IOS XRでのgNMIの設定とpYANGの実装

内容

はじめに

このドキュメントでは、Cisco IOS® XRでのgNMIの概要と、PYANGの使用方法およびモデルツリーの確認方法について説明します。

前提条件

要件

次の項目に関する知識があることが推奨されます。

- Cisco IOS XRプラットフォーム。
- pythonを使用します。
- ネットワーク管理プロトコル。

使用するコンポーネント

このドキュメントは、64ビットバージョン(eXR)に適用される特定のハードウェアバージョンに限定されるものではありません。

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

背景説明

gNMI定義

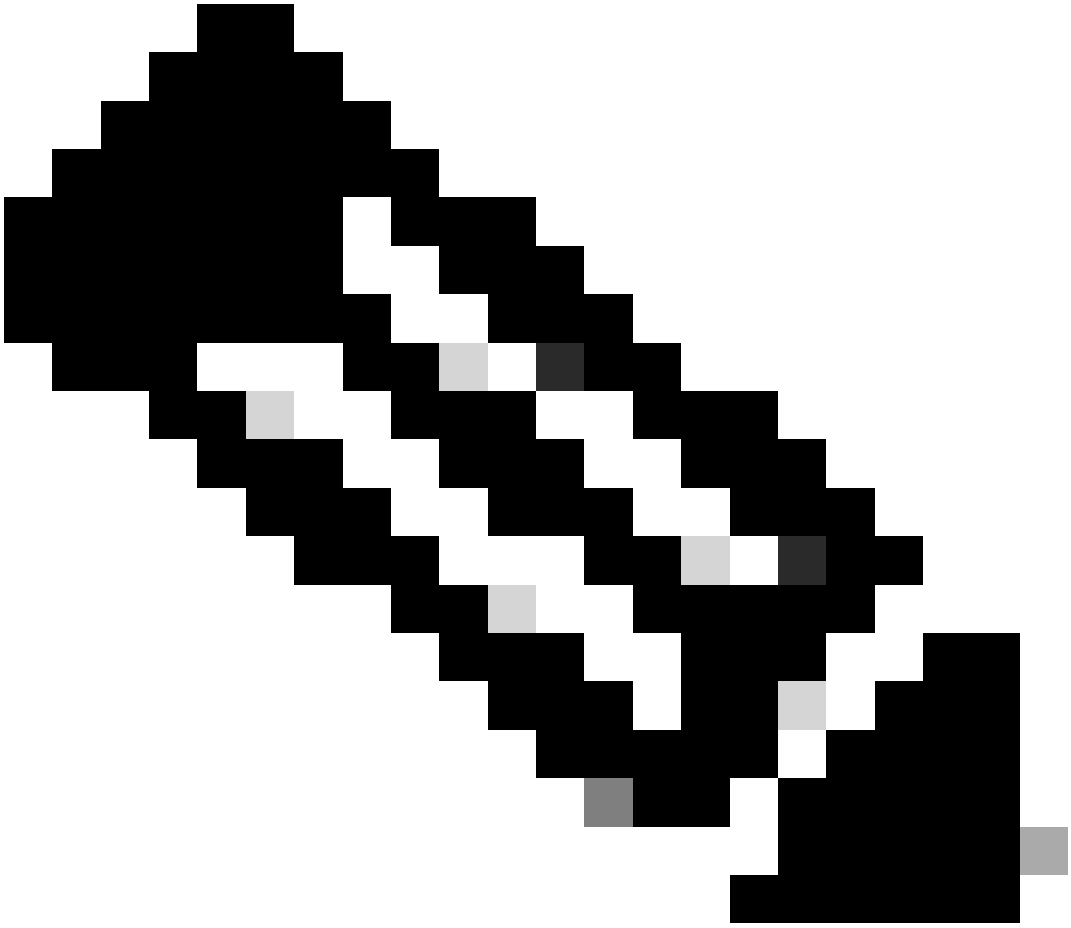
全体的に異なるネットワーク設定プロトコルがあり、NETCONF、RESTCONF、gNMI(Googleリモートプロシージャコール(gRPC)、gRPCネットワーク管理インターフェイス)などがあります。これらのモデルは、ネットワークデバイスを管理するための設定に使用され、常に機械的なプロセスの自動化を目的としています。

