# IOS XR에서 gNMI 구성 및 pYANG 구현

## 목차



## 소개

이 문서에서는 Cisco IOS® XR의 gNMI에 대한 간략한 설명과 PYANG 사용 방법 및 모델 트리 확인에 대해 설명합니다.

## 사전 요구 사항

### 요구 사항

다음 주제에 대한 지식을 보유하고 있으면 유용합니다.

- Cisco IOS XR 플랫폼
- 파이썬
- 네트워크 관리 프로토콜.

### 사용되는 구성 요소

이 문서는 64비트 버전(eXR)에 적용되는 특정 하드웨어 버전으로 제한되지 않습니다.

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다. 현재 네트워크가 작동 중인 경우 모든 명령의 잠재적인 영향을 미리 숙지하시기 바랍니다.

## 배경 정보

### gNMI 정의

전반적으로 NETCONF, RESTCONF, gNMI(Google Remote Procedure Calls (gRPC), gRPC

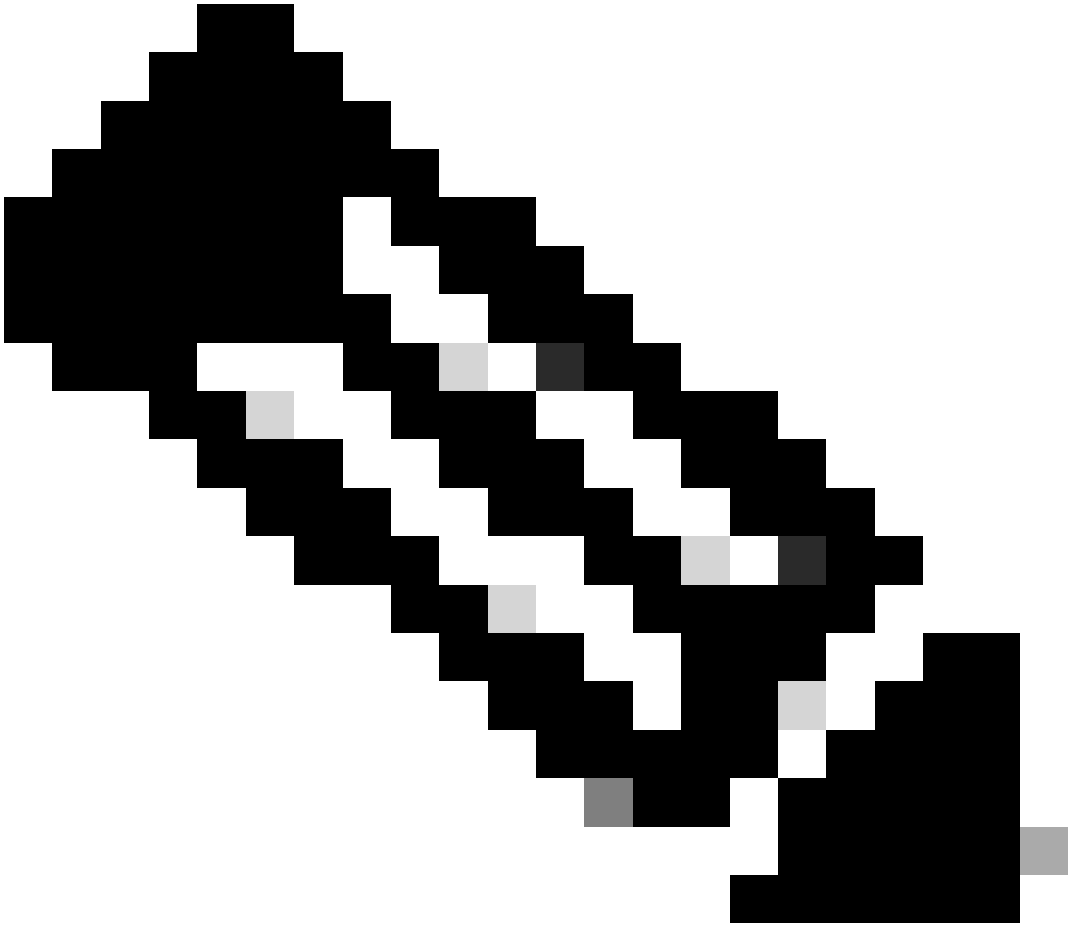
Network Management Interface) 등과 같은 다양한 네트워크 컨피그레이션 프로토콜이 있습니다. 이러한 모델은 네트워크 장치를 관리하기 위해 구성하는 데 사용되며, 항상 정비사가 될 수 있는 프로세스를 자동화하는 것을 목표로 합니다.

