

gRPC Agent

- About gRPC Agent, on page 1
- Guidelines and Limitations for gRPC Agent , on page 1
- Configuring the gRPC Agent , on page 3
- Troubleshooting, on page 11

About gRPC Agent

gRPC is a modern opensource high performance Remote Procedure Call (RPC) framework. Cisco NX-OS provides a gRPC agent to support gRPC related services including: gNMI and gNOI.

Guidelines and Limitations for gRPC Agent

Following are the guidelines and limitations for gRPC agent:

- When you enable gRPC on both the management VRF and default VRF and later disable on the default VRF, the gNMI operations on the management VRF stop working.

  As a workaround, disable gRPC completely by entering the **no feature grpc** command and reprovision it by entering the **feature grpc** command or with any existing gRPC configuration commands like, **grpc certificate** or **grpc port**. You must also resubscribe to any existing notifications on the management VRF.

- If the gRPC certificate is explicitly configured, after a reload with the saved startup configuration to a prior Cisco NX-OS 9.3(x) image, the gRPC feature does not accept connections.

  To confirm this issue, enter the **show grpc gnmi service statistics** command. The following status error message is displayed:

  ```
  Status: Not running - Initializing...Port not available or certificate invalid.
  ```

  Unconfigure and configure the proper certificate command to restore the service.

- By default, the switch generates a one-day temporary certificate to allow you enough time to install your custom certificate. The temporary one-day certificate gets regenerated when a switch reload or system switchover event occurs.

- If you have configured a custom gRPC certificate, upon entering the **reload ascii** command the configuration is lost. It reverts to the default day-1 certificate. After entering the **reload ascii** command, the switch reloads. Once the switch is up again, you must reconfigure the gRPC custom certificate.

Note This applies when entering the grpc certificate command.

- The reachability in non-default VRF for gRPC is supported only over L3VNI's/EVPN and IP. However, reachability over MPLS in non-default VRF and VXLAN Flood and Learn is not supported.
- Starting with Cisco NX-OS release 9.3(x), for information about supported platforms, see the Nexus Switch Platform Matrix.
- The gRPC process uses the HIGH_PRIO control group, which limits the CPU usage to 75% of CPU and memory to 4 GB.
- The gRPC agent supports management VRF and one user specified VRF for a total of two gRPC servers on each switch. Supporting a gRPC in the user specified VRF (for example: the default VRF) adds flexibility to offload processing gRPC calls from the management VRF, where significant traffic load is not desirable.
- If two gRPC servers are configured, be aware of the following:
  - VRF boundaries are strictly enforced, so each gRPC server process requests independent of the other. Requests do not cross between VRFs.
  - The two servers are not HA or fault tolerant. One gRPC server does not back up the other, and there is no switchover or switchback between them.
  - Any limits for the gRPC server are per VRF.
- Beginning with Cisco NX-OS Release 10.4(3)F, gRPC is supported on 92348GC-X.
- Beginning with Cisco NX-OS Release 10.4(3)F, TLS v1.3 is supported, and the minimally supported version is v1.2.
- The gRPC agent supports a passwordless unix domain socket for power customers who plant their own agent in the switch.
  - Username and password are not required as authentication is bypassed.
  - For authorization and accounting
    - If username is missing in the gRPC Meta, agent would assign it to the default 'admin' user.
    - If username is specified, agent would use the given username to process the request.
- Beginning with Cisco NX-OS Release 10.5(1)F, NX-API will restart whenever you execute **fips mode enable** or **fips mode disable**.

Configuring the gRPC Agent

Configuring gRPC

Configure the gNMI feature through the grpc commands.



Note When modifying connection related config like **grpc port**, **grpc certificate**, or the **trustpoint content**, the gRPC server might restart to apply the changes. When the gRPC server restarts, any active subscription is dropped, and you must resubscribe.

Before you begin

Prepare the pkcs12 file which bundles the complete server key, service cert, CA and intermediate cert for the server authentication. This is not specific to gRPC, so it is possible to re-use the existing pkcs12 files deployed by other Nexus services.

