gNMI-gRPC ネットワーク管理インターフェイス

gNMI について

gNMI は、トランスポート プロトコルとして gRPC（Google リモート プロシージャ コール）を使用します。

Cisco NX-OS は、Cisco Nexus 9000 シリーズ スイッチで実行されるテレメトリ アプリケーションへのダイヤルイン サブスクリプション用に gNMI をサポートします。過去のリリースでは gRPC を介したテレメトリ イベントがサポートされていましたが、スイッチはテレメトリ データをテレメトリ レシーバにプッシュしていました。この方法はダイヤル アウトと呼ばれていました。

gNMI を使用すると、アプリケーションはスイッチから情報をプルできます。サポートされているテレメトリ機能を学習し、必要なテレメトリ サービスのみをサブスクライブすることで、特定のテレメトリ サービスにサブスクライブします。

表 *1:* サポートされる *gNMI RPC*

gNMI RPC	サポート対象
機能	はい
結果	はい
設定	はい
登録	はい

gNMI サブスクライブ RPC

Cisco NX-OS 9.3(1) リリース以降では、次の gNMI サブスクリプション機能がサポートされています。

表 *2:* サブスクライブ オプション

タイプ	サブタイプ	サポートの有無	説明
[1 回（Once）]		はい	スイッチは、指定されたすべてのパスに対して現在の値を 1 回だけ送信します。
ポーリング（Poll）		はい	スイッチは、ポーリング メッセージを受信するたびに、指定されたすべてのパスの現在の値を送信します。

タイプ	サブタイプ	サポートの有無	説明
ストリーム	サンプル	はい	ストリームサンプル間隔ごとに1回、スイッチは指定されたすべてのパスの現在の値を送信します。サポートされるサンプル間隔の範囲は 1 ～ 604800 秒です。 デフォルトのサンプル間隔は 10 秒です。
	[変更時（On_Change）]	はい	スイッチは初期状態として現在の値を送信しますが、指定されたパスのいずれかに作成、変更、または削除などの変更が発生した場合にのみ値を更新します。
	[ターゲット定義（Target_Defined）]	はい	ターゲット定義モードを指定してサブスクリプションを作成する場合、ターゲットは、作成するサブスクリプションの最適なタイプを定義する必要があります。

（注） 10.2(1)F リリース以降、Target_Defined サブタイプのサブスクライブ オプションがサポートされています。

Cisco NX-OS リリース 10.2(3)F 以降では、gNMI サブスクリプションのキープアライブ間隔を変更する新しい CLI コマンドが導入されています。設定可能な限界値は 600 ～ 86400 秒です。

コマンドは **"[no] grpc gnmi keepalive-timeout <timeout>"** です。たとえば、**switch(config)# grpc gnmi keepalive-timeout 600** と入力します。

次に、CLI コマンドを確認する例を示します。

```
Verify in show statistics cmd

switch(config)# sh grpc gnmi service statistics

==============
```

```
gRPC Endpoint
=============

Vrf            : management
Server address : [::]:50051

Cert notBefore : Feb  6 01:15:06 2022 GMT
Cert notAfter  : Feb  7 01:15:06 2022 GMT
Client Root Cert notBefore : n/a
Client Root Cert notAfter  : n/a

Max concurrent calls            :  8
Listen calls                    :  1
Active calls                    :  0
KeepAlive Timeout                    :  1000
```

CLI コマンドの注意事項は次のとおりです。

- gnmi サーバーは、指定された間隔ごとに空の応答をサブスクリプション クライアントに送信します。
- 目的は、不正な接続や無意味な接続を検出して、クリーンアップすることです。
- デフォルトのキープアライブ間隔は 600 秒です。
- このコマンドは、間隔をユーザー指定の値に変更します。

オプションの SUBSCRIBE フラグ

SUBSCRIBE オプションでは、表にリストされているオプションへの応答を変更するオプションのフラグを使用できます。Cisco NX-OS リリース 9.3(1) 以降では、[更新のみ（updates_only）] オプション フラグがサポートされています。これは、ON_CHANGE サブスクリプションに適用されます。このフラグが設定されている場合、スイッチは通常最初の応答で送信される初期スナップショット データ（現在の状態）を抑制します。

次のフラグはサポートされていません。

- [エイリアス（aliases）]
- [集約許可（allow_aggregation）]
- [拡張（extensions）]
- prefix
- [qos]

Cisco NX-OS リリース 10.2(3)F 以降、次のフラグがサポートされています。

- [ハートビート間隔（heartbeat_interval）]
- [冗長抑制（suppress_redundant）]

サンプリングされたサブスクリプションでの [冗長抑制（suppress_redundant）] の動作を変更するために、[ハートビート間隔（Heartbeat_interval）] を指定できます。この場合、ターゲットは、suppress_redundant フラグが true に設定されているかどうかに関係なく、ハートビート間隔

ごとに1つのテレメトリ更新を生成する必要があります。この値は、ナノ秒単位の符号なし64ビット整数として指定されます。

サブスクリプション メッセージの suppress_redundant フィールドは、サンプリングされたサブスクリプションに設定できます。true に設定されている場合、レポートされているパスの値が最後の更新が生成されてから変更されていない限り、ターゲットはテレメトリ更新メッセージを生成してはなりません。更新は、変更されたサブスクリプション内の個々のリーフノードに対してのみ生成する必要があります。

たとえば、B ノードから分岐するリーフ C と D がある /A/B へのサブスクリプションで、C の値が変更されても D が変更されていない場合、C の更新を生成する必要があるのに対し、D の更新を生成してはなりません。

次に、サポートされているオプションの SUBSCRIBE フラグの例を示します。

```
{
 "SubscribeRequest":
 [
  {
   "subscribe":
   {
    "subscription":
    [
     {
      "_comment" : "1st subscription path",
      "path":
      {
       "origin": "openconfig",
       "elem":
       [
        {
         "name": "interfaces/interface[name=eht1/1]"
        }
       ]
      },
      "mode": "SAMPLE",
      "heartbeat_interval": 30000000000
      "suppress-redundant": true
     },
    "mode": "STREAM",
    "allow_aggregation" : false,
    "use_models":
    [
     {
      "name": "DME",
      "organization": "Cisco Systems, Inc.",
      "version": "1.0.0"
     }
    ],
    "encoding": "JSON"
   }
  }
```

サブスクライブ フラグのサポート メトリクスは次のとおりです。

表 *3 : SUBSCRIBE* フラグのサポート メトリクス

サブスクリプションタイプ	[ハートビート間隔（**heartbeat_interval**）]	[冗長抑制（**suppress_redundant**）]
[変更時（On_Change）]	発信元：デバイス YANG、OpenConfig YANG、DME	該当なし
サンプル	発信元：デバイス YANG、OpenConfig YANG、DME	発信元：デバイス YANG、OpenConfig YANG

Guidelines and Limitations for gNMI

Following are the guidelines and limitations for gNMI:

- Beginning with Cisco NX-OS Release 9.3(5), Get and Set are supported.
- gNMI queries do not support wildcards in paths.
- When you enable gRPC on both the management VRF and default VRF and later disable on the default VRF, the gNMI notifications 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 and any existing gRPC configuration commands. For example, **grpc certificate** or **grpc port**. You must also resubscribe to any existing notifications on the management VRF.
- When you attempt to subscribe an OpenConfig routing policy with a preexisting CLI configuration like the following, it returns empty values due to the current implementation of the OpenConfig model.

  ```
  ip prefix-list bgp_v4_drop seq 5 deny 125.2.0.0/16 le 32
  ipv6 prefix-list bgp_v6_drop seq 5 deny cafe:125:2::/48 le 128

  using the xpath

  openconfig-routing-policy:/routing-policy/defined-sets/prefix-sets/prefix-set[name=bgp_v4_drop]/config
  openconfig-routing-policy:/routing-policy/defined-sets/prefix-sets/prefix-set[name=bgp_v6_drop]/config
  ```

- Only server certificate authentication takes place. The client certificate is not authenticated by the server.
- 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 and the status line displays an error like the following:

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

  Unconfigure and configure the proper certificate command to restore the service.
- Beginning with Cisco NX-OS Release 9.3(3), 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.

- Use of origin, use_models, or both, is optional for gNMI subscriptions.
- gNMI Subscription supports Cisco DME and Device YANG data models. Beginning with Cisco NX-OS Release 9.3(3), Subscribe supports the OpenConfig model.
- For Cisco NX-OS prior to 9.3(x), information about supported platforms, see *Platform Support for Programmability Features* in the guide for that release. Starting with Cisco NX-OS release 9.3(x), for information about supported platforms, see the Nexus Switch Platform Matrix.
- The feature supports JSON and gnmi.proto encoding. The feature does not support protobuf.any encoding.
- Each gNMI message has a maximum size of 12 MB. If the amount of collected data exceeds the 12 MB maximum, the collected data is dropped. Applies to gNMI ON_CHANGE mode only.

  You can avoid this situation by creating more focused subscriptions that handle smaller, more granular data-collection sets. So, instead of subscribing to one higher-level path, create multiple subscriptions for different, lower-level parts of the path.
- Across all subscriptions, there is support of up to 150K aggregate MOs. Subscribing to more MOs can lead to collection data drops.
- The feature does not support a path prefix in the Subscription request, but the Subscription can contain an empty prefix field.
- The gRPC process that supports gNMI uses the HIGH_PRIO control group, which limits the CPU usage to 75% of CPU and memory to 1.5 GB.
- The **show grpc gnmi** command has the following considerations:
  - The gRPC agent retains gNMI calls for a maximum of one hour after the call has ended.
  - If the total number of calls exceeds 2000, the gRPC agent purges ended calls based on the internal cleanup routine.
- Beginning with Cisco NX-OS Release 10.2(3)F, on change subscription of Device YANG ephemeral data (Accounting-log and Multicast) is supported.