이러한 프로토콜은 서로 다른 데이터 모델을 사용하여 사용자가 네트워크 디바이스 프로세스가 어떤 것인지, 즉 정보를 표준화하고 디바이스(이 경우 라우터)에서 어떻게 사용하는지 이해할 수 있도록 합니다.

gNMI는 데이터 전달을 감독하고 RPC(Remote Procedure Calls)를 제공하여 네트워크의 여러 디바이스를 제어합니다.

gNMI에는 네 가지 기능이 있습니다.

- 기능: gNMI는 라우터에 설치된 모델을 라우터에 요청합니다. 이에 대해서는 이 문서에서 자세히 설명합니다.
- Get: 데이터 트리의 모든 리프 구성 요소를 라우터에 요청할 수 있습니다. 이 작업은 요청된 정보를 요청합니다.
- 설정: Leaf는 변수로 간주되며 변경 기능을 제공하는 요소이며, 설정 작업은 사용자가 데이터 모델의 값을 업데이트할 수 있도록 지원합니다.
- 구독: 텔레메트리(Telemetry)에서 활용된 이 기능은 모델의 특정 모듈에서 데이터를 가져오는 데 도움이 됩니다.



참고: Cisco는 이 주제에 대해 많은 정보를 공유했습니다. gRPC에서 추가 정보를 보려면 다음 링크를 클릭합니다. xrdocs [blog - OpenConfig gNMI]