SUMMARY STEPS

1. **configure terminal**
2. (Optional) **crypto ca trustpoint** <*server-trustpoint*>
3. **crypto ca import** <*server-trustpoint*> **pkcs12 bootflash:** *:*<*server-ca-file*> <*pkcs-password*>
4. **feature grpc**
5. (Optional) **grpc port** *port-id*
6. **grpc certificate** *certificate-id*
7. (Optional) **use-vrf default**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	(Optional) **crypto ca trustpoint** <*server-trustpoint*> **Example:** `switch# crypto ca trustpoint tls_server_trustpoint`	Creates a trustpoint for server authentication. Step 2-3 is optional if there already exist imported pkcs12 server trustpoint.
Step 3	**crypto ca import** <*server-trustpoint*> **pkcs12 bootflash:** *:*<*server-ca-file*> <*pkcs-password*> **Example:**	Imports the server pkcs12 file to the trustpoint. If the file does not contain the complete server and CA certification chain, the import may fail.

	Command or Action	Purpose
	`switch# crypto ca import tls_server_trustpoint pkcs12 bootflash:server.pfx test`	

Step 4	**feature grpc**
	Example:
	`switch# feature grpc`
	`switch(config)#`
		Enables the gRPC agent, which supports the gNMI interface for dial-in.

Step 5	(Optional) **grpc port** *port-id* **Example:** `switch(config)# grpc port 50051`	Configures the port number. The range of port-id is from 1024 to 65535. 50051 is the default.
Step 6	**grpc certificate** *certificate-id* **Example:** `switch(config)# grpc certificate cert-1`	Specify the certificate trustpoint ID. This shall match the trustpoint name imported at step 2-3.
Step 7	(Optional) **use-vrf default** **Example:** `switch(config)# grpc use-vrf default`	Enables the gRPC agent to accept incoming (dial-in) RPC requests from the default VRF. This step enables the default VRF to process incoming RPC requests. By default, the management VRF processes incoming RPC requests when the gRPC feature is enabled.

Example for Generating PKCS12 file

If the purpose is to experiment the feature, the following is an example for generating a self-signed pkcs12 bundle in the switch bash shell, then import the pkcs12 file to the trustpoint.

Note This task is an example of how a certificate can be generated on a switch. You can also generate a certificate in any Linux environment. In a production environment, you should consider using a CA signed certificate.

SUMMARY STEPS

1. Generate the self-signed key and pem files.
2. After generating the key and pem files, you must bundle the key and pem files for use in the trustpoint CA Association.
3. Set up the trustpoint CA Association by importing in the pkcs12 bundle into the trustpoint.
4. Verify the setup.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	Generate the self-signed key and pem files.	`switch# run bash sudo su` `bash-4.3# openssl req -x509 -newkey rsa:2048`

	Command or Action	Purpose
		`-keyout self_sign2048.key -out self_sign2048.pem -days 365 -nodes`

Step 2	After generating the key and pem files, you must bundle the key and pem files for use in the trustpoint CA Association.	`After generating the key and pem files, you must bundle the key and pem files for use in the trustpoint CA Association.`
		`switch# run bash sudo su bash-4.3# cd /bootflash/ bash-4.3# openssl pkcs12 -export -out self_sign2048.pfx -inkey self_sign2048.key -in self_sign2048.pem -certfile self_sign2048.pem -password pass:Ciscolab123!`
		`bash-4.3# exit`

Step 3	Set up the trustpoint CA Association by importing in the pkcs12 bundle into the trustpoint.	`switch(config)# crypto ca trustpoint mytrustpoint switch(config-trustpoint)# crypto ca import mytrustpoint pkcs12 self_sign2048.pfx Ciscolab123!`