The gRPC server runs in the management VRF. As a result, the gRPC process communicates only in this VRF forcing the management interface to support all gRPC calls.

gRPC functionality now includes the default VRF for a total of two gRPC servers on each switch. You can run one gRPC server in each VRF, or run only one gRPC server in the management VRF. Supporting a gRPC in 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.

The following are the limitations for gNMI:

- multi-level wildcard "..." in path is not allowed
- wildcard '*' in the top of the path is not allowed
- wildcard '*' in key name is not allowed
- wildcard and value cannot be mixed in keys

The following table shows the wildcard support details for gNMI:

Table 4: Wildcard Support for gNMI Requests

Type of Request	Wildcard Support
gNMI GET	YES
gNMI SET	NO
gNMI SUBSCRIBE, ONCE	YES
gNMI SUBSCRIBE, POLL	YES
gNMI SUBSCRIBE, STREAM, SAMPLE	YES
gNMI SUBSCRIBE, STREAM, TARGET_DEFINED	YES
gNMI SUBSCRIBE, STREAM, ON_CHANGE	NO

Configuring gNMI

Configure the gNMI feature through the **grpc gnmi** commands.

To import certificates used by the **grpc certificate** command onto the switch, see the Installing Identity Certificates section of the Cisco Nexus 9000 Series NX-OS Security Configuration Guide, Release 9.3(x).

Note When modifying the installed identity certificates or **grpc port** and **grpc certificate** values, the gRPC server might restart to apply the changes. When the gRPC server restarts, any active subscription is dropped and you must resubscribe.

SUMMARY STEPS

1. **configure terminal**
2. **feature grpc**

3. (Optional) **grpc port** *port-id*
4. **grpc certificate** *certificate-id*
5. **grpc gnmi max-concurrent-call** *number*
6. (Optional) **grpc use-vrf default**
7. **grpc gnmi subscription target-defined min-interval**
8. **grpc gnmi subscription query-condition keep-data-timestamp**

DETAILED STEPS

	Command or Action	Purpose

ステップ 1	**configure terminal**
	Example:
	`switch-1# configure terminal`
	`switch-1(config)#`
		Enters global configuration mode.

ステップ 2	**feature grpc**
	Example:
	`switch-1# feature grpc`
	`switch-1(config)#`
		Enables the gRPC agent, which supports the gNMI interface for dial-in.

ステップ 3	(Optional) **grpc port** *port-id* **Example:** `switch-1(config)# grpc port 50051`	Configure the port number. The range of *port-id* is from 1024 to 65535. 50051 is the default. **Note** This command is available beginning with Cisco NX-OS Release 9.3(3).
ステップ 4	**grpc certificate** *certificate-id* **Example:** `switch-1(config)# grpc certificate cert-1`	Specify the certificate trustpoint ID. For more information, see the Installing Identity Certificates section of the Cisco Nexus 9000 Series NX-OS Security Configuration Guide, Release 9.3(x) for importing the certificate to the switch. **Note** This command is available beginning with Cisco NX-OS Release 9.3(3).

ステップ 5	**grpc gnmi max-concurrent-call** *number*
	Example:
	`switch-1(config)# grpc gnmi max-concurrent-call 16`
	`switch-1(config)#`
		Sets the limit of simultaneous dial-in calls to the gNMI server on the switch. Configure a limit from 1 through 16. The default limit is 8.
		The maximum value that you configure is for each VRF. If you set a limit of 16 and gNMI is configured for both management and default VRFs, each VRF supports 16 simultaneous gNMI calls.
		This command does not affect and ongoing or in-progress gNMI calls. Instead, gRPC enforces the limit on new calls, so any in-progress calls are unaffected and allowed to complete.
		Note The configured limit does not affect the gRPCConfigOper service.

	Command or Action	Purpose
ステップ 6	(Optional) **grpc 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. **Note** Both VRFs process requests individually, so that requests do not cross between VRFs.
ステップ 7	**grpc gnmi subscription target-defined min-interval** **Example:** `switch(config)# grpc gnmi subscription target-defined min-interval ? <1-65535> Default 30`	Allows the user to modify the default target-defined sample interval from 30 seconds to other value.
ステップ 8	**grpc gnmi subscription query-condition keep-data-timestamp**	This command enables sample/once/poll subscriptions to get timestamp from database when the data was last updated. **Note** • This feature is supported only for PROTO and not JSON encoding. • This feature is supported only for once, poll, sample, and not on_change. • This feature is supported for DME, YANG, and OpenConfig data sources. • Most of the other properties are supported, but for the properties which are not supported, this feature will default back to the collection time instead of last database change time. **Note** This feature would generate verbose responses for each timestamp. If the user client is not able to consume the messages on time, the switch may need to drop the collection/messages.

サーバー証明書の構成

TLS 証明書を設定し、スイッチに正常にインポートした場合の **show grpc gnmi service statistics** コマンドの出力例を次に示します。

```
switch(config)# sh grpc gnmi service statistics

============== gRPC Endpoint
Vrf : management
```

```
Server address : [::]:50051

Cert notBefore : Nov 5 16:48:58 2015 GMT
Cert notAfter : Nov 5 16:48:58 2035 GMT
Client Root Cert notBefore : n/a
Client Root Cert notAfter : n/a

Max concurrent calls : 8
Listen calls : 1
Active calls : 0
KeepAlive Timeout : 120

Number of created calls : 1
Number of bad calls : 0

Subscription stream/once/poll : 0/0/0

Max gNMI::Get concurrent : 6
Max grpc message size : 25165824
gNMI Synchronous calls : 3
gNMI Synchronous errors : 3
gNMI Adapter errors : 3
gNMI Dtx errors : 0
```

gNMI は gRPC を介して通信し、TLS を使用してスイッチとクライアント間のチャネルをセキュアにします。デフォルトのハードコードされた gRPC 証明書は、スイッチに同梱されなくなりました。デフォルトの動作は、次に示すように、スイッチで生成される有効期限が 1 日の自己署名キーと証明書です。

証明書の有効期限が切れているか、正常にインストールできなかった場合は、1 日限りのデフォルト証明書が表示されます。次に、**show grpc gnmi service statistics** コマンドの出力を示します。

```
#show grpc gnmi service statistics

=============
gRPC Endpoint
=============

Vrf            : management
Server address : [::]:50051

Cert notBefore : Wed Mar 11 19:43:01 PDT 2020
Cert notAfter  : Thu Mar 12 19:43:01 PDT 2020

Max concurrent calls            :  8
Listen calls                    :  1
Active calls                    :  0

Number of created calls         :  1
Number of bad calls             :  0

Subscription stream/once/poll   :  0/0/0
```

有効期限は 1 日ですが、この一時証明書を使用してテストを簡単に行えます。長期的には、新しいキー/証明書を生成する必要があります。

キー/証明書の生成の例

キー/証明書を生成するには、次の例に従います。

- Generating and Configuring Key/Certificate Examples for Cisco NX-OS Release 9.3(3) and Later

Generating and Configuring Key/Certificate Examples for Cisco NX-OS Release 9.3(3) and Later

The following is an example for generating key/certificate.

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.

For more information on generating identity certificates, see the Installing Identity Certificates section of the *Cisco Nexus 9000 Series NX-OS Security Configuration Guide, Release 9.3(x)*.

ステップ 1 Generate the selfsigned key and pem files.

a) switch# openssl req -x509 -newkey rsa:2048 -keyout self_sign2048.key -out self_sign2048.pem -days 365 -nodes

ステップ 2 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
```

ステップ 3 Set up the trustpoint CA Association by inputting in the pkcs12 bundle into the trustpoint.

```
switch(config)# crypto ca import mytrustpoint pkcs12 switch(config-trustpoint)# crypto ca import
mytrustpoint pkcs12 self_sign2048.pfx Ciscolab123!
```

ステップ 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
```

ステップ 5 Configure gRPC to use the trustpoint.

```
switch(config)# grpc certificate mytrustpoint
switch(config)# show run grpc

!Command: show running-config grpc
!Running configuration last done at: Thu Jul  2 12:24:02 2020
!Time: Thu Jul  2 12:24:05 2020

version 9.3(5) Bios:version 05.38
feature grpc

grpc gnmi max-concurrent-calls 16
grpc use-vrf default
grpc certificate mytrustpoint
```

ステップ 6 Verify gRPC is now using the certificate.

```
switch# show grpc gnmi service statistics

=============
gRPC Endpoint
=============

Vrf : management
Server address : [::]:50051

Cert notBefore : Nov 5 16:48:58 2015 GMT
Cert notAfter : Nov 5 16:48:58 2035 GMT

Max concurrent calls : 16
Listen calls : 1
Active calls : 0

Number of created calls : 953
Number of bad calls : 0

Subscription stream/once/poll : 476/238/238

Max gNMI::Get concurrent : 5
Max grpc message size : 8388608
gNMI Synchronous calls : 10
gNMI Synchronous errors : 0
gNMI Adapter errors : 0
gNMI Dtx errors : 0
```

gNMI の確認

gNMI 構成を確認するには、次のコマンドを入力します。

コマンド	説明
show grpc gnmi service statistics	管理 VRF またはデフォルト VRF（構成されている場合）のエージェント実行ステータスの概要を表示します。また、次の項目も表示されます。 • 基本の全般的なカウンタ • 証明書の有効期限日時 （注） 証明書の有効期限が切れている場合、エージェントは要求を受け入れることができません。
show grpc gnmi rpc summary	次のステータスが表示されます。 • 受信した機能 RPC の数。 • 機能 RPC エラー。 • 受信した Get RPC の数。 • Get RPC エラー。 • 受信した Set RPC の数。 • Set RPC エラー。 • 詳細なエラー タイプとエラー数。