これらのプロトコルはさまざまなデータモデルを利用して、ネットワークデバイスが何を処理するか（つまり、情報を正規化する構造化情報、つまりスキーマ）、およびデバイス（この場合はルータ）によってどのように消費されるかをユーザが理解できるようにします。

gNMIはデータ処理を監視し、ネットワーク内のさまざまなデバイスを制御するためのRPC（リモートプロシージャコール）を提供します。

gNMIには4つの機能があります。

- 機能：gNMIは、ルータにインストールされているモデルをルータに問い合せます。これについては、このドキュメントで詳しく説明します。
- Get：データツリー内のすべてのリーフコンポーネントをルータに要求できます。この操作は要求された情報を要求します。
- セット：リーフは変数と見なされ、変更機能を提供します。セット操作は、データモデルの値を更新することを可能にします。
- 購読：テレメトリで使用されます。この機能は、モデル内の特定のモジュールからデータを取得するのに役立ちます。



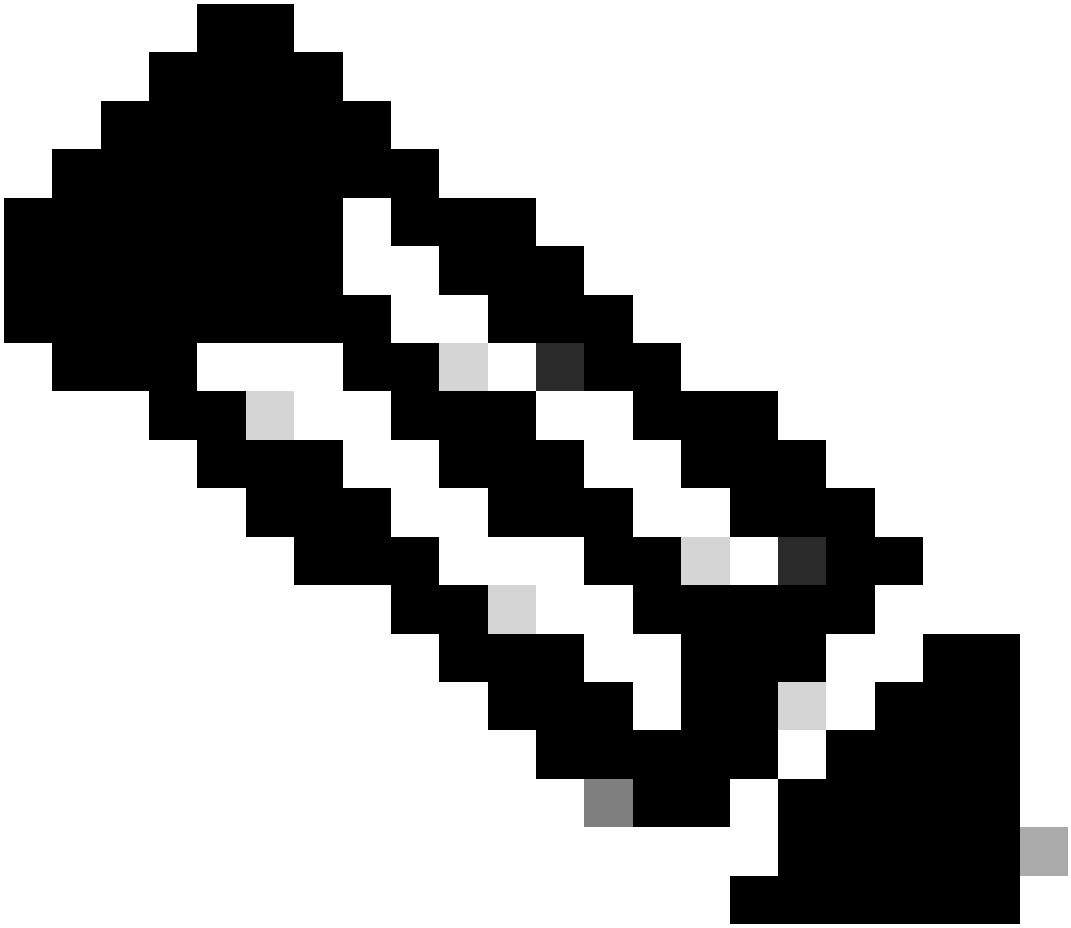
注：シスコはこのトピックに関して多くの情報を共有しています。gRPCの詳細については、次のリンクをクリックしてください。[xrdocsブログ – OpenConfig gNMI]