Step 4	Verify the setup.	`switch(config)# show crypto ca certificates`
		`Trustpoint: mytrustpoint certificate:`
		`subject= /C=US/O=Cisco Systems, Inc./OU=CSG/L=San Jose/ST=CA/street=3700 Cisco Way/postalCode=95134/CN=ems.cisco.com/serialNumber=FGE18420K0R`
		`issuer= /C=US/O=Cisco Systems, Inc./OU=CSG/L=San Jose/ST=CA/street=3700 Cisco Way/postalCode=95134/CN=ems.cisco.com/serialNumber=FGE18420K0R`
		`serial=0413`
		`notBefore=Nov 5 16:48:58 2015 GMT`
		`notAfter=Nov 5 16:48:58 2035 GMT`
		`SHA1 Fingerprint=2E:99:2C:CE:2F:C3:B4:EC:C7:E2:52:3A:19:A2:10:D0:54:CA:79:3E`
		`purposes: sslserver sslclient`
		`CA certificate 0: subject= /C=US/O=Cisco Systems, Inc./OU=CSG/L=San Jose/ST=CA/street=3700 Cisco Way/postalCode=95134/CN=ems.cisco.com/serialNumber=FGE18420K0R`
		`issuer= /C=US/O=Cisco Systems, Inc./OU=CSG/L=San Jose/ST=CA/street=3700 Cisco Way/postalCode=95134/CN=ems.cisco.com/serialNumber=FGE18420K0R`
		`serial=0413`
		`notBefore=Nov 5 16:48:58 2015 GMT notAfter=Nov 5 16:48:58 2035 GMT`
		`SHA1 Fingerprint=2E:99:2C:CE:2F:C3:B4:EC:C7:E2:52:3A:19:A2:10:D0:54:CA:79:3E`
		`purposes: sslserver sslclient`

Configuring gRPC Client Certificate Authentication

gRPC also allows to authenticate the client based on the cert files (public key). This provides password-less authentication, which is considered more secure than password-based authentication.

Before you begin

Prepare the CA certificate chain for client authentication.

This is not specific to gRPC, so it is possible to re-use the existing file deployed by other Nexus services.

SUMMARY STEPS

1. **configure terminal**

2. (Optional) **crypto ca trustpoint** <*server-trustpoint*>
3. (Optional) **crypto ca authenticate** <*client-root-trustpoint*>
4. **grpc client root certificate** <*client-root-trustpoint*>

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	(Optional) **crypto ca trustpoint** <*server-trustpoint*> **Example:** `switch# crypto ca trustpoint tls_server_trustpoint`	Creates a trustpoint for server authentication. Step 2-3 is optional if there already exist usable server trustpoint.
Step 3	(Optional) **crypto ca authenticate** <*client-root-trustpoint*> **Example:** `switch# crypto ca authenticate client_trustpoint`	Imports the client certificate. This step requires manual copy paste. Please follow the instruction.
Step 4	**grpc client root certificate** <*client-root-trustpoint*> **Example:** `switch(config)# grpc client root certificate client_trustpoint`	Enters the trustpoint to host the client CA root certificate.

Example

Configuraion example

To config the gRPC client root cert, the recommendation is to go through the user's own specific workflow to fetch the CA certificate.

This section provides an example configuration sequence for illustration.