コマンド	説明
show grpc gnmi transactions	

コマンド	説明
	show grpc gnmi transactions コマンドは最も密度が高く、多くの情報が含まれています。これは、スイッチが受信した最新の 50 gNMI トランザクションの履歴バッファです。新しい RPC が着信すると、末尾から最も古い履歴エントリが削除されます。次に、表示内容について説明します。 • [RPC]：受信した RPC のタイプ（Get、Set、Capabilities）を示します。 • [データタイプ（DataType）]：Get の場合のみです。値は ALL、CONFIG、および STATE です。 • [セッション（Session）]：このトランザクションに割り当てられている一意のセッション ID を示します。他のログ ファイルで見つかったデータを関連付けるために使用できます。 • [入力時間（Time In）]：gNMI ハンドラが RPC を受信したときのタイムスタンプを示します。 • [期間（Duration）]：要求を受信してから応答を返すまでの時間差です（ミリ秒単位）。 • [ステータス（Status）]：クライアントに返された操作のステータス コード（0 は成功、0 以外はエラー） 。 このセクションは、単一の gNMI トランザクション内のパスごとに保持されるデータです。たとえば、単一の Get または Set です。 • [サブタイプ（subtype）]：Set RPC の場合、パスごとに要求される特定の操作（削除、更新、置換）を示します。Get の場合、サブタイプはありません。 • [dtx]：このパスが DTX の「高速」パスで処理されたかどうかを示します。ダッシュ「-」は「いいえ」を意味し、アスタリスク「*」は「はい」を意味します。 • [st]：このパスのステータス。意味は次の

コマンド	説明
	とおりです。 • OK：パスは有効で、インフラによって正常に処理されました。 • ERR:パスが無効であるか、インフラによってエラーが生成されました。 • --：パスはまだ処理されていません。有効な場合と無効な場合がありますが、まだインフラに送信されていません。 • [path]：パス

show grpc gnmi service statistics の例

```
=============
gRPC Endpoint
=============

Vrf : management
Server address : [::]:50051

Cert notBefore : Mar 13 19:05:24 2020 GMT
Cert notAfter : Nov 20 19:05:24 2033 GMT

Max concurrent calls : 8
Listen calls : 1
Active calls : 0

Number of created calls : 1
Number of bad calls : 0

Subscription stream/once/poll : 0/0/0

Max gNMI::Get concurrent : 5
Max grpc message size : 8388608
gNMI Synchronous calls : 74
gNMI Synchronous errors : 0
gNMI Adapter errors : 0
gNMI Dtx errors : 0
```

show grpc gnmi rpc summary の例

```
=============
gRPC Endpoint
=============

Vrf            : management
Server address : [::]:50051

Cert notBefore : Mar 31 20:55:02 2020 GMT
Cert notAfter  : Apr  1 20:55:02 2020 GMT
```

```
Capability rpcs    : 1
Capability errors  : 0
Get rpcs           : 53
Get errors         : 19
Set rpcs           : 23
Set errors         : 8
Resource Exhausted : 0
Option Unsupported : 6
Invalid Argument   : 18
Operation Aborted  : 1
Internal Error     : 2
Unknown Error      : 0

RPC Type        State      Last Activity  Cnt Req    Cnt Resp   Client
--------------- ---------- -------------- ---------- ---------- -----------------
-----------------------
Subscribe       Listen     04/01 07:39:21          0          0
```

show grpc gnmi transactions の例

```
=============
gRPC Endpoint
=============

Vrf            : management
Server address : [::]:50051

Cert notBefore : Mar 31 20:55:02 2020 GMT
Cert notAfter  : Apr  1 20:55:02 2020 GMT

RPC          DataType   Session         Time In              Duration(ms) Status
------------ ---------- --------------- -------------------- ------------ ------
Set          -          2361443608      04/01 07:43:49       173          0
subtype: dtx:  st: path:
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo789]

Set          -          2293989720      04/01 07:43:45       183          0
subtype: dtx:  st: path:
Replace  -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo6]

Set          -          2297110560      04/01 07:43:41       184          0
subtype: dtx:  st: path:
Update   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo7]

Set          -          0               04/01 07:43:39       0            10

Set          -          3445444384      04/01 07:43:33       3259         0
subtype: dtx:  st: path:
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo789]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo790]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo791]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo792]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo793]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo794]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo795]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo796]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo797]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo798]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo799]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo800]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo801]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo802]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo803]
```

```
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo804]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo805]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo806]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo807]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo808]

Set          -          2297474560      04/01 07:43:26       186          0
subtype: dtx:  st: path:
Update   -     OK  /System/ipv4-items/inst-items/dom-items/Dom-list[name=foo]/rt-
items/Route-list[prefix=0.0.0.0/0]/nh-items/Nexthop-list[nhAddr=192.168.1.1/32][n
hVrf=foo][nhIf=unspecified]/tag


Set          -          2294408864      04/01 07:43:17       176          13
subtype: dtx:  st: path:
Delete   -     ERR /System/intf-items/lb-items/LbRtdIf-list/descr


Set          -          0               04/01 07:43:11       0            3
subtype: dtx:  st: path:
Update   -     --  /System/intf-items/lb-items/LbRtdIf-list[id=lo4]/descr
Update   -     ERR /system/processes

Set          -          2464255200      04/01 07:43:05       708          0
subtype: dtx:  st: path:
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo2]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo777]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo778]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo779]
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo780]
Replace  -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo3]/descr
Replace  -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo4]/descr
Replace  -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo5]/descr
Update   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo3]/descr
Update   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo4]/descr
Update   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo5]/descr


Set          -          3491213208      04/01 07:42:58       14           0
subtype: dtx:  st: path:
Replace  -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo3]/descr

Set          -          3551604840      04/01 07:42:54       35           0
subtype: dtx:  st: path:
Delete   -     OK  /System/intf-items/lb-items/LbRtdIf-list[id=lo1]

Set          -          2362201592      04/01 07:42:52       13           13
subtype: dtx:  st: path:
Delete   -     ERR /System/intf-items/lb-items/LbRtdIf-list[id=lo3]/lbrtdif-items
/operSt


Set          -          0               04/01 07:42:47       0            3
subtype: dtx:  st: path:
Delete   -     ERR /System/*

Set          -          2464158360      04/01 07:42:46       172          3
subtype: dtx:  st: path:
Delete   -     ERR /system/processes/shabang

Set          -          2295440864      04/01 07:42:46       139          3
subtype: dtx:  st: path:
Delete   -     ERR /System/invalid/path
```

```
Set           -          3495739048      04/01 07:42:44       10           0


Get           ALL        3444580832      04/01 07:42:40       3            0
subtype: dtx:  st: path:
-         -     OK  /System/bgp-items/inst-items/disPolBatch

Get           ALL        0               04/01 07:42:36       0            3
subtype: dtx:  st: path:
-         -     --  /system/processes/process[pid=1]

Get           ALL        3495870472      04/01 07:42:36       2            0
subtype: dtx:  st: path:
-         *     OK  /system/processes/process[pid=1]

Get           ALL        2304485008      04/01 07:42:36       33           0
subtype: dtx:  st: path:
-         *     OK  /system/processes

Get           ALL        2464159088      04/01 07:42:36       251          0
subtype: dtx:  st: path:
-         -     OK  /system

Get           ALL        2293232352      04/01 07:42:35       258          0
subtype: dtx:  st: path:
-         -     OK  /system

Get           ALL        0               04/01 07:42:33       0            12
subtype: dtx:  st: path:
-         -     --  /intf-items
```

gRPC クライアント証明書認証

10.1(1) リリース以降、gRPC に追加の認証方式が提供されます。 10.1(1) リリースより前の gRPC サービスは、サーバー証明書のみをサポートしていました。10.1(1) 以降では、クライアント証明書のサポートも追加するように認証が拡張され、gRPC でサーバー証明書とクライアント証明書の両方を検証できるようになっています。この機能拡張により、さまざまなクライアントにパスワードなしの認証が提供されます。

Generating New Client Root CA Certificates

The following is the example for generating a new certificate to the client root:

- Trusted Certificate Authorities (CA)

Perform the following steps when you use a trusted CA such as a DigiCert:

SUMMARY STEPS

1. Download the CA certificate file.
2. Import to NX-OS using the steps in Cisco NX-OS Security Configuration Guide.

DETAILED STEPS

	Command or Action	Purpose
ステップ 1	Download the CA certificate file.	
ステップ 2	Import to NX-OS using the steps in Cisco NX-OS Security Configuration Guide.	• To create a trustpoint label, use steps in Creating a Trustpoint CA Association • To authenticate the trustpoint using the trusted CA certificates, use steps in Authenticating the CA. **Note** Use the CA Certificate from cat [CA_cert_file].