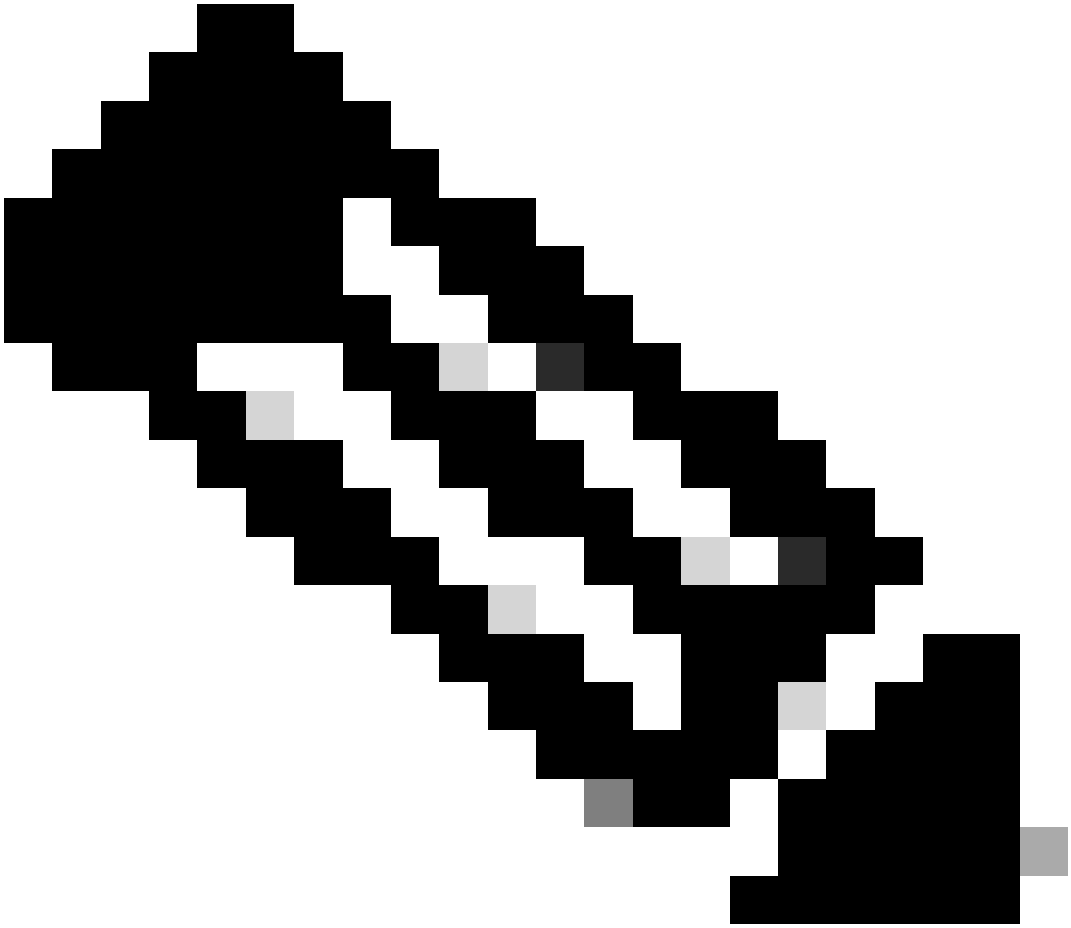
## gNMI 기능

| | |
|---|---|
| 네트워크 관리 프로토콜 | gNMI |
| 전송 사용 | HTTP/2 |
| 지원자 | 중립적 공급업체 |
| 인코딩 | Proto Buff |

Proto Buff는 두 디바이스 간의 데이터를 역직렬화하고 직렬화하는 언어 중립적 플랫폼 중립적 방법

입니다. 각 요청에는 Reply가 있습니다.

---



참고: gRCP 및 Proto Buff에 대한 자세한 내용을 보려면 다음 링크인 grpc Guide를 클릭하십시오.