gNMI機能

ネットワーク管理プロトコル	NMI
トランスポート使用率	HTTP/2
サポート元	ベンダーニュートラル
符号化	プロトバフ

Proto Buffは、2つのデバイス間でデータのシリアル化とシリアル化を行う、言語に依存しない、

プラットフォームに依存しない方法です。この方法では、各リクエストに応答があります。

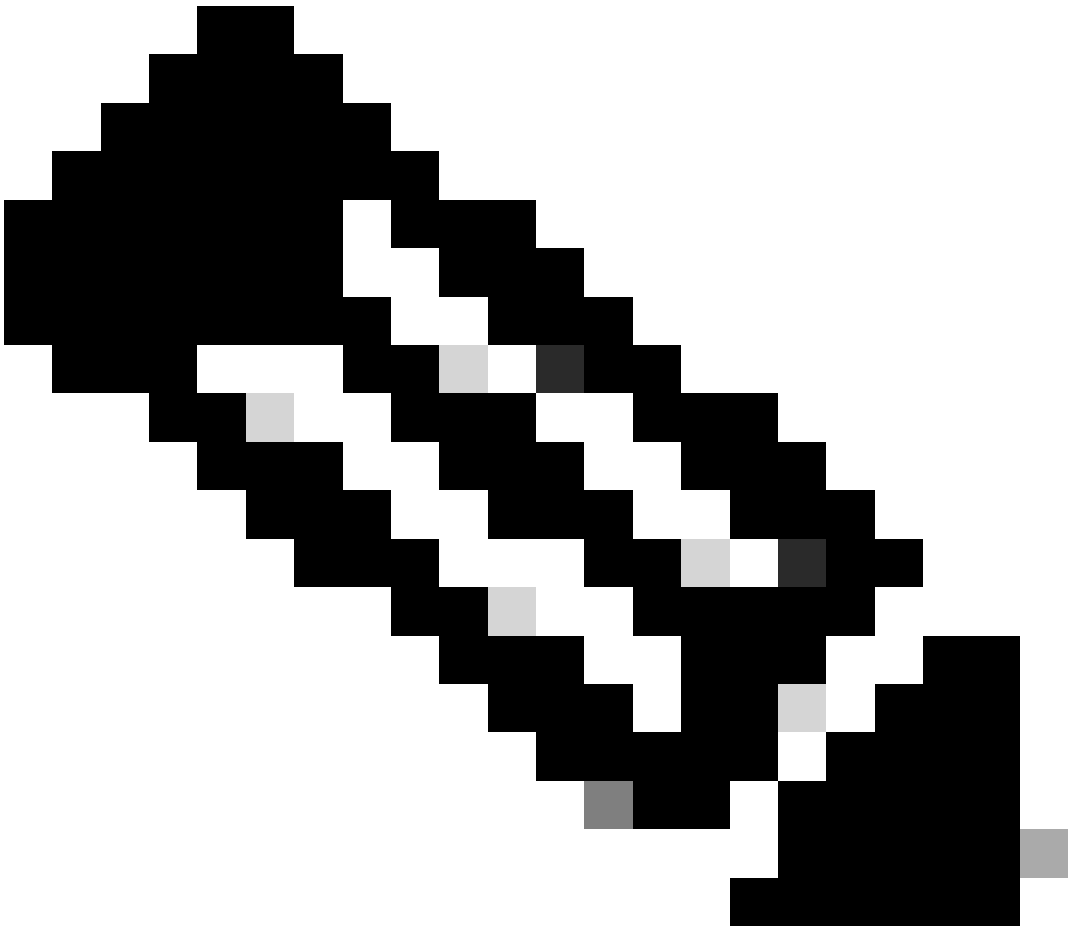


注:gRCPとProto Buffの詳細については、次のリンクをクリックしてください。 grpc Guide。

Cisco IOS XRでのgNMI基本設定

次に、ルータの基本設定を示します。

```
RP/0/RSP0/CPU0:XR(config)#grpc
RP/0/RSP0/CPU0:XR(config-grpc)#address-family ipv4
RP/0/RSP0/CPU0:XR(config-grpc)#max-request-total 256
RP/0/RSP0/CPU0:XR(config-grpc)#max-request-per-user 32

grpc
 address-family ipv4
 max-request-total 256
 max-request-per-user 32
```



注：ポートは、設定に基づいて設定できます。デフォルトでは、TLSを使用せずに設定できます。詳細については、57400をクリックしてください。github - grpc getting started

バリデータとしてのpYANG

pYANGは、Pythonで記述されたYANGバリデータです。このPython用ライブラリは、YANGモデルのチェックと、モデルを知る上で役立ちます。

ドキュメント(pYANGドキュメント)と同様に実行するには、コンピュータに仮想環境を作成することをお勧めします。

仮想環境でvenvドキュメントを実行する場合

以下を実行する必要があります。

```
python -m venv <name of the directory>
```

例（MacOS端末）：

```
% mkdir test
% cd test
% python3 -m venv virtual_env
% ls
virtual_env
```

pYANGをこの仮想環境cdにインストールしてディレクトリに貼り付け、次の手順を実行します。

```
% cd virual_env
% git clone https://github.com/mbj4668/pyang.git
% cd pyang
% pip install -e .
```