NX-OS デバイスでの生成されたルート CA 証明書の構成

クライアント root に対する新しい証明書が正常に生成されたときの、スイッチで証明書を構成するためのコマンド例とその出力を次に示します。

```
switch(config)# crypto ca trustpoint my_client_trustpoint
enticate 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
```

gRPC へのトラストポイントの関連付け

クライアントルートに新しい証明書を正常に構成した場合に、スイッチ上でトラストポイントを gRPC に関連付ける出力例を次に示します。

（注） クライアント認証用のルート証明書を構成または削除すると、gRPCプロセスが再起動します。

```
# 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
```

証明書の詳細の検証

スイッチの gRPC にトラストポイントを正常に関連付けられた場合の、証明書の詳細を検証するための出力例を次に示します。

```
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

Max concurrent calls : 14
Listen calls : 1
```

```
Active calls : 0

Number of created calls : 1
Number of bad calls : 0

Subscription stream/once/poll : 0/0/0

Max gNMI::Get concurrent : 5
Max grpc message size : 8388608
gNMI Synchronous calls : 0
gNMI Synchronous errors : 0
gNMI Adapter errors : 0
gNMI Dtx errors : 0
```

任意の gNMI クライアントのクライアント証明書認証を使用した接続の確認

クライアント証明書は、秘密キー（pkey）と CA チェーン（cchain）を使用して要求を行います。現在では、パスワードはオプションです。

```
Performing GetRequest, encoding = JSON to 172.19.199.xxx with the following gNMI Path
 -------------------------
 [elem {
  name: "System"
}
elem {
  name: "bgp-items"
}
]
The GetResponse is below
-------------------------

notification {
  timestamp: 1608071208072199559
  update {
    path {
      elem {
        name: "System"
      }
      elem {
        name: "bgp-items"
      }
    }
    val {
      json_val: ""
    }
  }
}
```

gRPC からトラストポイント参照を削除するには（no コマンド）、次のコマンドを使用します。

```
 [no] grpc client root certificate <my_client_trustpoints>
switch(config)# no grpc client root certificate my_client_trustpoint
```

コマンドは、gRPC エージェントのトラストポイント参照だけを削除します。トラストポイント CA 証明書は削除されません。スイッチ上の gRPC サーバーへのクライアント証明書認証を使用する接続は確立されませんが、ユーザー名とパスワードによる基本認証は通過します。

（注） クライアントの証明書が中間 CA によって署名されているが、上記の構成からインポートされたルート CA によって直接署名されていない場合、grpc クライアントは、ユーザー、中間 CA 証明書、およびルート CA 証明書を含む完全な証明書チェーンを提供する必要があります。

クライアント

gNMIサブスクリプションには、使用可能なクライアントがいくつかあります。このようなクライアントの 1 つは https://github.com/influxdata/telegraf/tree/master/plugins/inputs/cisco_telemetry_gnmi にあります。

gNMI のアカウンティング ログ

GNMI では、SET RPC はスイッチの設定を変更します。UPDATE、REPLACE、DELETE などの SET 要求の場合、gNMI は対応するアカウンティング ログを出力します。これには、受信した元の要求と、スイッチに適用された最終的な変更の両方が含まれます。

アカウンティングログは、**show accounting log** コマンドを使用して表示できます。

次の要求の例を考えます。

次の gNMI パスを使用して、SetRequest、encoding = JSON を localhost に実行します。

```
<<<<<<< set_delete >>>>>>>
[]
<<<<<<< set_replace >>>>>>>
[] []
<<<<<<< set_update >>>>>>>
[elem {
  name: "System"
}
elem {
  name: "tm-items"
}
elem {
  name: "certificate-items"
}
] [json_val: "{\"hostname\": \"test\", \"trustpoint\": \"foo\"}"
]
The SetRequest response is below
------------------------
response {
  path {
    elem {
      name: "System"
    }
    elem {
      name: "tm-items"
    }
    elem {
      name: "certificate-items"
    }
```

```
  }
  op: UPDATE
}
timestamp: 1656512303065384369
```

アカウンティング ログには、次の項目が含まれます。

- スイッチに適用される変更：

項目	説明
コンテキスト	セッション ID とユーザー
オペレーション	**コミット/中止**
データベース	実行または候補
ConfigMO	MO ツリーのテキスト表現。最大 3,000 文字。
ステータス	成功/失敗

例:

```
Wed Jun 29 14:18:23
2022:type=update:id=1430425712:user=admin:cmd=(COMMIT),database=[candidate],
configMo=[<topSystem childAction="" dn="sys" status="created,modified"><telemetryEntity
childAction="" rn="tm" status="created,modified"><telemetryCertificate childAction=""
hostname="test" rn="certificate" status="created,modified"
trustpoint="foo"/></telemetryEntity></topSystem>] (SUCCESS)
```

```
Wed Jun 29 14:18:23
2022:type=update:id=1430425712:user=admin:cmd=(COMMIT:CANDIDATE-TO-RUNNING),
database=[running] (SUCCESS)
```

- 受信した元の要求

項目	説明
コンテキスト	セッション ID とユーザー
オペレーション	**gNMI:SET:UPDATE**, **gNMI:SET:REPLACE**, **gNMI:SET:DELETE**, **COMMIT:CANDIDATE-TO-RUNNING**
送信元 IP（Source IP）	gNMI クライアント IP
パス	テキスト フォーマットの gNMI パス
ペイロード	受け取った JSON 要求。最大 3,000 文字。
ステータス	成功/失敗

例:

```
Wed Jun 29 14:18:23
2022:type=update:id=1430425712:user=admin:cmd=(GNMI:SET:UPDATE),sourceIp=[192.168.1.2],
```

```
path=[/System/tm-items/certificate-items],payload=[{"hostname":"test","trustpoint":"foo"}]
(SUCCESS)
```

失敗した要求の場合、失敗のシナリオによっては、ユーザーは両方のログを確認できない場合があります。

- 無効な要求：

  無効な要求は、構成の変更なしに拒否されるため、元の要求のみがログに記録されます。

  例:

  ```
  Wed Jun 29 14:18:23
  2022:type=update:id=1430425712:user=admin:cmd=(GNMI:SET:UPDATE),
  sourceIp=[192.168.1.2],path=[/System/tm-items/certificate-
  items],payload=[{"hostname":"test","trustpoint":"foo"}] (FAILED)
  ```

- さまざまな構成制限による要求の失敗：

  この場合、失敗した構成試行と元の要求の両方がログに記録されます。

  例:

  ```
  Wed Jun 29 20:52:15
  2022:type=update:id=1429663200:user=admin:cmd=(COMMIT),database=[candidate],
  configMo=[<topSystem childAction="" dn="sys"
  status="created,modified"><telemetryEntity childAction="" rn="tm"
  status="created,modified"><telemetryCertificate childAction="" filename="foo"
  hostname="test" rn="certificate" status="created,modified,replaced"
  trustpoint="foo"/></telemetryEntity></topSystem>] (FAILED)

  Wed Jun 29 20:52:15
  2022:type=update:id=1429663200:user=admin:cmd=(GNMI:SET:REPLACE),
  sourceIp=[192.168.1.2],path=[/System/tm-items/certificate-items],
  payload=[{"hostname":"test","trustpoint":"foo","filename":"foo"}] (FAILED)
  ```

- 要求のコミットに失敗しました：

  構成の試行と元の要求の両方が、失敗したコミットとともに正しく記録されます。

  例：

  ```
  Wed Jun 29 14:18:23
  2022:type=update:id=1430425712:user=admin:cmd
  (COMMIT),database=[candidate],configMo=[<topSystem childAction="" dn="sys"
  status="created,modified"><telemetryEntity childAction="" rn="tm"
  status="created,modified"><telemetryCertificate childAction="" hostname="test"
  rn="certificate" status="created,modified"
  trustpoint="foo"/></telemetryEntity></topSystem>] (SUCCESS)

  Wed Jun 29 14:18:23
  2022:type=update:id=1430425712:user=admin:cmd=(GNMI:SET:UPDATE),
  sourceIp=[192.168.1.2],path=[/System/tm-items/certificate-items],
  payload=[{"hostname":"test","trustpoint":"foo"}] (SUCCESS)

  Wed Jun 29 20:34:06
  2022:type=update:id=1429665744:user=admin:cmd=(COMMIT:CANDIDATE-TO-RUNNING),
  database=[running] (FAILED)
  ```

DMEサブスクリプションの例：PROTOエンコーディング

```
gnmi-console --host >iip> --port 50051 -u <user> -p <pass> --tls --
operation=Subscribe --rpc /root/gnmi-console/testing_bl/once/61_subscribe_bgp_dme_gpb.json

[Subscribe]--------------------------------
### Reading from file ' /root/gnmi-console/testing_bl/once/61_subscribe_bgp_dme_gpb.json
 '
Wed Jun 26 11:49:17 2019
### Generating request : 1 -----------
### Comment : ONCE request
### Delay : 2 sec(s) ...
### Delay : 2 sec(s) DONE
subscribe {
subscription {
path {
origin: "DME"
elem {
name: "sys"
}
elem {
name: "bgp"
}
}
mode: SAMPLE
}
mode: ONCE
use_models {
name: "DME"
organization: "Cisco Systems, Inc."
version: "1.0.0"
}
encoding: PROTO
}
Wed Jun 26 11:49:19 2019
Received response 1 ---------------------------
update {
timestamp: 1561574967761
prefix {
elem {
name: "sys"
}
elem {
name: "bgp"
}
}
update {
path {
elem {
}
elem {
name: "version_str"
}
}
val {
string_val: "1.0.0"
}
}
update {
path {
elem {
```

```
}
elem {
name: "node_id_str"
}
}
val {
string_val: "n9k-tm2"
}
}
update {
path {
elem {
}
elem {
name: "encoding_path"
}
}
val {
string_val: "sys/bgp"
}
}
update {
path {
elem {
}
elem {
/Received --------------------------------------
Wed Jun 26 11:49:19 2019
Received response 2 --------------------------
sync_response: true
/Received --------------------------------------
(_gnmi) [root@tm-ucs-1 gnmi-console]#
```