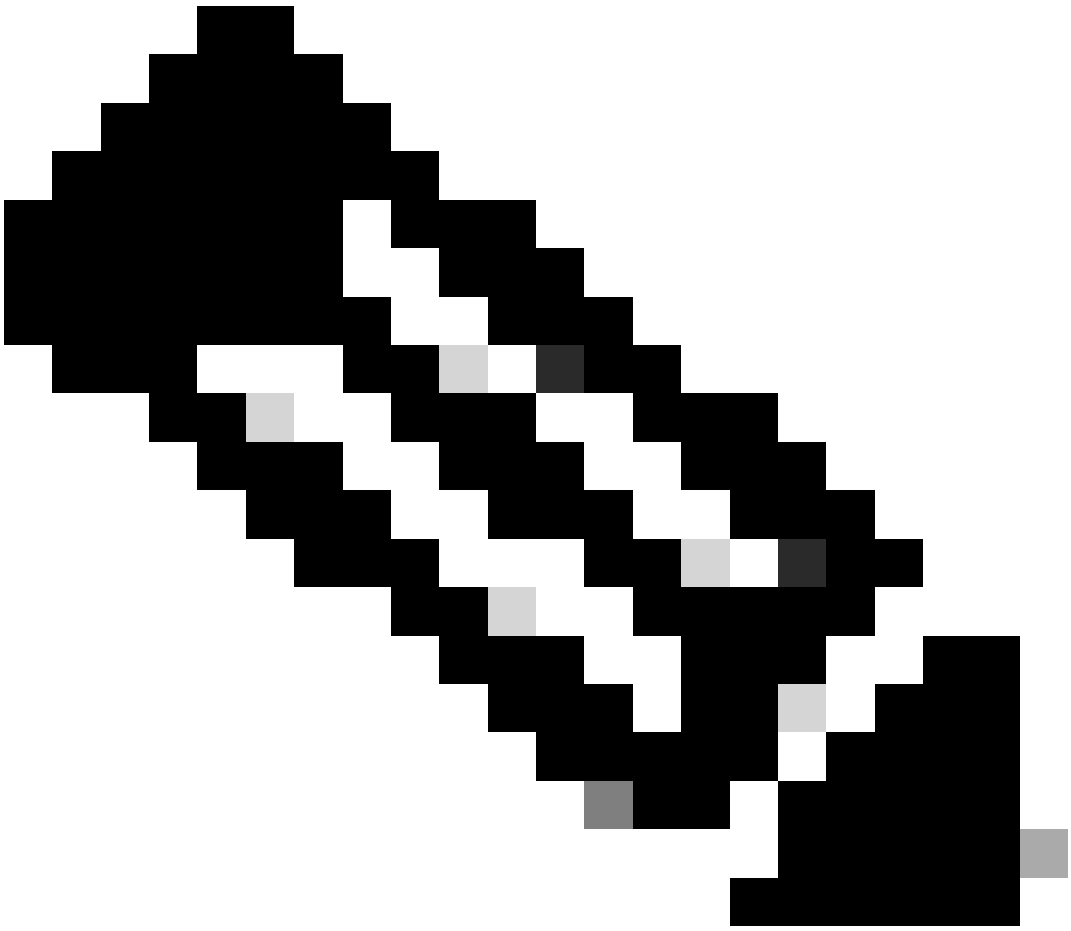
---

## Cisco IOS XR의 gNMI 기본 컨피그레이션

다음은 라우터의 기본 컨피그레이션입니다.

```
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
```



참고: TLS를 사용하지 않고 기본값인 설정에 따라 포트를 구성할 수 있습니다. 자세한 내용을 보려면 github - grpc getting started를 57400.

## pYANG을 검증자로 사용

pYANG은 python으로 작성된 YANG 유효성 검사기입니다. 이 python 라이브러리는 YANG 모델을 확인하는 데 도움이 되며, 또한 해당 모델을 알고 있습니다.

이 작업을 설명서(pYANG 설명서)에서 실행하는 것처럼 컴퓨터에 가상 환경을 만드는 것이 좋습니다.

가상 환경에서 공급업체 설명서를 실행하기 위한

다음을 실행해야 합니다.

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

예(MacOS 터미널에서):

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

이 가상 환경 cd에서 pYANG을 디렉토리에 설치하고 다음 명령을 붙여넣습니다.

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

이 데모에서는 python3 pip를 사용했으며 pip 설치 -e가 실행되면 venv를 활성화합니다. source <virtual environment directory>/bin/activate(MacOS용).

```
% 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
```

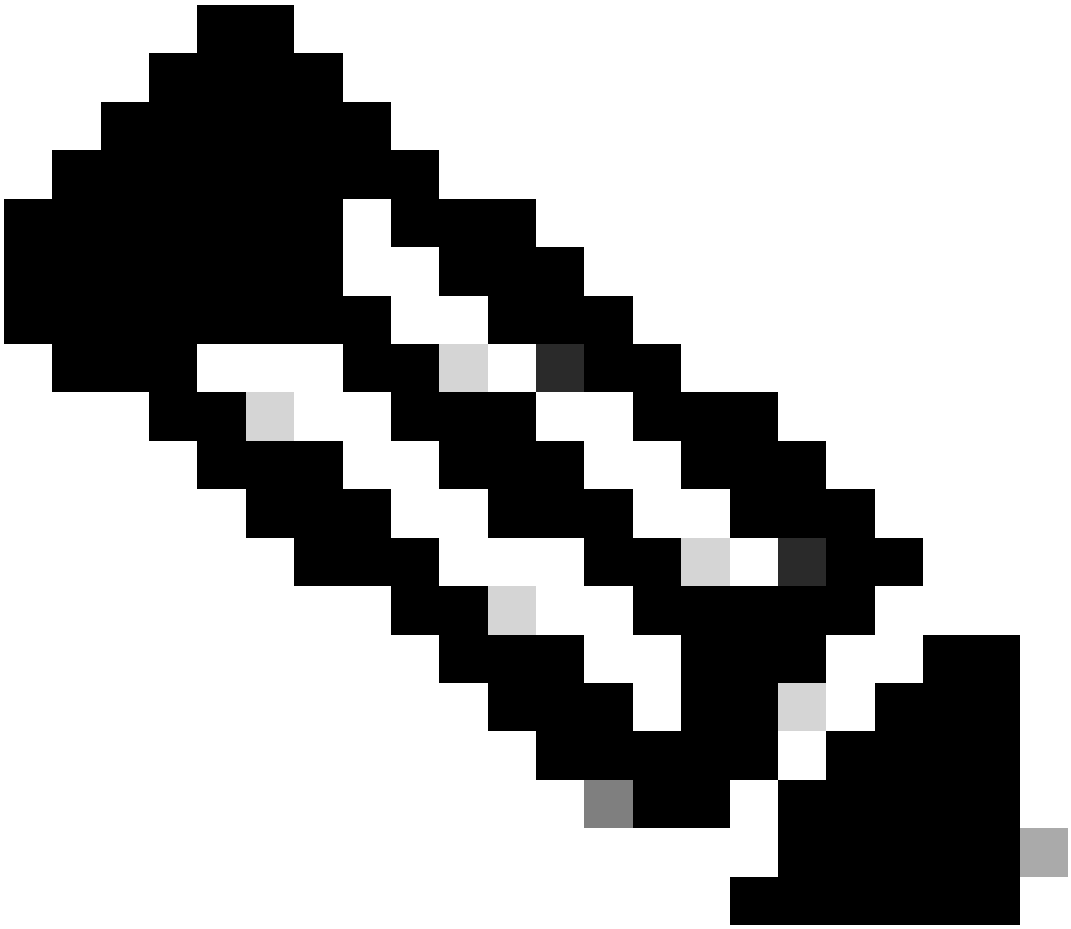
```
% pyang -h
Usage: pyang [options] [<filename>...]

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>
```