このデモンストレーションでは、python3 pipが使用されており、pip install -eを発行すると、venv:source <virtual environment directory>/bin/activate（MacOSの場合）がアクティブになります。

```
% source virtual_env/bin/activate
```

```
% python3 -m pip install pyang
Collecting pyang
  Downloading pyang-2.6.0-py2.py3-none-any.whl (594 kB)
     |████████████████████████████████| 594 kB 819 kB/s
Collecting lxml
  Downloading lxml-5.1.0-cp39-cp39-macosx_11_0_arm64.whl (4.5 MB)
     |████████████████████████████████| 4.5 MB 14.2 MB/s
Installing collected packages: lxml, pyang
Successfully installed lxml-5.1.0 pyang-2.6.0
```

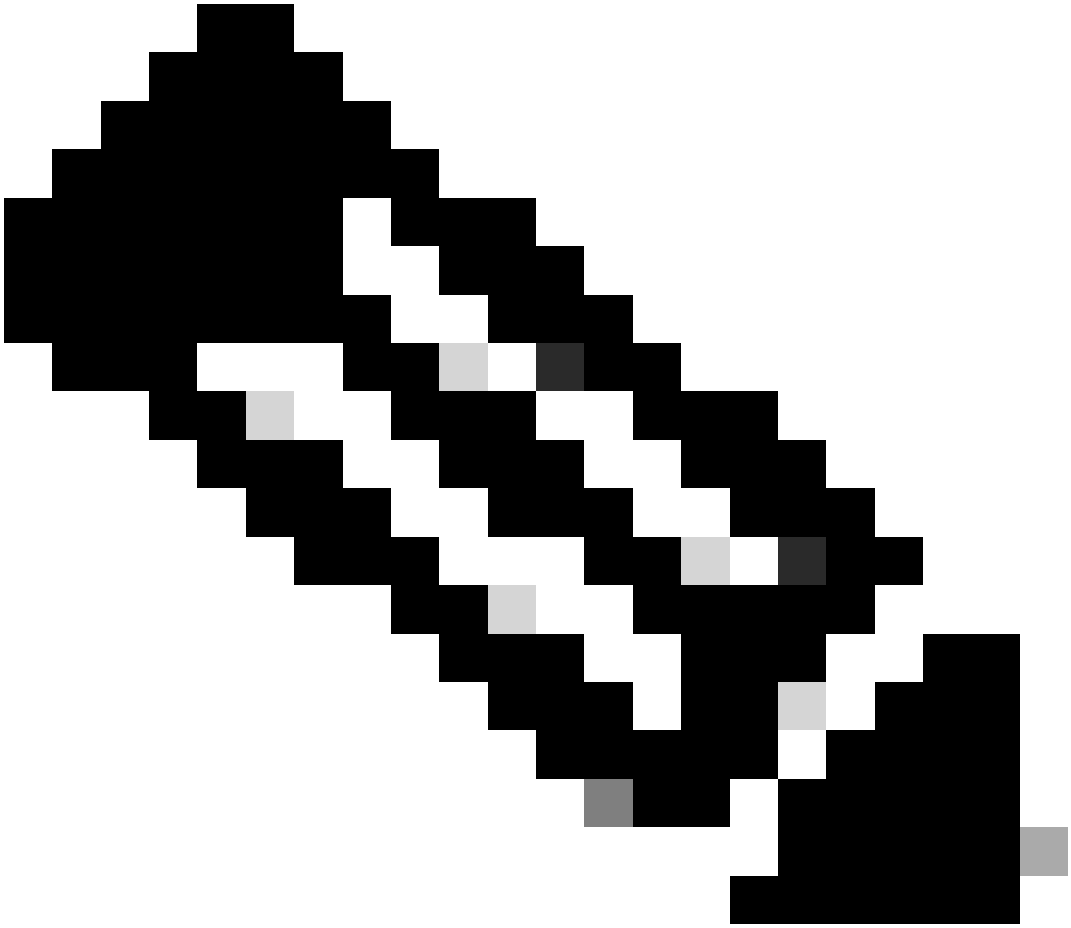
```
% pyang -h
Usage: pyang [options] [<filename>...]

Validates the YANG module in <filename> (or stdin), and all its dependencies.

Options:
  -h, --help            Show this help message and exit
  -v, --version         Show version number and exit
<snip>
```

pYANGがインストールされ、動作している状態で、モデルのダウンロードを続行します。

次のリンクでは、Cisco IOS XRで稼働するすべてのモデル(Cisco IOS XRモデル)を示します。

次のコードリンクを使用して、このモデルをvenvディレクトリにgit cloneすることをお勧めします。https://github.com/YangModels/yang.git