GNMI サブスクリプションの例：PROTO エンコーディング

GNMI サブスクリプションパス：openconfig-platform:components/component/state/ temperature/instant

```
[Subscribe]--------------------------------
  {'SubscribeRequest': [{'_comment': 'SAMPLE STREAM request ',
                         '_delay': 2,
                         'subscribe': {'allow_aggregation': False,
                                       'encoding': 'PROTO',
                                       'mode': 'STREAM',
                                       'subscription': [{'mode': 'SAMPLE',
                                                         'path': {'elem': [{'name':
'components'},
                                                                           {'name':
'component'},
                                                                           {'name':
'state'},
                                                                           {'name':
'temperature'},
                                                                           {'name':
'instant'}],
                                                                  'origin': 'openconfig'},
                                                         'sample_interval': 10000000000}],
```

```
                                        'use_models': [{'_comment': '1st module',
                                                        'name': 'openconfig-platform',
                                                        'organization': 'OpenConfig '
                                                                        'working '
                                                                        'group',
                                                        'version': '0.8.1'}]}}]}
Tue Jun 27 15:53:28 2023
### Generating request : 1 -----------
### Comment :  SAMPLE STREAM request
### Delay   :  2  sec(s) ...
### Delay   :  2  sec(s) DONE
subscribe {
  subscription {
    path {
      origin: "openconfig"
      elem {
        name: "components"
      }
      elem {
        name: "component"
      }
      elem {
        name: "state"
      }
      elem {
        name: "temperature"
      }
      elem {
        name: "instant"
      }
    }
    mode: SAMPLE
    sample_interval: 10000000000
  }
  use_models {
    name: "openconfig-platform"
    organization: "OpenConfig working group"
    version: "0.8.1"
  }
  encoding: PROTO
}

Tue Jun 27 15:53:33 2023
Received response 1 ---------------------------
update {
  timestamp: 1687906413767860352
  prefix {
    origin: "openconfig"
    elem {
      name: "components"
    }
  }
  update {
    path {
      elem {
        name: "component"
        key {
          key: "name"
          value: "27"
        }
      }
      elem {
        name: "state"
```

```
        }
        elem {
          name: "temperature"
        }
        elem {
          name: "instant"
        }
      }
      val {
        double_val: 61.0
      }
    }
}
```

NGINX の GRPC プロキシ機能

gNMI および gNOI 要求は、スイッチで実行されている gRPC エージェントによって処理されます。NGINX は、NX-API サービスへのアクセスに使用される HTTP サーバーです。リリース 10.3(3) 以降、NGINX は gNMI および gNOI 要求を受信し、それらを GRPC エージェントに転送することで、GRPC プロキシとして機能できるようになっています。これは、次のようなユースケースで役立ちます。

- GRPC ポートがブロックされている場合：GRPC エージェントはポート 50051 でリッスンします。このポートがファイアウォールによってブロックされている場合、GRPC クライアントは HTTPS ポート 443 を介して gRPC サービスにアクセスできます。
- VRF サポートの強化：現在、GRPC サービスにはデフォルト VRF と管理 VRF を介してのみアクセスできます。NGINX プロキシは、任意の VRF へのアクセスを提供できます。

Configuration Needed

Now a GRPC client can connect to the GRPC agent directly, or connect to the NGINX server, which proxy the GRPC requests to the GRPC agent.

GRPC Services Through NGINX

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

- Server certificate authentication: configure the NXAPI server certificate. See Using NX-API CLI for details.

```
feature grpc
feature nxapi
nxapi certificate httpscrt certfile bootflash:nxapi.crt
nxapi certificate httpskey keyfile bootflash:nxapi.key password cisco123

nxapi certificate enable
```

- Client certificate authentication: configure the NXAPI server and client certificates. See Using NX-API CLI for details.

```
feature grpc
feature nxapi
nxapi certificate httpscrt certfile bootflash:nxapi.crt
nxapi certificate httpskey keyfile bootflash:nxapi.key password cisco123

nxapi certificate enable
crypto ca trustpoint grpcClientCA
crypto ca authenticate grpcClientCA
nxapi client certificate authentication
```

GRPC Services Directly to The GRPC Agent

All server and client authentication will be handled by the GRPC agent. The regular GRPC server and client certificate configurations apply in this case.

機能

機能について

Capabilities RPC は、gNMI サービスの機能のリストを返します。RPC 要求に対する応答メッセージには、gNMI サービスのバージョン、バージョン管理されたデータ モデル、およびサーバーでサポートされているデータ エンコーディングが含まれます。

Guidelines and Limitations for Capabilities

Following are the guidelines and limitations for Capabilities:

- Beginning with Cisco NX-OS Release 9.3(3), Capabilities supports the OpenConfig model.
- For information about supported platforms, see Nexus Switch Platform Matrix.
- The gNMI feature supports Subscribe and Capability as options of the gNMI service.
- The feature supports JSON and gnmi.proto encoding. The feature does not support protobuf.any encoding.
- Each gNMI message has a maximum size of 12 MB. If the amount of collected data exceeds the 12-MB maximum, the collected data is dropped.

  You can avoid this situation by creating more focused subscriptions that handle smaller, more granular data-collection sets. So, instead of subscribing to one higher-level path, create multiple subscriptions for different, lower-level parts of the path.
- All paths within the same subscription request must have the same sample interval. If the same path requires different sample intervals, create multiple subscriptions.
- The feature does not support a path prefix in the Subscription request, but the Subscription can contain an empty prefix field.
- The feature supports Cisco DME and Device YANG data models.

- The gRPC process that supports gNMI uses the HIGH_PRIO cgroup, which limits the CPU usage to 75% of CPU and memory to 1.5 GB.
- The **show grpc gnmi** command has the following considerations:
  - The commands are not XMLized in this release.
  - The gRPC agent retains gNMI calls for a maximum of 1 hour after the call has ended.
  - If the total number of calls exceeds 2000, the gRPC agent purges ended calls based an internal cleanup routine.

The gRPC server runs in the management VRF. As a result, the gRPC process communicates only in this VRF forcing the management interface to support all gRPC calls.

gRPC functionality now includes the default VRF for a total of 2 gRPC servers on each Cisco Nexus 9000 switch. You can run one gRPC server in each VRF, or run only one gRPC server in the management VRF. Supporting a gRPC in the default VRF adds flexibility to offload processing gRPC calls from the management VRF, where significant traffic load might not be desirable.

If two gRPC servers are configured, be aware of the following:

- VRF boundaries are strictly enforced, so each gRPC server processes requests independent of the other, and 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.

機能のクライアント出力の例

この例では、すべての OpenConfig モデル RPM がスイッチにインストールされています。

次に、機能のクライアント出力の例を示します。

```
hostname user$ ./gnmi_cli -a 172.19.193.166:50051 -ca_crt ./grpc.pem -insecure
-capabilities
supported_models: <
  name: "Cisco-NX-OS-device"
  organization: "Cisco Systems, Inc."
  version: "2019-11-13"
>
supported_models: <
  name: "openconfig-acl"
  organization: "OpenConfig working group"
  version: "1.0.0"
>
supported_models: <
  name: "openconfig-bgp-policy"
  organization: "OpenConfig working group"
  version: "4.0.1"
>
supported_models: <
  name: "openconfig-interfaces"
  organization: "OpenConfig working group"
  version: "2.0.0"
>
supported_models: <
```

```
  name: "openconfig-if-aggregate"
  organization: "OpenConfig working group"
  version: "2.0.0"
>
supported_models: <
  name: "openconfig-if-ethernet"
  organization: "OpenConfig working group"
  version: "2.0.0"
>
supported_models: <
  name: "openconfig-if-ip"
  organization: "OpenConfig working group"
  version: "2.3.0"
>
supported_models: <
  name: "openconfig-if-ip-ext"
  organization: "OpenConfig working group"
  version: "2.3.0"
>
supported_models: <
  name: "openconfig-lacp"
  organization: "OpenConfig working group"
  version: "1.0.2"
>
supported_models: <
  name: "openconfig-lldp"
  organization: "OpenConfig working group"
  version: "0.2.1"
>
supported_models: <
  name: "openconfig-network-instance"
  organization: "OpenConfig working group"
  version: "0.11.1"
>
supported_models: <
  name: "openconfig-network-instance-policy"
  organization: "OpenConfig working group"
  version: "0.1.1"
>
supported_models: <
  name: "openconfig-ospf-policy"
  organization: "OpenConfig working group"
  version: "0.1.1"
>
supported_models: <
  name: "openconfig-platform"
  organization: "OpenConfig working group"
  version: "0.12.2"
>
supported_models: <
  name: "openconfig-platform-cpu"
  organization: "OpenConfig working group"
  version: "0.1.1"
>
supported_models: <
  name: "openconfig-platform-fan"
  organization: "OpenConfig working group"
  version: "0.1.1"
>
supported_models: <
  name: "openconfig-platform-linecard"
  organization: "OpenConfig working group"
  version: "0.1.1"
>
```

```
supported_models: <
  name: "openconfig-platform-port"
  organization: "OpenConfig working group"
  version: "0.3.2"
>
supported_models: <
  name: "openconfig-platform-psu"
  organization: "OpenConfig working group"
  version: "0.2.1"
>
supported_models: <
  name: "openconfig-platform-transceiver"
  organization: "OpenConfig working group"
  version: "0.7.0"
>
supported_models: <
  name: "openconfig-relay-agent"
  organization: "OpenConfig working group"
  version: "0.1.0"
>
supported_models: <
  name: "openconfig-routing-policy"
  organization: "OpenConfig working group"
  version: "2.0.1"
>
supported_models: <
  name: "openconfig-spanning-tree"
  organization: "OpenConfig working group"
  version: "0.2.0"
>
supported_models: <
  name: "openconfig-system"
  organization: "OpenConfig working group"
  version: "0.3.0"
>
supported_models: <
  name: "openconfig-telemetry"
  organization: "OpenConfig working group"
  version: "0.5.1"
>
supported_models: <
  name: "openconfig-vlan"
  organization: "OpenConfig working group"
  version: "3.0.2"
>
supported_models: <
  name: "DME"
  organization: "Cisco Systems, Inc."
>
supported_models: <
  name: "Cisco-NX-OS-Syslog-oper"
  organization: "Cisco Systems, Inc."
  version: "2019-08-15"
>
supported_encodings: JSON
supported_encodings: PROTO
gNMI_version: "0.8.0"

hostname user$
```