pYANG이 설치되어 작동하면 모델 다운로드를 계속 진행합니다.

다음 링크에는 Cisco IOS XR에서 실행하는 모든 모델, 즉 Cisco IOS XR 모델이 있습니다.

다음 코드 링크를 사용하여 venv 디렉터리에서 이 모델을 복제하는 것이 좋습니다.
https://github.com/YangModels/yang.git

---



참고: 이 작업은 가상 환경이 활성화된 상태에서 수행되지 않습니다.

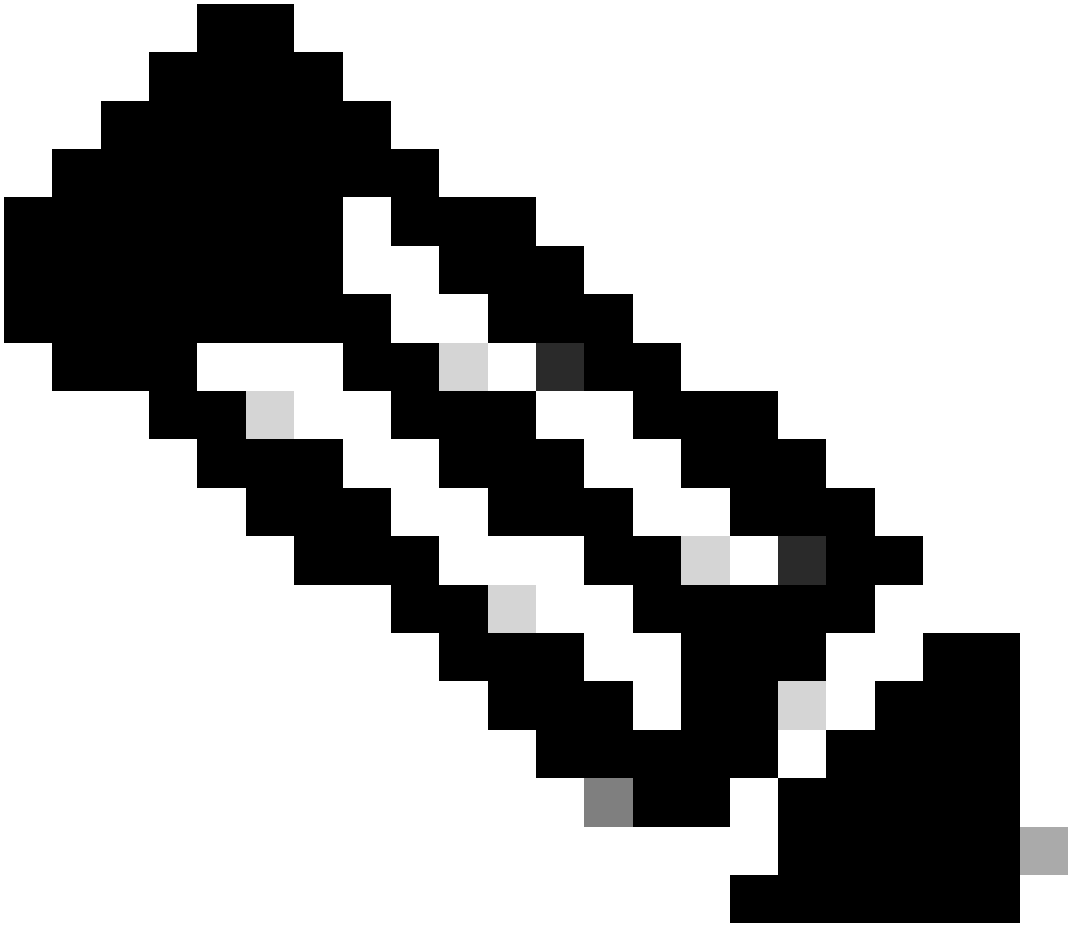
---

```
% 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.
```