---

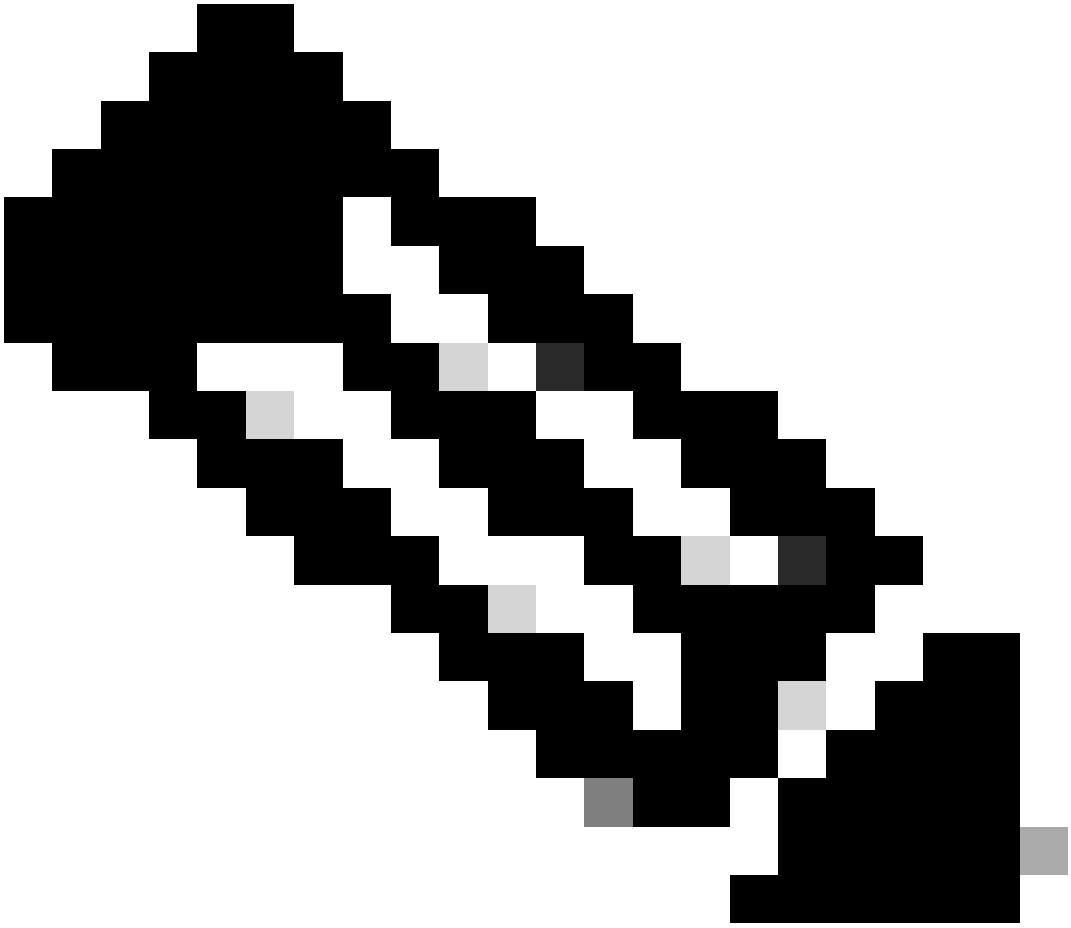


注：これは、仮想環境をアクティブ化した状態では実行されません。

```
% git clone https://github.com/YangModels/yang.git
Cloning into 'yang'...
remote: Enumerating objects: 54289, done.
remote: Counting objects: 100% (1910/1910), done.
remote: Compressing objects: 100% (323/323), done.
remote: Total 54289 (delta 1643), reused 1684 (delta 1586), pack-reused 52379
Receiving objects: 100% (54289/54289), 116.64 MiB | 8.98 MiB/s, done.
Resolving deltas: 100% (42908/42908), done.
Updating files: 100% (112197/112197), done.
```