1. Prepare the Client Root CA Certificates.
2. Import the certificate

   When you have generated a new certificate to the client root successfully, following are the sample commands to configure them in the switch, and their output.

```
switch(config)# crypto ca trustpoint my_client_trustpoint
switch(config-trustpoint)# crypto ca authenticate my_client_trustpoint
input (cut & paste) CA certificate (chain) in PEM format; end the input with a line
containing only END OF INPUT :
-----BEGIN CERTIFICATE-----
MIIDUDCCAjigAwIBAgIJAJLisBKCGjQOMA0GCSqGSIb3DQEBCwUAMD0xCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJDQTERMA8GA1UEBwwIU2FuIEpvc2UxDjAMBgNVBAoM
BUNpc2NvMB4XDTIwMTAxNDIwNTYyN1oXDTQwMTAwOTIwNTYyN1owPTELMAkGA1UE
BhMCVVMxCzAJBgNVBAgMAkNBMREwDwYDVQQHDAhTYW4gSm9zZTEOMAwGA1UECgwF
```

```
Q2lzY28wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDEX7qZ2EdogZU4
EW0NSpB3EjY0nSlFLOw/iLKSXfIiQJD0Qhaw16fDnnYZj6vzWEa0ls8canqHCXQl
gUyxFOdGDXa6neQFTqLowSA6UCSQA+eenN2PIpMOjfdFpaPiHu3mmcTI1xP39Ti3
/y548NNORSepApBNkZ1rJSB6Cu9AIFMZgrZXFqDKBGSUOf/CPnvIDZeLcun+zpUu
CxJLA76Et4buPMysuRqMGHIX8CYw8MtjmuCuCTHXNN31ghhgpFxfrW/69pykjU3R
YOrwlSUkvYQhtefHuTHBmqym7MFoBEchwrlC5YTduDzmOvtkhsmpogRe3BiIBx45
AnZdtdi1AgMBAAGjUzBRMB0GA1UdDgQWBBSh3IqRrm+mtB5GNsoLXFb3bAVg5TAf
BgNVHSMEGDAWgBSh3IqRrm+mtB5GNsoLXFb3bAVg5TAPBgNVHRMBAf8EBTADAQH/
MA0GCSqGSIb3DQEBCwUAA4IBAQAZ4Fpc6lRKzBGJQ/7oK1FNcTX/YXkneXDk7Zrj
8W0RS0Khxgke97d2Cwl5P5reXO27kvXsnsz/VZn7JYGUvGSlxTlcCb6x6wNBr4Qr
t9qDBu+LykwqNOFe4VCAv6e4cMXNbH2wHBVS/NSoWnM2FGZ10VppjEGFm6OM+N6z
8n4/rWslfWFbn7T7xHH+Nl0Ffc+8q8h37opyCnb0ILj+a4rnyus8xXJPQb05DfJe
ahPNfdEsXKDOWkrSDtmKwtWDqdtjSQC4xioKHoshnNgWBJbovPlMQ64UrajBycwV
z9snWBm6p9SdTsV92YwF1tRGUqpcI9olsBgH7FUVU1hmHDWE
-----END CERTIFICATE----END OF INPUT
Fingerprint(s): SHA1
Fingerprint=0A:61:F8:40:A0:1A:C7:AF:F2:F7:D9:C7:12:AE:29:15:52:9D:D2:AE
Do you accept this certificate? [yes/no]:yes switch(config)#
NOTE: Use the CA Certificate from the .pem file content.
switch# show crypto ca certificates Trustpoint: my_client_trustpoint CA certificate 0:
subject=C = US, ST = CA, L = San Jose, O = Cisco
issuer=C = US, ST = CA, L = San Jose, O = Cisco
serial=B7E30B8F4168FB87 notBefore=Oct 1 17:29:47 2020 GMT notAfter=Sep 26 17:29:47 2040
 GMT
SHA1 Fingerprint=E4:91:4E:D4:41:D2:7D:C0:5A:E8:F7:2D:32:81:B3:37:94:68:89:10 purposes:
sslserver sslclient
```

3. Associating Trustpoints to gRPC

   When you have configured a new certificate to the client root successfully, the following is the output example for associating trustpoints to gRPC server on the switch:

```
switch(config)# feature grpc
switch(config)# grpc client root certificate my_client_trustpoint switch(config)# show
run grpc
!Command: show running-config grpc
!Running configuration last done at: Wed Dec 16 20:18:35 2020
!Time: Wed Dec 16 20:18:40 2020
version 10.1(1) Bios:version N/A feature grpc
grpc gnmi max-concurrent-calls 14 grpc use-vrf default grpc certificate my_trustpoint
grpc client root certificate my_client_trustpoint grpc port 50003
```

