.local:9100/metrics |
| --- |

This should return all metrics (which we know now have been sent to prometheus)..

So then do the port forward & access grafana & also port-forward prometheus - see metrics:

| kubectl port-forward service/prometheus-stack-grafana 8080:80  k port-forward -n default svc/prometheus-stack-kube-prom-prometheus 9090:9090 |
| --- |

Note: Can later set these up as ingresses in traefik!

Should then be able to access above ports for each application via http://locahost:port.

## Deploy app with TLS on websecure

So now we have:

* DNS created for our new targeted app e.g. rsa2 (for RSA 2048 encryption)
* traefik setup on full TLS through a new load balancer via web secure

So we need to

* Ensure the docker image is created for RSA2048 - is this via httpd.conf in image? And upload to container registry. How do we ensure this cipher?
* deploy the app into the cluster first via helm and using websecure (full TLS).

Helm config is found in ../tfm/tlson/helm/rsa2048app/ (change to docker image above after, will deploy standard image in websecure first)

* Need to load in specific docker
* Ensure configured for web secure
* Change liveness & readiness probes to https
* Change the service account to be deployed within this apps namespace

| helm install rsa2 helm/rsa2048app -n rsa2 --set dbpass=tfm2023 –create-namespace |
| --- |

## Appendix

Can create node pool with private addresses only via terraform (so don’t use all external IP addresses):

The google\_container\_cluster resource has a block private\_cluster\_config this field: <https://www.terraform.io/docs/providers/google/r/container_cluster.html#enable_private_nodes>

Does setting that remove the external IP on the node pool instances?