結果

Get について

Get RPC の目的は、クライアントがデバイスからデータ ツリーのスナップショットを取得できるようにすることです。1 つの要求で複数のパスを要求できます。gNMI パス規約に従って、XPATH の簡易形式である gNMI スキーマ パス エンコーディング規約がパスに使用されます。

Get 操作の詳細については、gNMI の仕様である gRPC ネットワーク管理インターフェイス（gNMI）の「状態情報のスナップショット取得」セクションを参照してください。

Get に関する注意事項と制限事項

次に、Get および Set に関する注意事項と制限事項を示します。

- GetRequest.encoding は JSON のみをサポートします。
- GetRequest.type の場合、DataType CONFIG と STATE のみが YANG で直接の相関関係と式を持ちます。OPERATIONAL はサポートされていません。
- 1 つの要求に OpenConfig（OC）YANG パスとデバイス YANG パスの両方を含めることはできません。要求には、OC YANG パスまたはデバイス YANG パスのみを含める必要があります。両方を含めることはできません。
- ルートパス（「/」：**すべて**のモデルのすべて）の GetRequest は許可されていません。
- gNMI Get はすべてのデフォルト値を返します（RFC 6243 [4] の report-all モードを参照）。
- Subscribe は、モデル `Cisco-NX-OS-syslog-oper` をサポートします。
- Get はモデル `Cisco-NX-OS-syslog-oper` をサポートしていません。
- `openconfig-procmon` データを取得するため、クエリをパス `/system/processes` または `/system` に送信できます。パス `/system` は、10.3.0 リリースより前のリリースではデータを取得しません。
- 次のオプション項目はサポートされていません。
  - パスのプレフィックス
  - パスのエイリアス
  - パス内のワイルドカード
- 1 つの GetRequest には最大 10 のパスを含めることができます。
- GetResponse で返される値フィールドのサイズが 12 MB を超える場合、システムはエラーステータス `grpc::RESOURCE_EXHAUSTED` を返します。

- 最大 gRPC 受信バッファサイズは 8 MB に設定されています。
- 大規模な構成がスイッチに適用されているときに Get 操作を実行すると、gRPC プロセスが使用可能なすべてのメモリを消費する可能性があります。メモリ枯渇状態が発生すると、次の syslog が生成されます。

  ```
  MTX-API: The memory usage is reaching the max memory resource limit (3072) MB
  ```

  この条件が複数回連続して発生すると、次の syslog が生成されます。

  ```
  The process has become unstable and the feature should be restarted.
  ```

  この時点で gRPC 機能を再起動して、gNMI トランザクションの通常の処理を続行することをお勧めします。
- Get の合計同時セッションの最大数は、構成されている最大同時呼び出しの 75% です。たとえば、MTX 同時呼び出しが 16 に構成されている場合、Get の合計同時セッションの最大数は 12 になります。
- Get と Set の同時セッションの合計数は、現在構成されている gNMI の同時最大数から 1 を引いたものです。たとえば、gnmi の同時呼び出しが 16 に構成されている場合、Get および Set の合計同時セッションの最大数は 15 になります。

設定

Set について

Set RPC は、デバイスの構成を変更するためにクライアントによって使用されます。デバイスデータに適用できる操作は削除、置換、更新で、順番を付けて行われます。単一の Set 要求のすべての操作はトランザクションとして扱われます。つまり、すべての操作が成功しなかった場合は、デバイスが元の状態にロールバックされます。Set 操作は、SetRequest で指定された順序で適用されます。パスが複数回指定されている場合、互いを上書きすることになったとしても、変更が適用されます。データの最終状態は、トランザクションの最終操作によって実現されます。SetRequest::delete、replace、update フィールドで指定されたすべてのパスはCONFIG データパスであり、クライアントによって書き込み可能であると想定されています。

Set 操作の詳細については、gNMI 仕様、
https://github.com/openconfig/reference/blob/1cf43d2146f9ba70abb7f04f6b0f6eaa504cef05/rpc/gnmi/gnmi-specification.md
の「Modifying State」のセクションを参照してください。

Set に関する注意事項と制限事項

次に、Set に関する注意事項と制限事項を示します。

- SetRequest.encoding は JSON のみをサポートします。

- 1 つの要求に OpenConfig（OC）YANG パスとデバイス YANG パスの両方を含めることはできません。要求には、OC YANG パスまたはデバイス YANG パスのみを含める必要があります。両方を含めることはできません。
- Subscribe は、モデル `Cisco-NX-OS-syslog-oper` をサポートします。
- 次のオプション項目はサポートされていません。
  - パスのプレフィックス
  - パスのエイリアス
  - パス内のワイルドカード
- 1 つの SetRequest には最大 20 のパスを含めることができます。
- 最大 gRPC 受信バッファサイズは 8 MB に設定されます。
- Get と Set の同時セッションの合計数は、現在構成されている gNMI の同時最大数から 1 を引いたものです。たとえば、gNMI の同時呼び出しが 16 に設定されている場合、Get および Set の合計同時セッションの最大数は 15 になります。
- Set::Delete RPC で、操作対象の構成が大きすぎる可能性がある場合、MTX ログ メッセージに警告が記録されます。

```
Configuration size for this namespace exceeds operational limit. Feature may
become unstable and require restart.
```

登録

Guidelines and Limitations for Subscribe

Following are the guidelines and limitations for Subscribe:

- If you configure a routing-policy **prefix-list** using the CLI and request gNMI Subscription for the routing-policy OpenConfig model, it is not supported. For example, when you attempt to subscribe an OpenConfig routing policy with a preexisting CLI configuration like the following, it returns empty values due to the current implementation of the OpenConfig model.

```
ip prefix-list bgp_v4_drop seq 5 deny 125.2.0.0/16 le 32
ipv6 prefix-list bgp_v6_drop seq 5 deny cafe:125:2::/48 le 128
Using the example paths,
openconfig-routing-policy:/routing-policy/defined-sets/prefix-sets/prefix-set[name=bgp_v4_drop]/config
openconfig-routing-policy:/routing-policy/defined-sets/prefix-sets/prefix-set[name=bgp_v6_drop]/config
```

- Beginning with Cisco NX-OS Release 9.3(3), Subscribe supports the OpenConfig model.
- For information about supported platforms, see the Nexus Switch Platform Matrix.
- The gNMI feature supports Subscribe and Capability RPCs.

- The feature supports JSON and gnmi.proto encoding. The feature does not support protobuf.any encoding.
- Each gNMI message has a maximum size of 12 MB. If the amount of collected data exceeds the 12 MB maximum, the collected data is dropped.

  You can avoid this situation by creating more focused subscriptions that handle smaller, more granular data-collection sets. So, instead of subscribing to one higher-level path, create multiple subscriptions for different, lower-level parts of the path.
- All paths within the same subscription request must have the same sample interval. If the same path requires different sample intervals, create multiple subscriptions.
- The feature does not support a path prefix in the Subscription request, but the Subscription can contain an empty prefix field.
- The feature supports Cisco DME and Device YANG data models.
- The gRPC process that supports gNMI uses the HIGH_PRIO cgroup, which limits the CPU usage to 75% of CPU and memory to 1.5 GB.
- The **show grpc gnmi** command has the following considerations:
  - The commands are not XMLized in this release.
  - The gRPC agent retains gNMI calls for a maximum of 1 hour after the call has ended.
  - If the total number of calls exceeds 2000, the gRPC agent purges ended calls based an internal cleanup routine.

The gRPC server runs in the management VRF. As a result, the gRPC process communicates only in this VRF forcing the management interface to support all gRPC calls.

gRPC functionality now includes the default VRF for a total of 2 gRPC servers on each Cisco Nexus 9000 switch. You can run one gRPC server in each VRF, or run only one gRPC server in the management VRF. Supporting a gRPC in the default VRF adds flexibility to offload processing gRPC calls from the management VRF, where significant traffic load might not be desirable.