4. Validating the Certificate Details

   When you have successfully associated the trustpoints to gRPC on the switch, the following is the output example for validating the certificate details:

```
switch# show grpc gnmi service statistics
============== gRPC Endpoint ==============
Vrf : management
Server address : [::]:50003
Cert notBefore : Mar 13 19:05:24 2020 GMT
Cert notAfter : Nov 20 19:05:24 2033 GMT
Client Root Cert notBefore : Oct 1 17:29:47 2020 GMT
Client Root Cert notAfter : Sep 26 17:29:47 2040 GMT
...
```

5. Verifying the Connection using Client Certificate Authentication for any gNMI Clients.

   The client certificate requests with a private key (pkey) and ca chain (cchain). The password is now optional. Make sure the client needs to supply full chain from to root CA to its client cert.

6. For removing trustpoint reference from gRPC (no command) use the following command:

```
switch(config)# no grpc client root certificate my_client_trustpoint
```

The command will remove the trustpoint reference only from gRPC agent, but the trustpoints CA certificates will NOT be removed. Connections that use client certificate authentication to gRPC server on switch will not establish, but basic authentication with username and password will go through.

Configuring gRPC Client Certificate Authentication With CA bundle

This requires to import a "pkcs7" file to the trustpoint, and such file shall include all the relevent root, intermediate CA, sub CAs.



Note Configuring or removing the root certificate for client authentication will cause gRPC to restart the servers.

Before you begin

Prepare and sign and the required certificate files for the client authentication. This is not specific to gRPC, so it is possible to re-use the existing trustpoint files.

SUMMARY STEPS

1. **configure terminal**
2. (Optional)**crypto ca trustpoint** <*client-root-trustpoint*>
3. (Optional)**crypto ca import** <*client-root-trustpoint*> **pkcs7** *bootflash:<p7b file>*
4. **grpc client root certificate** <*client-root-trustpoint*>

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	(Optional)**crypto ca trustpoint** <*client-root-trustpoint*> **Example:** `switch# crypto ca trustpoint client_trustpoint`	Create a trustpoint for client authentication.
Step 3	(Optional)**crypto ca import** <*client-root-trustpoint*> **pkcs7** *bootflash:<p7b file>* **Example:** `switch# crypto ca import client_trustpoint bootflash:my_ca_bundle.p7b`	Import the CA bundle. The input is a p7b file, which includes the certificates. This would automatically create multiple child trustpoint, each represents a unique certificate chain.

	Command or Action	Purpose
Step 4	**grpc client root certificate** <*client-root-trustpoint*> **Example:** `switch(config)# grpc client root certificate client_trustpoint`	Associate the main trustpoint to gRPC.

Configuring NGINX proxy for GRPC

Like Netconf and Restconf, gRPC agent runs on a dedicate server/port. The gRPC client would need to directly connect to the gRPC agent/server.

Starting from release 10.3(3)F, NX-OS NGINX can also act as GRPC proxy by relaying the gRPC traffic and can be useful for certain use cases.

- GRPC port blocked: The GRPC agent listens on port 50051. If this port is blocked by the firewall, GRPC clients can access the gRPC services indirectly through the NGINX HTTPS port 443.
- Increased VRF support: Currently GRPC services are accessible only via management or one user specified VRFs. NGINX proxy can forward the gRPC requests from any VRF.

This new support does not affect existing behavior. The GRPC client can still connect to the GRPC agent directly. It can also instead connect to the NGINX server, which then proxies the GRPC requests to the GRPC agent. Note that such redirection deems to incur additional request-response latency.

All server and client authentication will be handled by NGINX. Just enable GRPC and configure NGINX server certificate and/or client certificates.

Before you begin

Enable the grpc feature.

Prepare the NX-API certificates. See Using NX-API CLI for detail.

SUMMARY STEPS

1. **configure terminal**
2. **feature nxapi**
3. **nxapi certificate httpscrt certfile** *cert-file*
4. **nxapi certificate httpscrt keyfile** *key-file* **password** <*password*>
5. **nxapi certificate enable**
6. (Optional) **crypto ca trustpoint** <*trustpoint*>
7. (Optional) **crypto ca authenticate** <*trustpoint*>
8. (Optional) **nxapi client certificate authentication**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**feature nxapi**
	Example:
	`switch# feature nxapi`
	`switch(config)#`
		Enables the feature nxapi.