仮想環境を再度アクティブ化し、次のクエリpyang -f tree yang/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yangをテストします。

```
(virtual_env) % pyang -f tree yang/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang
yang/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang:5: error: module "Cisco-IOS-XR-types" not found in
yang/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang:8: error: module "cisco-semver" not found in searc
module: Cisco-IOS-XR-ifmgr-cfg
  +--rw global-interface-configuration
  |  +--rw link-status?   Link-status-enum
  +--rw interface-configurations
     +--rw interface-configuration* [active interface-name]
        +--rw dampening
        |  +--rw args?                 enumeration
        |  +--rw half-life?            uint32
        |  +--rw reuse-threshold?      uint32
        |  +--rw suppress-threshold?   uint32
        |  +--rw suppress-time?        uint32
        |  +--rw restart-penalty?      uint32
        +--rw mtus
        |  +--rw mtu* [owner]
        |     +--rw owner    xr:Cisco-ios-xr-string
        |     +--rw mtu      uint32
        +--rw encapsulation
        |  +--rw encapsulation?         string
        |  +--rw capsulation-options?   uint32
        +--rw shutdown?                      empty
        +--rw interface-virtual?             empty
        +--rw secondary-admin-state?         Secondary-admin-state-enum
        +--rw interface-mode-non-physical?   Interface-mode-enum
        +--rw bandwidth?                     uint32
        +--rw link-status?                   empty
        +--rw description?                   string
        +--rw active                         Interface-active
        +--rw interface-name                 xr:Interface-name
```



注：リーフのデータ形式はStringやuint32のようになっていますが、ルートではこの情報は表示されません。GETやSETなどの操作は、これらの値の取得/更新専用です。