가상 환경을 다시 활성화하고 다음 쿼리를 테스트합니다. pyang -f tree

yang/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang.

```
(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
```



참고: leaf에 String, uint32 등의 데이터 형식이 있지만 루트는 이 정보를 표시하지 않습니다. GET 및 SET와 같은 작업은 이러한 값을 끌어오거나 업데이트하는 데 사용됩니다.

---

또 다른 참고 사항은 대부분의 모델에서 전체 컨피그레이션을 사용하려면 추가 기능이 필요하며, CLI 출력에는 기본 인터페이스 관리 컨피그레이션이 있으며, IPv4를 표시해야 하는 경우 다음을 사용합니다.

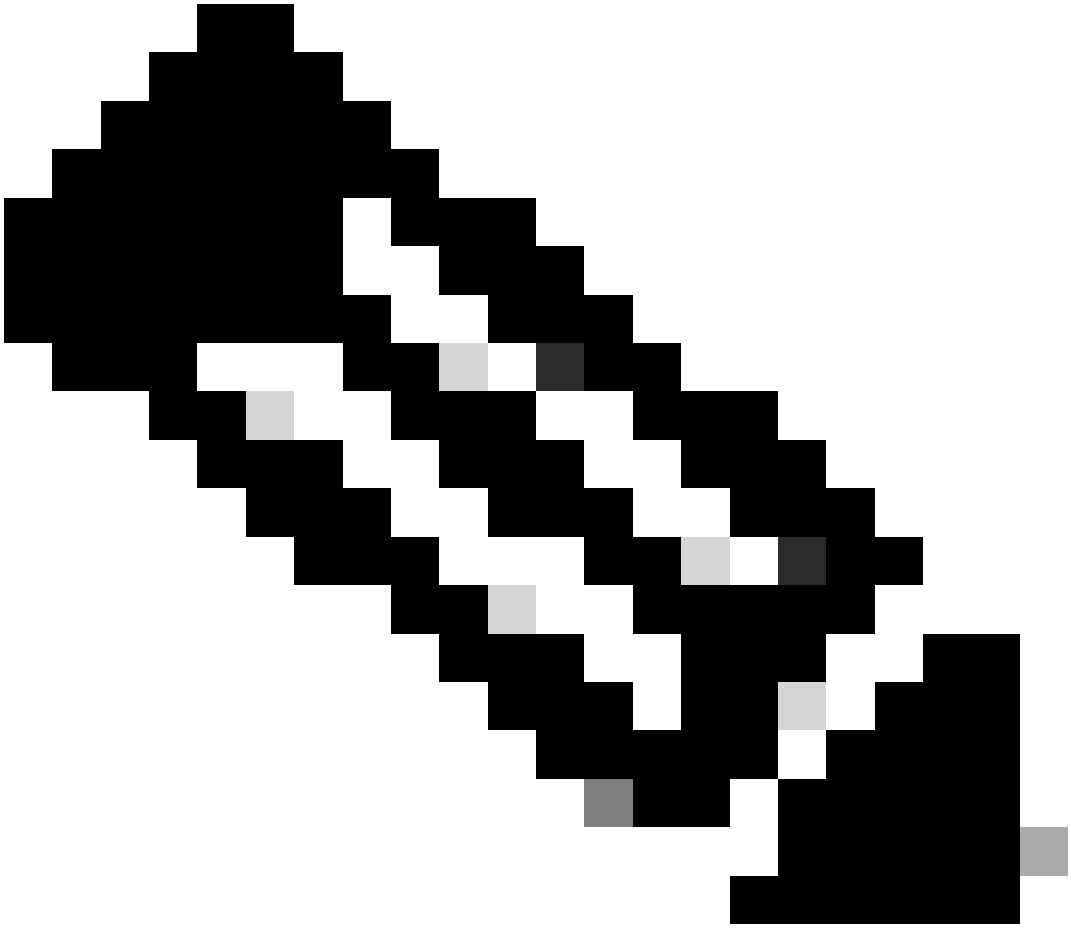
```
% pyang -f tree yan2/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang  yan2/vendor/cisco/xr/711/Cisco-IO
module: Cisco-IOS-XR-ifmgr-cfg
  +--rw global-interface-configuration
  |  +--rw link-status?   Link-status-enum
  +--rw interface-configurations
     +--rw interface-configuration* [active interface-name]
        +--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
```