Step 3	**nxapi certificate httpscrt certfile** *cert-file* **Example:** `switch# nxapi certificate httpscrt certfile bootflash:nxapi.crt`	Configures the cert file.
Step 4	**nxapi certificate httpscrt keyfile** *key-file* **password** *<password>* **Example:** `switch# nxapi certificate httpskey keyfile bootflash:nxapi.key password cisco123`	Configures the key file.
Step 5	**nxapi certificate enable** **Example:** `switch# nxapi certificate enable`	Enables the certificate authentication.
Step 6	(Optional) **crypto ca trustpoint** *<trustpoint>* **Example:** `switch# crypto ca trustpoint grpcClientCA`	Creates a trustpoint for server authentication.
Step 7	(Optional) **crypto ca authenticate** *<trustpoint>* **Example:** `switch# crypto ca authenticate grpcClientCA`	Imports the client root certificate to the trustpoint.
Step 8	(Optional) **nxapi client certificate authentication** **Example:** `switch# nxapi client certificate authentication`	Enables the client certificate authentication.

Troubleshooting

Check Feature Status

- In Cisco NX-OS device, enter the **show feature grpc** command to check the agent config.
- To view the status of the gRPC agent, use the **show feature** command.

```
switch-1# show feature | grep grpc
restconf 1 enabled
switch-1#
```

Check Connectivity

From a client system, ping the management port of the switch to verify that the switch is reachable.

Client tools usage example

NX-OS provides reference client **py_gnmicli_kpi.py** and **gnmi_console_kpi.py** which can connect to Unix domain socket and they can be executed within the switch.

Client tools

Client tools which can be used for debugging purpose, is packaged into image.

```
bash-5.1# cd /opt/mtx/scripts
bash-5.1# tar xf kpi.tgz
bash-5.1# cd /opt/mtx/scripts/kpi
bash-5.1# ls -l
total 144
-rwxr-xr-x 1 43720 floppy 37294 Jul 14 13:56 gnmi_console_kpi.py
-rwxr-xr-x 1 43720 floppy 444 Jul 14 13:56 gnxi_console_kpi
-rwxr-xr-x 1 43720 floppy 4723 Jul 14 13:56 kpi_tool.py
drwxr-xr-x 5 43720 floppy 140 Jul 14 13:56 lib
-rwxr-xr-x 1 43720 floppy 61977 Sep 23 12:23 libgnoicli.py
-rwxr-xr-x 1 43720 floppy 25862 Sep 23 12:23 py_gnmicli_kpi.py
bash-5.1#
```

Send request without username and password

In this example, the client sends the gNMI requests without giving the -user / -pass options. When it connects via the **grpc_be_np.sock**, the server would assign the default 'admin' user to the connection.

```
bash-5.1# python py_gnmicli_kpi.py -t unix:/var/run/grpc/grpc_be_np.sock -n -m capabilities
bash-5.1# python gnmi_console_kpi.py --host unix:/var/run/grpc/grpc_be_np.sock --operation
 Subscribe -s on_change -x System/intf-items/phys-items/PhysIf-list[id=eth1/5]/descr
```

Send request with username only

In this example, the client sends the gNMI requests with the -user foo. This can connect as well, and the user **foo** would be used process the request. Please note that most the gRPC operations can only be done by the **network-admin** group. Therefore, make sure you have configure such username with the proper role.

```
bash-5.1# python py_gnmicli_kpi.py -t unix:/var/run/grpc/grpc_be_np.sock -n -user foo -m
capabilities

bash-5.1# vsh -c 'show run | grep foo'
username foo role network-admin
```

Docker container

When using client tools from a docker container, please mount directory /var/run into the docker container with flag **-v /var/run:/var/run**.