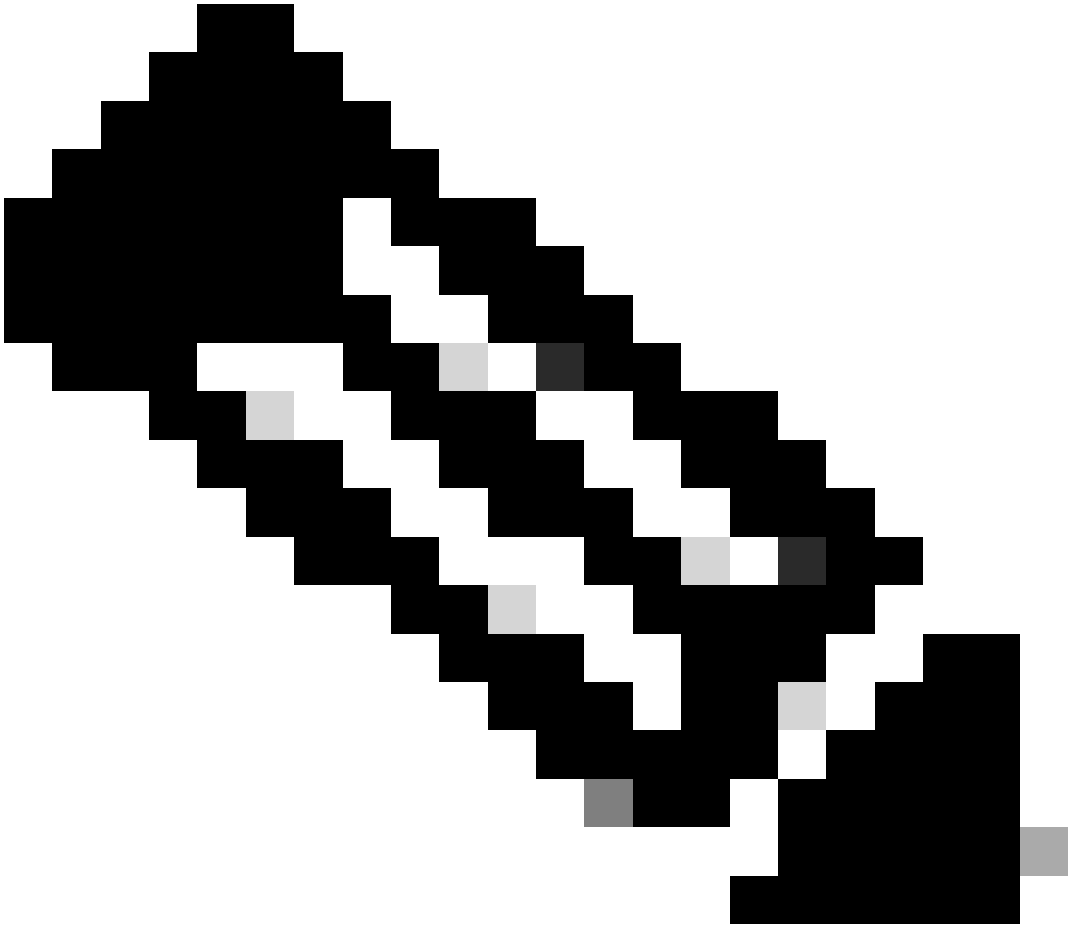
また、ほとんどのモデルでは拡張機能を使用して完全な設定が行われている点にも注意してください。CLI出力には、基本的なインターフェイス管理設定が含まれており、IPv4を表示する必要がある場合には、次のコマンドを使用します。

```
% pyang -f tree yan2/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang  yan2/vendor/cisco/xr/711/Cisco-IO
module: Cisco-IOS-XR-ifmgr-cfg
  +--rw global-interface-configuration
  |  +--rw link-status?   Link-status-enum
  +--rw interface-configurations
     +--rw interface-configuration* [active interface-name]
        +--rw dampening
        |  +--rw args?                 enumeration
        |  +--rw half-life?            uint32
        |  +--rw reuse-threshold?      uint32
        |  +--rw suppress-threshold?   uint32
        |  +--rw suppress-time?        uint32
```

```
|  +--rw restart-penalty?      uint32
+--rw mtus
|  +--rw mtu* [owner]
|     +--rw owner    xr:Cisco-ios-xr-string
|     +--rw mtu      uint32
+--rw encapsulation
|  +--rw encapsulation?         string
|  +--rw capsulation-options?   uint32
+--rw shutdown?                                empty
+--rw interface-virtual?                       empty
+--rw secondary-admin-state?                   Secondary-admin-state-enum
+--rw interface-mode-non-physical?             Interface-mode-enum
+--rw bandwidth?                               uint32
+--rw link-status?                             empty
+--rw description?                             string
+--rw active                                   Interface-active
+--rw interface-name                           xr:Interface-name
+--rw ipv4-io-cfg:ipv4-network
|  +--rw ipv4-io-cfg:bgp-pa
|  |  +--rw ipv4-io-cfg:input
|  |  |  +--rw ipv4-io-cfg:source-accounting?        boolean
|  |  |  +--rw ipv4-io-cfg:destination-accounting?   boolean
|  |  +--rw ipv4-io-cfg:output
|  |     +--rw ipv4-io-cfg:source-accounting?        boolean
|  |     +--rw ipv4-io-cfg:destination-accounting?   boolean
|  +--rw ipv4-io-cfg:verify
|  |  +--rw ipv4-io-cfg:reachable?      Ipv4-reachable
|  |  +--rw ipv4-io-cfg:self-ping?      Ipv4-self-ping
|  |  +--rw ipv4-io-cfg:default-ping?   Ipv4-default-ping
|  +--rw ipv4-io-cfg:bgp
|  |  +--rw ipv4-io-cfg:qppb
|  |  |  +--rw ipv4-io-cfg:input
|  |  |     +--rw ipv4-io-cfg:source?        Ipv4-interface-qppb
|  |  |     +--rw ipv4-io-cfg:destination?   Ipv4-interface-qppb
|  |  +--rw ipv4-io-cfg:flow-tag
|  |     +--rw ipv4-io-cfg:flow-tag-input
|  |        +--rw ipv4-io-cfg:source?        boolean
|  |        +--rw ipv4-io-cfg:destination?   boolean
|  +--rw ipv4-io-cfg:addresses
|  |  +--rw ipv4-io-cfg:secondaries
|  |  |  +--rw ipv4-io-cfg:secondary* [address]
|  |  |     +--rw ipv4-io-cfg:address      inet:ipv4-address-no-zone
|  |  |     +--rw ipv4-io-cfg:netmask      inet:ipv4-address-no-zone
|  |  |     +--rw ipv4-io-cfg:route-tag?   uint32
|  |  +--rw ipv4-io-cfg:primary!
|  |  |  +--rw ipv4-io-cfg:address      inet:ipv4-address-no-zone
|  |  |  +--rw ipv4-io-cfg:netmask      inet:ipv4-address-no-zone
|  |  |  +--rw ipv4-io-cfg:route-tag?   uint32
|  |  +--rw ipv4-io-cfg:unnumbered?    xr:Interface-name
|  |  +--rw ipv4-io-cfg:dhcp?          empty
|  +--rw ipv4-io-cfg:helper-addresses
|  |  +--rw ipv4-io-cfg:helper-address* [address vrf-name]
|  |     +--rw ipv4-io-cfg:address     inet:ipv4-address-no-zone
|  |     +--rw ipv4-io-cfg:vrf-name    xr:Cisco-ios-xr-string
|  +--rw ipv4-io-cfg:forwarding-enable?       empty
|  +--rw ipv4-io-cfg:icmp-mask-reply?         empty
|  +--rw ipv4-io-cfg:tcp-mss-adjust-enable?   empty
|  +--rw ipv4-io-cfg:ttl-propagate-disable?   empty
|  +--rw ipv4-io-cfg:point-to-point?          empty
|  +--rw ipv4-io-cfg:mtu?                     uint32
+--rw ipv4-io-cfg:ipv4-network-forwarding
   +--rw ipv4-io-cfg:directed-broadcast?   empty
```

```
+--rw ipv4-io-cfg:unreachables?          empty
+--rw ipv4-io-cfg:redirects?             empty
```