이 쿼리에서는 Cisco-IOS-XR-ifmgr-cfg.yang 및 Cisco-IOS-XR-ipv4-io-cfg.yang의 두 가지 모델이 사용되는데, 이제 IPv4 주소가 leaf로 표시됩니다.

---



> 참고: "yang/vendor/cisco/xr/711/Cisco-IOS-XR-ifmgr-cfg.yang:5: error: module "Cisco-IOS-XR-types" not found in search path"와 같은 오류가 표시되는 경우 명령에 —path=를 추가합니다.

---

이 작업을 완료하고 선택하면 모든 사용자가 gNMI 작업 및 변경 날짜와 함께 정보를 요청할 수 있습니다. 예를 들어 다음 링크를 클릭합니다. Programmability Configuration Guide

사용자가 간단한 API를 실행하려는 경우 grpcc와 같은 도구가 있습니다.

이 API는 NPM을 통해 설치되며 Programmability Configuration Guide 링크에서 사용되는 도구입니다. 이 링크는 사용자가 쿼리 및 응답을 테스트할 수 있는 더 많은 예를 공유합니다.

문제 해결:

gNMI의 경우 다음과 같은 대부분의 API를 수집하기 전에 쿼리를 확인해야 합니다.

- 모나
- 독단
- gRPC

모두, 라우터가 생성한 오류를 표시합니다.

예를 들면 다음과 같습니다.

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

또는

```
"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
   }
  ]
```

이는 라우터를 따라 확인해야 하는 플랫폼 종속 오류입니다. 쿼리의 명령도 CLI를 통해 라우터에서 실행할 수 있는지 확인하는 것이 좋습니다.

이러한 유형의 오류 또는 Cisco IOS XR 플랫폼과 관련된 다른 오류의 경우 다음 정보를 TAC에 공유합니다.

- 사용된 쿼리 및 작업:

```
    {
     "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"
           }
          }
         }
        }
       ]
     }
    }
```

- 표시되는 오류(위에 표시된 항목).
- 다음 명령을 실행합니다.

```
show grpc trace all
```

쿼리를 두 번 테스트하고 "show grpc trace all" 명령을 반복합니다.

오류는 변형되지만 문제를 일으킬 수 있는 구성 요소도 표시됩니다.

예를 들면 다음과 같습니다.

- "'sysdb'가 'warning' 조건 '검증자 또는 EDEDM 콜백 함수가 반환되었습니다: 'not found'": 이 오류는 라우터에서 이 프로세스에 대한 show tech 명령을 수집하는 데 필요한 sysdb를 설명합니다.

  다음 예에서는 sysdb 프로세스용 show tech에서 오류를 보여줍니다.

```
show tech-support sysdb
```

  이 출력에서 오류는 구성 요소를 표시하고, 오류는 표시되는 오류와 관련될 수 있는 show tech-support를 수집합니다.

- "'YANG framework'에서 '치명적인' 조건 '작업 실패'를 감지했습니다.": 이 오류는 라우터에 프로세스를 표시하지 않습니다. 즉, 모델에서 쿼리가 실패했음을 나타냅니다. 이 정보를 TAC에 공유하여 실패 가능한 항목을 검토하십시오.

이 정보가 수집되면 다음 명령 집합도 추가합니다.

XR VM에서

show tech-support tctcpsr

show tech-support grpcc

show tech-support gsp

show tech-support

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.