GRPC-GRPCS (NODE-GO-Nginx-Java)

Requisitos

Openssl
Nginx 1.15 >
Node
GO
JAVA
*Pm2 solo opcional si lo requieres
Acceso root a local y a servidor
Uso y manejo de archivos .env en sus micro servicios (opcional)

Generar certificados SSL para GRPCS

*******MUY IMPORTANTE********

Para el uso de SSL se requiere el uso de dominios o subdominios no se usan ip en caso de usar localhost se requiere modificar el archivo /etc/hosts de lo contrario mandara errores de SSL la coneccion

---

*Cambiar 1111 por la contraseña segura que deseen
*Cambiar /C=MX/ST=QRO/L=Queretaro/O=miempresa.com/OU=Tectonogia/CN=*miempresa.com : por su info
Para subdominios (*miempresa.com)
Para un solo dominio (miempresa.com) ojo no es lo mismo www.miempresa.com a http://miempresa.com

- CN - CommonName
- L - LocalityName
- ST - StateOrProvinceName
- O - OrganizationName
- OU - OrganizationalUnitName
- C - CountryName

#Code SH nombre.sh con permisos 777 y ejecutar como root
echo "Creating certs folder ..."
mkdir certs && cd certs
echo "Generating certificates ..."
openssl genrsa -passout pass:1111 -des3 -out ca.key 4096
openssl req -passin pass:1111 -new -x509 -days 365 -key ca.key -out ca.crt -subj
"/C=MX/ST=RM/L=Queretaro/O=miempresa.com/OU=Tectonogia/CN=*miempresa.com"
openssl genrsa -passout pass:1111 -des3 -out server.key 4096
openssl req -passin pass:1111 -new -key server.key -out server.csr -subj
"/C=MX/ST=RM/L=Queretaro/O=miempresa.com/OU=Server/CN=*miempresa.com "
openssl x509 -req -passin pass:1111 -days 365 -in server.csr -CA ca.crt -CAkey ca.key -set_serial 01 -out server.crt
openssl rsa -passin pass:1111 -in server.key -out server.key
openssl genrsa -passout pass:1111 -des3 -out client.key 4096
openssl req -passin pass:1111 -new -key client.key -out client.csr -subj
"/C=MX/ST=RM/L=Queretaro/O=miempresa.com/OU=Client/CN=*miempresa.com"
openssl x509 -passin pass:1111 -req -days 365 -in client.csr -CA ca.crt -CAkey ca.key -set_serial 01 -out client.crt
openssl rsa -passin pass:1111 -in client.key -out client.key
openssl pkcs8 -topk8 -nocrypt -passin pass:1111 -in server.key -out server.pem
openssl pkcs8 -topk8 -nocrypt -passin pass:1111 -in client.key -out client.pem
cat client.crt ca.crt > miempresac.bundle.crt
cat server.crt ca.crt > miempresas.bundle.crt
openssl x509 -inform DER -outform PEM -in server.csr -out server.pem
##Fin de archivo

La estructura final de los certificados seria

certs/

```
-rw-r--r--@ 1 staff   2.0K Mar  6 17:43 ca.crt
-rw-r--r--  1 staff   3.2K Mar  6 17:43 ca.key
-rw-r--r--  1 staff   1.9K Mar  6 17:43 client.crt
-rw-r--r--@ 1 staff   1.7K Mar  6 17:43 client.csr
-rw-r--r--@ 1 staff   1.7K Mar  6 17:43 client.pem
-rw-r--r--  1 staff   3.2K Mar  6 17:43 client.key
-rw-r--r--  1 staff   3.8K Mar  6 17:43 cmiempresac.bundle.crt
-rw-r--r--  1 staff   3.8K Mar  6 17:43 miempresas.bundle.crt
-rw-r--r--@ 1 staff   1.9K Mar  6 17:43 server.crt
-rw-r--r--@ 1 staff   1.7K Mar  6 17:43 server.csr
-rw-r--r--@ 1 staff   3.2K Mar  6 17:43 server.key
-rw-r--r--@ 1 alejandrorico  staff   3.2K Mar  6 17:43 server.pem
```

****ojo vencen cada año y se tienen que crear todos y remplazarlos antes de que caduquen y guardar en una ruta accesible pero segura de preferencia con acceso solo a root por ejemplo /etc/ssl/certificados/

NGINX

Tener instalado nginx sea el SO que sea se tiene que agregar la info siguiente.
Te recomindo ejecutar ngenix-debug para tus pruebas y ya después nginx normal

```
#http SOLO SI EL MICRO ES SERVIDOR
log_format  node  '$remote_addr - $remote_user [$time_local] "$request" '
                  '$status $body_bytes_sent "$http_referer" '
                 '"$http_user_agent"';
##

server {
        listen 50000 ssl http2;  #sin ssl: listen 50000 http2
        server_name grpcs.miempresa.com;
        ssl_certificate /rutadondeestanlosssl/miempresas.bundle.crt;
        ssl_certificate_key /rutadondeestanlosssl/server.key;
        add_header Strict-Transport-Security "max-age=31536000; includeSubdomains; preload";
        proxy_hide_header X-Powered-By;
        add_header X-Frame-Options SAMEORIGIN;
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";
        ssl_prefer_server_ciphers on;
        ssl_ciphers "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:ECDHE-
RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-SHA:DHE-RSA-
AES256-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA:ECDHE-RSA-DES-CBC3-
SHA:EDH-RSA-DES-CBC3-SHA:AES256-GCM-SHA384:AES128-GCM-SHA256:AES256-SHA256:AES128-
SHA256:AES256-SHA:AES128-SHA:DES-CBC3-SHA:HIGH:!aNULL:!eNULL:!EXPORT:!DES:!MD5:!PSK:!RC4:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-
AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-
RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-
RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-
AES256-SHA:DHE-RSA-AES256-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-
SHA256:AES128-SHA:AES256-SHA:AES:CAMELLIA:DES-CBC3-
SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!aECDH:!EDH-DSS-DES-CBC3-SHA:!EDH-RSA-DES-CBC3-
SHA:!KRB5-DES-CBC3-SHA";
        expires $expires;
        access_log /var/log/nginx/50000.log node; #aqui puedes ver todo lo que entra por ese puerto
location ~ /\.ht {
            deny  all;
}
 location / {
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header Host $http_host;
                proxy_set_header X-NginX-Proxy true;
                proxy_set_header X-Ssl on;
                # Verificar el puerto de su microservicio en este caso en el 50001 si no se usa ssl usar grpc no grpcs
                #Node
                grpc_pass grpcs://127.0.0.1:50001;
                #Go
                #grpc_pass grpcs://127.0.0.1:50001;
                proxy_redirect off;

        }
}
```