If two gRPC servers are configured, be aware of the following:

- VRF boundaries are strictly enforced, so each gRPC server processes requests independent of the other, and 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.

gNMI ペイロード

gNMI は、特定のペイロード形式を使用して次のものにサブスクライブします：

- DME ストリーム
- YANG ストリーム

サブスクライブ操作は、次のモードでサポートされています：

- ONCE：データを 1 回サブスクライブして受信し、セッションを閉じます。
- POLL：サブスクライブしてセッションを開いたままにします。クライアントはデータが必要になるたびにポーリング要求を送信します。
- STREAM：特定の頻度でデータをサブスクライブし、受信します。ペイロードはナノ秒単位で値を受け入れます。1 秒 = 1000000000 ナノ秒です。
- ON_CHANGE：サブスクライブしてスナップショットを受信します。ツリーで何かが変更された場合にのみデータを受信します。
- TARGET_DEFINED：作成できるサブスクリプションの最適なタイプを決定します。

設定モード：

- 各モードには、内部サブと外部サブの 2 つの設定が必要です。
- ONCE：SAMPLE、ONCE
- POLL：SAMPLE、POLL
- STREAM：SAMPLE、STREAM
- ON_CHANGE：ON_CHANGE、STREAM
- TARGET_DEFINED：TARGET_DEFINED、STREAM

Origin

- DME：DME モデルへのサブスクライブ
- device：YANG モデルへのサブスクライブ
- openconfig：Openconfig モデルへのサブスクライブ

名前

- DME = DME モデルへのサブスクライブ
- Cisco-NX-OS-device = YANG モデルへのサブスクライブ

エンコーディング

- JSON = ストリームは JSON 形式で送信されます。
- PROTO = ストリームは protobuf.any 形式で送信されます。

DME ストリームの gNMI ペイロードの例

（注） クライアントごとに独自の入力形式があります。

```
{
    "SubscribeRequest":
    [
        {
            "_comment" : "ONCE request",
            "_delay" : 2,
            "subscribe":
            {
                "subscription":
                [
                    {
                        "_comment" : "1st subscription path",
                        "path":
                        {
                            "origin": "DME",
                            "elem":
                            [
                                {
                                    "name": "sys"
                                },
                                {
                                    "name": "bgp"
                                 }
                             ]
                        },
                        "mode": "SAMPLE"
                    }
                ],
                "mode": "ONCE",
                "allow_aggregation" : false,
                "use_models":
                [
                    {
                        "_comment" : "1st module",
                        "name": "DME",
                        "organization": "Cisco Systems, Inc.",
                        "version": "1.0.0"
                    }
                ],
                "encoding": "JSON"
            }
        }
    ]
}
```

gNMI ペイロード YANG ストリームの例

```
{
    "SubscribeRequest":
    [
        {
            "_comment" : "ONCE request",
            "_delay" : 2,
            "subscribe":
            {
                "subscription":
                [
                    {
                        "_comment" : "1st subscription path",
                        "path":
                        {
                            "origin": "device",
                            "elem":
```

```
                                [
                                    {
                                        "name": "System"
                                     },
                                     {
                                         "name": "bgp-items"
                                      }
                                ]
                            },
                                                        "mode": "SAMPLE"
                        }
                    ],
                    "mode": "ONCE",
                    "allow_aggregation" : false,
                    "use_models":
                    [
                        {
                            "_comment" : "1st module",
                            "name": "Cisco-NX-OS-device",
                            "organization": "Cisco Systems, Inc.",
                            "version": "0.0.0"
                        }
                    ],
                    "encoding": "JSON"
                }
            }
        ]
}
```

Openconfig ペイロードの例

```
{
    "SubscribeRequest":
    [
        {
            "_comment" : "STREAM request",
            "_delay"   : 2,
            "subscribe":
            {
                "subscription":
                [
                    {
                        "_comment" : "1st subscription path",
                        "path":
                        {
                            "origin": "openconfig",
                            "elem":
                            [
                                {
                                    "name": "interfaces"
                                }
                            ]
                        },
                        "mode": "SAMPLE",
                        "sample_interval": 10000000000
                    }
                ],
                "mode": "ONCE",
                "allow_aggregation" : false,
                "use_models":
                [
                    {
                        "_comment" : "1st module",
                        "name": "openconfig-interfaces",
```

```
                        "organization": "OpenConfig working group",
                        "version": "0.8.1"
                    }
                ],
                "encoding": "JSON"
            }
        }
    ]
}
```

ストリーミング Syslog

gNMI のストリーミング Syslog について

gNMI サブスクライブは、gNMI サブスクライブ要求に従って構造化データをプッシュすることで、システムで何が起こっているのかをリアルタイムで表示する、ネットワークをモニターする新しい方法です。

Cisco NX-OS リリース 9.3(3) 以降では、gNMI サブスクライブ機能の亜ポートが追加されました。

gNMI サブスクライブ サポートの詳細

- Syslog-oper モデルのストリーミング
  - stream_on_change

この機能は、8 GB 以上のメモリを搭載した Cisco Nexus 9000 シリーズ スイッチに適用されます。

ストリーミング Syslog に関する注意事項と制限事項：gNMI

ストリーミング Syslog に関する注意事項と制限事項は次のとおりです。

- 無効な syslog はサポートされていません。たとえば、フィルタまたはクエリ条件を持つ syslog です。
- 次のパスだけがサポートされます：
  - Cisco-NX-OS-Syslog-oper:syslog
  - Cisco-NX-OS-Syslog-oper:syslog/messages
- 次のモードはサポートされていません。
  - ストリーム サンプル
  - 投票
- 要求は YANG モデル フォーマットである必要があります。

- 内部アプリケーションを使用することも、独自のアプリケーションを作成することもできます。
- ペイロードはコントローラから送信され、gNMI は応答を送信します。
- エンコーディング フォーマットは JSON と PROTO です。

Syslog ネイティブ YANG モデル

YangModel はここにあります。

（注） タイムゾーン フィールドは、**clock format show-timezone syslog** が入力された場合にのみ設定されます。デフォルトでは設定されていないため、タイムゾーン フィールドは空です。

```
PYANG Tree for Syslog Native Yang Model:
>>> pyang -f tree Cisco-NX-OS-infra-syslog-oper.yang
module: Cisco-NX-OS-syslog-oper
+--ro syslog
+--ro messages
+--ro message* [message-id]
+--ro message-id int32
+--ro node-name? string
+--ro time-stamp? uint64
+--ro time-of-day? string
+--ro time-zone? string
+--ro category? string
+--ro group? string
+--ro message-name? string
+--ro severity? System-message-severity
+--ro text? string
```

サブスクライブ要求の例

次に、サブスクライブ要求の例を示します。

```
{
    "SubscribeRequest":
    [
        {
            "_comment" : "STREAM request",
            "_delay"   : 2,
            "subscribe":
            {
                "subscription":
                [
                    {
                        "_comment" : "1st subscription path",
                        "path":
                        {
                            "origin": "syslog-oper",
                            "elem":
                              [
                                {
                                    "name": "syslog"
```

```
                                    },
                                    {
                                        "name":"messages"
                                    }
                                ]
                            },
                            "mode": "ON_CHANGE"
                        }
                    ],
                    "mode": "ON_CHANGE",
                "allow_aggregation" : false,
                    "use_models":
                    [
                        {
                            "_comment" : "1st module",
                            "name": "Cisco-NX-OS-Syslog-oper",
                            "organization": "Cisco Systems, Inc.",
                            "version": "0.0.0"
                        }
                    ],
                    "encoding":"JSON"
                }
            }
        ]
}
```

PROTO 出力の例

これは PROTO 出力のサンプルです。

```
############################

[Subscribe]--------------------------------

### Reading from file ' /root/gnmi-console/testing_bl/stream_on_change/OC_SYSLOG.json '

Sat Aug 24 14:38:06 2019

### Generating request : 1 -----------

### Comment : STREAM request

### Delay : 2 sec(s) ...

### Delay : 2 sec(s) DONE

subscribe {

subscription {

path {

origin: "syslog-oper"

elem {

name: "syslog"

}

elem {
```

```
name: "messages"

}

}

mode: ON_CHANGE

}

use_models {

name: "Cisco-NX-OS-Syslog-oper"

organization: "Cisco Systems, Inc."

version: "0.0.0"

}

encoding: PROTO

}

Thu Nov 21 14:26:41 2019
Received response 3 --------------------------
update {
timestamp: 1574375201665688000
prefix {
origin: "Syslog-oper"
elem {
name: "syslog"
}
elem {
name: "messages"
}
}
update {
path {
elem {
name: "message-id"
}
}
val {
uint_val: 529
}
}
update {
path {
elem {
name: "node-name"
}
}
val {
string_val: "task-n9k-1"
}
}
update {
path {
elem {
name: "message-name"
}
}
val {
```

```
string_val: "VSHD_SYSLOG_CONFIG_I"
}
}
update {
path {
elem {
name: "text"
}
}
val {
string_val: "Configured from vty by admin on console0"
}
}
update {
path {
elem {
name: "group"
}
}
val {
string_val: "VSHD"
}
}
update {
path {
elem {
name: "category"
}
}
val {
string_val: "VSHD"
}
}
update {
path {
elem {
name: "time-of-day"
}
}
val {
string_val: "Nov 21 2019 14:26:40"
}
}
update {
path {
elem {
name: "time-zone"
}
}
val {
string_val: ""
}
}
update {
path {
elem {
name: "time-stamp"
}
}
val {
uint_val: 1574375200000
}
}
update {
```

```
path {
elem {
name: "severity"
}
}
val {
uint_val: 5
}
}
}

/Received -----------------------------------
```