このクエリでは、Cisco-IOS-XR-ifmgr-cfg.yangとCisco-IOS-XR-ipv4-io-cfg.yangの2つのモデルが使用され、IPv4アドレスがリーフとして表示されます。



注:「yang/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang:5: error: module "Cisco-IOS-XR-types" not found in search path」のようなエラーが表示される場合は、コマンドに—path=を追加してください。

これを実行してオンにすると、すべてのユーザがgNMI操作で情報を要求し、日付を変更できます。たとえば、次のリンクをクリックします。プログラマビリティ設定ガイド

ユーザがシンプルなAPIを実行したい場合は、grpccのようなツールがあります。

このAPIはNPMを介してインストールされ、プログラマビリティ設定ガイドのリンクで使用されるツールです。このリンクでは、ユーザがクエリと応答をテストするためのより多くの例を共有

します。

トラブルシューティング：

gNMIの場合、エントリを収集する前にクエリを確認する必要があります。ほとんどのAPIは次のようになります。

- gnmicの略
- grpccの略
- gRPC

すべて、ルータが生成したエラーを表示します。

例：

```
"cisco-grpc:errors": {
"error": [
{
"error-type": "application",
"error-tag": "operation-failed",
"error-severity": "error",
"error-message": "'YANG framework' detected the 'fatal' condition 'Operation failed'"
}
]
}
}
```

または

```
"error": [
   {
    "error-type": "application",
    "error-tag": "operation-failed",
    "error-severity": "error",
    "error-path": <path>,
"error-message": "'sysdb' detected the 'warning' condition 'A verifier or EDM callback function returne
   }
  ]
```

これらはプラットフォームに依存するエラーであり、ルータに沿ってチェックする必要があります。クエリのコマンドをCLI経由でルータでも発行できることを確認することを推奨します。

このタイプのエラー、またはCisco IOS XRプラットフォームに関連するその他のエラーについては、次の情報をTACと共有します。

- 使用されるクエリと操作：

```
{
 "Cisco-IOS-XR-ifmgr-cfg:interface-configurations":
  { "interface-configuration": [
    {
      "active": "act",
      "interface-name": "Loopback0",
      "description": "LOCAL TERMINATION ADDRESS",
      "interface-virtual": [
        null
       ],
       "Cisco-IOS-XR-ipv4-io-cfg:ipv4-network": {
        "addresses": {
            "primary": {
              "address": "172.16.255.1",
              "netmask": "255.255.255.255"
       }
      }
     }
    }
   ]
 }
}
```

- 表示されるエラー (上記の任意) 。
- 次のコマンドを発行します。

```
show grpc trace all
```

クエリを何度かテストし、「show grpc trace all」コマンドを繰り返します。

エラーはバリアント型ですが、問題を生成する可能性のあるコンポーネントも表示されます。

例：

- 「'sysdb' detected the 'warning' condition 'A verifier or EDEDM callback function returned: 'not found'」：このエラーは、ルータでこのプロセス用のshow techコマンドを収集するために必要なsysdbについて説明しています。

  次の例は、エラーを示すshow tech for sysdbプロセスを示しています。

```
show tech-support sysdb
```

  この出力では、エラーによってコンポーネントとエラーが表示されています。表示されているエラーに関連する可能性のあるshow tech-supportの情報をすべて収集してください。

- 「'YANG framework' detected the 'fatal' condition 'Operation failed'：このエラーでは、ルータでのプロセスは表示されません。つまり、モデルでクエリが失敗しているということです。この情報をTACに提供し、失敗している可能性のあるプロセスを確認してください。

この情報を収集したら、次のコマンドセットも追加します。

XR VMで次の手順を実行します。

show tech-support tctcpsr（隠しコマンド）

show tech-support grpcc（可能な場合）

show tech-support gspコマンド

show tech-support

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。