.ENV

Para comodidad y mejor manejo de configuración se requiere .env y manejar por bloques las siguientes variables según los micros que se usen, pero recuerda de tener todos los certificados en la misma ubicación.

Servidor NODE

```
# gRPC Port
GRPC_PORT=50001
# gRPC
GRPC_HOST=t1.miempresa.com #depende de tu configuracion puede ser localhost
# gRPC SSL
GRPC_INSECURE=false
GRPC_CA_CRT=/usr/local/certs/ca.crt
GRPC_SERVER_CRT=/usr/local/certs/server.crt
GRPC_SERVER_KEY=/usr/local/certs/server.key
```

Cliente NODE

```
# gRPC service
GRPC_ INSERCURE=false
GRPC _PORT=50001
GRPC _HOST=t1.miempresa.com
GRPC _CA_CRT=/usr/local/certs/miempresas.bundle.crt
GRPC _CRT=/usr/local/certs/client.crt
GRPC_KEY=/usr/local/certs/client.key
```

Servidor GO

```
# gRPC SSL
GRPC_SERVER_INSECURE=false
GRPC_SERVER_DNS=localhost
GRPC_SERVER_PORT=50001
GRPC_SERVER_CRT=/usr/local/certs/server.crt
GRPC_SERVER_PEM=/usr/local/certs/server.pem
```

Cliente GO

```
# gRPC SSL
GRPC_SERVER_INSECURE=false
GRPC_SERVER_DNS=localhost
GRPC_SERVER_PORT=50001
GRPC_CA_CRT=/usr/local/certs/miempresas.bundle.crt
```

Java CLIENTE

```
HOST=t1.miempresa.com
PORT=50088
CRT=/usr/local/certs/client.crt
PEM=/usr/local/certs/client.pem
CA_CRT=/usr/local/certs/cubobits.bundle.crt
```

Java SERVIDOR

```
HOST=localhost
PORT=50088
CRT=/usr/local/certs/server.crt
PEM=/usr/local/certs/server.pem
```

Archivo host de cliente

10.0.0.10 t1.miempresa.com

Archivo host de servidor

127.0.0.1 localhost t1.miempresa.com
localhost tq.miempresa.com

Con esto ya tienes configurado el uso de ssl en tu servidor con GRPCs y Tus micros ya tienen GRPCs para una conexión segura entre si y entre el medio de comunicación.

Recuerda que GRPCs y GRPC son del tipo http2 y pueden o no ser ssl y se pueden usar en balanceo igual que cualquier otro tipo de conexión

Para debug de conexión te recomiendo activar ngnix-debug y verificar como pasa tu info ya que el servidor tiene que responder con un 200 y tienes 400 o 500 estan mal enrutados verifica puertos y respuesta del micro y se da un error de SSL muy seguramente son los Nombres de dominio y los certificados en Nginx y/o los extremos
Ejemplo:

BIEN
IP - - [19/Mar/2019:12:50:41 -0600] "POST /micro/ssss HTTP/2.0" 200 7 "-" "grpc-node/1.19.0 grpc-c/7.0.0 (osx; chttp2; gold)" 50052 https [o http]

MAL
IP - - [19/Mar/2019:12:45:57 -0600] "PRI * HTTP/2.0" 400 150 "-" "-" 50052 http

Con PM2 puedes realizar un pm2 logs [MICRO] --lines 100 para ver que está pasando igual error de SSL son certificados no le busques más chécalos valídalos y verifica que estén en su lugar y verifica permisos del archivo y propietario del archivo

Referencias:
https://github.com/eamexicano/certs/blob/master/certs.sh
https://blog.eamexicano.com/ssl/certificado-openssl/
https://gist.github.com/Soarez/9688998
https://www.nginx.com/blog/nginx-1-13-10-grpc/
https://coderwall.com/p/dgwwuq/installing-nginx-in-mac-os-x-maverick-with-homebrew

PD:si tienes dudas manda correo a arelis.mex@gmail.com
Donaciones que serán repartidas entre los que ayudaron a realizar este turorial:
BTC 38qjUSrs8E9qyPnxV29BiLicqPP8Mv9cBF
LTC MTMdioquuDUpkJWWNBH5XiVmJTjNcRoQjr
Ripple rGXo5PQtVSBZb7CciaoS1X6SfgsYXCPLzu TAG 3
DASH 7ejSTiLbr7YHD56Eio4N9bufTEWvxAtPgw
Stellar Lumens GCMKSHPPJNHBY3G7SBAEHYTMJP257TQ33JE3LNQUDCSZHXR4OE4KD2I2 MEMO 3