-

JSON 出力の例

これは JSON 出力の例です。

```
[Subscribe]--------------------------------
### Reading from file ' testing_bl/stream_on_change/OC_SYSLOG.json '


Tue Nov 26 11:47:00 2019
### Generating request : 1 -----------
### Comment : STREAM request
### Delay : 2 sec(s) ...
### Delay : 2 sec(s) DONE
subscribe {
subscription {
path {
origin: "syslog-oper"
elem {
name: "syslog"
}
elem {
name: "messages"
}
}
mode: ON_CHANGE
}
use_models {
name: "Cisco-NX-OS-Syslog-oper"
organization: "Cisco Systems, Inc."
version: "0.0.0"
}
}

Tue Nov 26 11:47:15 2019
Received response 5 ---------------------------
update {
timestamp: 1574797636002053000
prefix {
}
update {
path {
origin: "Syslog-oper"
elem {
name: "syslog"
}
}
val {
json_val: "[ { \"messages\" : [[
```

```
{\"message-id\":657},{\"node-name\":\"task-n9k-1\",\"time-stamp\":\"1574797635000\",\"time-of-day\":\"Nov
 26 2019
11:47:15\",\"severity\":3,\"message-name\":\"HDR_L2LEN_ERR\",\"category\":\"ARP\",\"group\":\"ARP\",\"text\":\"arp
 [30318] Received packet with incorrect layer 2 address length (8 bytes), Normal pkt
with S/D MAC: 003a.7d21.d55e ffff.ffff.ffff eff_ifc mgmt0(9), log_ifc mgmt0(9), phy_ifc
 mgmt0(9)\",\"time-zone\":\"\"} ]] } ]"
}
}
}

/Received -----------------------------------
```

トラブルシューティング

TM トレース ログの収集

```
1. tmtrace.bin -f gnmi-logs gnmi-events gnmi-errors following are available
2. Usage:

bash-4.3# tmtrace.bin -d gnmi-events | tail -30 Gives the last 30
}
}
}
[06/21/19 15:58:38.969 PDT f8f 3133] [3981658944][tm_transport_internal.c:43] dn:
Cisco-NX-OS-device:System/cdp-items, sub_id: 0,
sub_id_str: 2329, dc_start_time: 0, length: 124, sync_response:1
[06/21/19 15:58:43.210 PDT f90 3133] [3621780288][tm_ec_yang_data_processor.c:93] TM_EC:
 [Y] Data received for 2799743488: 49
{
"cdp-items" : {
"inst-items" : {
"if-items" : {
"If-list" : [
{
"id" : "mgmt0",
"ifstats-items" : {
"v2Sent" : "74",
"validV2Rcvd" : "79"
}
}
]
}
}
}
}
[06/21/19 15:58:43.210 PDT f91 3133] [3981658944][tm_transport_internal.c:43] dn:
Cisco-NX-OS-device:System/cdp-items, sub_id: 0,
sub_id_str: 2329, dc_start_time: 0, length: 141, sync_response:1
[06/21/19 15:59:01.341 PDT f92 3133] [3981658944][tm_transport_internal.c:43] dn:
Cisco-NX-OS-device:System/intf-items, sub_id:
4091, sub_id_str: , dc_start_time: 1561157935518, length: 3063619, sync_response:0
[06/21/19 15:59:03.933 PDT f93 3133] [3981658944][tm_transport_internal.c:43] dn:
Cisco-NX-OS-device:System/cdp-items, sub_id:
4091, sub_id_str: , dc_start_time: 1561157940881, length: 6756, sync_response:0
[06/21/19 15:59:03.940 PDT f94 3133] [3981658944][tm_transport_internal.c:43] dn:
Cisco-NX-OS-device:System/lldp-items, sub_id:
4091, sub_id_str: , dc_start_time: 1561157940912, length: 8466, sync_response:1
bash-4.3#
```

MTX 内部ログの収集

```
1. Modify the following file with below /opt/mtx/conf/mtxlogger.cfg

<config name="nxos-device-mgmt">
  <container name="mgmtConf">
    <container name="logging">
      <leaf name="enabled" type="boolean" default="false">true</leaf>
      <leaf name="allActive" type="boolean" default="false">true<
/leaf>
      <container name="format">
        <leaf name="content" type="string" default="$DATETIME$
$COMPONENTID$ $TYPE$: $MSG$">$DATETIME$ $COMPONENTID$ $TYPE$
$SRCFILE$ @ $SRCLINE$ $FCNINFO$:$MSG$</leaf>
           <container name="componentID">
          <leaf name="enabled" type="boolean" default="true"></leaf>
           </container>
           <container name="dateTime">
          <leaf name="enabled" type="boolean" default="true"></leaf>
          <leaf name="format" type="string" default="%y%m%d.%H%M%S"><
/leaf>
            </container>
            <container name="fcn">
           <leaf name="enabled" type="boolean" default="true"></leaf>
           <leaf name="format" type="string"
default="$CLASS$::$FCNNAME$($ARGS$)@$LINE$"></leaf>
            </container>
      </container>
      <container name="facility">
          <leaf name="info" type="boolean" default="true">true</leaf>
          <leaf name="warning" type="boolean" default="true">true<
/leaf>
          <leaf name="error" type="boolean" default="true">true</leaf>

Note: Beginning with Cisco NX-OS Release 9.3(4), the following default configuration is
 changed from
      default true to false. To investigate an issue which requires the debug messages,
 edit
      the following configuration and toggle it to true.

          <leaf name="debug" type="boolean" default="false">true<
/leaf>
        </container>
        <container name="dest">
          <container name="console">
            <leaf name="enabled" type="boolean" default="false">true<
/leaf>
          </container>
          <container name="file">
         <leaf name="enabled" type="boolean" default="false">true<
/leaf>
    <leaf name="name" type="string" default="mtx-internal.log">
/leaf>


              <leaf name="location" type="string" default="./mtxlogs">
/volatile</leaf>
              <leaf name="mbytes-rollover" type="uint32" default="10"
>50</leaf>
              <leaf name="hours-rollover" type="uint32" default="24"
>24</leaf>
              <leaf name="startup-rollover" type="boolean" default="
false">true</leaf>
```

```
            <leaf name="max-rollover-files" type="uint32" default="10"
>10</leaf>
        </container>
      </container>
      <list name="logitems" key="id">
        <listitem>
                <leaf name="id" type="string">*</leaf>
                    <leaf name="active" type="boolean" default="false"
>false</leaf>
         </listitem>
          <listitem>
                 <leaf name="id" type="string">MTX-EvtMgr</leaf>
                     <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
        <listitem>
               <leaf name="id" type="string">TM-ADPT</leaf>
                   <leaf name="active" type="boolean" default="true"
>false</leaf>
        </listitem>
        <listitem>
              <leaf name="id" type="string">TM-ADPT-JSON</leaf>
                  <leaf name="active" type="boolean" default="true"
>false</leaf>
        </listitem  >
        <listitem>
                <leaf name="id" type="string">SYSTEM</leaf>
                    <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
        <listitem>
                <leaf name="id" type="string">LIBUTILS</leaf>
                     <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
        <listitem>
                <leaf name="id" type="string">MTX-API</leaf>
                    <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
         <listitem>
                 <leaf name="id" type="string">Model-*</leaf>
                     <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
        <listitem>
                <leaf name="id" type="string">Model-Cisco-NX-OS-
device</leaf>
                 <leaf name="active" type="boolean" default="true"
>false</leaf>
        </listitem>
        <listitem>
                 <leaf name="id" type="string">Model-openconfig-bgp<
/leaf>
                     <leaf name="active" type="boolean" default="true"
>false</leaf>
        </listitem>
        <listitem>
               <leaf name="id" type="string">INST-MTX-API</leaf>
                   <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
        <listitem>
                <leaf name="id" type="string">INST-ADAPTER-NC</leaf>
```

```
                    <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
        <listitem>
               <leaf name="id" type="string">INST-ADAPTER-RC</leaf>
                   <leaf name="active" type="boolean" default="true"
>true</leaf>
        </listitem>
        <listitem>
               <leaf name="id" type="string">INST-ADAPTER-GRPC</leaf>
                   <leaf name="active" type="boolean" default="true"
>true</leaf>
         </listitem>
      </list>
    </container>
  </container>
</config>

2. Run "no feature grpc" / "feature grpc"
3. The /volatile directory houses the mtx-internal.log, the log rolls over time so be
sure to grab what you need before then.

bash-4.3# cd /volatile/
bash-4.3# cd /volatile -al
total 148
drwxrwxrwx 4 root root 340 Jun 21 15:47 .
drwxrwxr-t 64 root network-admin 1600 Jun 21 14:45 ..
-rw-rw-rw- 1 root root 103412 Jun 21 16:14 grpc-internal-log
-rw-r--r-- 1 root root 24 Jun 21 14:44 mtx-internal-19-06-21-14-46-21.log
-rw-r--r-- 1 root root 24 Jun 21 14:46 mtx-internal-19-06-21-14-46-46.log
-rw-r--r-- 1 root root 175 Jun 21 15:11 mtx-internal-19-06-21-15-11-57.log
-rw-r--r-- 1 root root 175 Jun 21 15:12 mtx-internal-19-06-21-15-12-28.log
-rw-r--r-- 1 root root 175 Jun 21 15:13 mtx-internal-19-06-21-15-13-17.log
-rw-r--r-- 1 root root 175 Jun 21 15:13 mtx-internal-19-06-21-15-13-42.log
-rw-r--r-- 1 root root 24 Jun 21 15:13 mtx-internal-19-06-21-15-14-22.log
-rw-r--r-- 1 root root 24 Jun 21 15:14 mtx-internal-19-06-21-15-19-05.log
-rw-r--r-- 1 root root 24 Jun 21 15:19 mtx-internal-19-06-21-15-47-09.log
-rw-r--r-- 1 root root 24 Jun 21 15:47 mtx-internal.log
-rw-rw-rw- 1 root root 355 Jun 21 14:44 netconf-internal-log
-rw-rw-rw- 1 root root 0 Jun 21 14:45 nginx_logflag
drwxrwxrwx 3 root root 60 Jun 21 14:45 uwsgipy
drwxrwxrwx 2 root root 40 Jun 21 14:43 virtual-instance
bash-4.3#.
```

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。