Data Models for Network Automation

YANG communication protocols

A YANG communication protocol is a set of rules that

- establishes standardized exchanges of information between network devices,
- enables reliable and secure communication between clients and routers, and
- facilitates automation and programmable operations across different platforms.

In YANG-based systems, communication protocols connect the router and the client. These protocols enable clients to consume YANG data models, automating and programming network operations.

Supported protocols

YANG uses one of these protocols:

- Network Configuration Protocol (NETCONF)
- RPC framework (gRPC) by Google

Note gRPC is supported only on 64-bit platforms.

The transport and encoding mechanisms for these two protocols are shown in the table:

Table 1: Protocols and their supported transport and encoding or decoding mechanisms

Protocol	Transport	Encoding or Decoding
NETCONF	SSH	XML

Protocol	Transport	Encoding or Decoding
gRPC	HTTP/2	JSON

NETCONF protocols

A NETCONF protocol is a network management protocol family that

- enables installation, modification, and deletion of device configuration data,
- uses XML-based data encoding for both configuration and protocol messages, and
- supports client-server communication through a simple RPC (Remote Procedure Call) mechanism.

NETCONF protocols provide standard tools for automating the management of network devices. Clients can programmatically update or retrieve configuration and operational data by exchanging XML-formatted messages with servers.

A NETCONF client issues an RPC call to configure a new interface, and the NETCONF server responds with an XML message confirming the change.

gRPC protocols

A gRPC protocol is a network communication framework that

- is based on Protocol Buffers (Protobuf), an open-source binary serialization format,
- enables flexible, efficient, and automated serialization of structured data for remote procedure calls (RPCs), and
- requires you to define message types using `.proto` files, where each message contains a series of name-value pairs for clear data structure.

Additional reference information: gRPC is open source and offers a lightweight alternative to traditional protocols like XML for transmitting data. Its design aims for both simplicity and high performance in exchanging data between systems.

Note gRPC is supported only on 64-bit platforms.

Examples:

- A network management system can use gRPC protocols to issue NETCONF RPCs and configure device features using data models.
- Developers use `.proto` files to define the messages and service methods that gRPC clients and servers exchange.
- REST APIs typically use JSON payloads and HTTP methods and do not rely on Protocol Buffers or `.proto` definitions as gRPC does.

Access data models

You can access the data models using one of these options:

Access data models from router

To access data models directly from the router, you can use these steps:

Procedure

Step 1 Enter the global configuration mode.

Example:

```
Router#configure
```

Step 2 Configure the NETCONF network management protocol to remotely configure and manage the router using YANG data models.

Example:

```
Router(config)#netconf-yang agent ssh
```

Step 3 Commit the configuration.

Example:

```
Router(config)#commit
```

Step 4 Establish a NETCONF session with the device and retrieve the capabilities information.

Example:

```
Router#show netconf-yang capabilities
Tue Sep 19 22:03:26.305 UTC
[Netconf capabilities]

   D: Has deviations

 Capability                                                                                | Revision |D
-------------------------------------------------------------------------------------------+----------+-
urn:ietf:params:netconf:base:1.1                                                           | -        |
urn:ietf:params:netconf:capability:candidate:1.0                                           | -        |
urn:ietf:params:netconf:capability:confirmed-commit:1.1                                    | -        |
urn:ietf:params:netconf:capability:interleave:1.0                                          | -        |
urn:ietf:params:netconf:capability:notification:1.0                                        | -        |
urn:ietf:params:netconf:capability:rollback-on-error:1.0                                   | -        |
urn:ietf:params:netconf:capability:validate:1.1                                            | -        |
http://cisco.com/ns/yang/Cisco-IOS-XR-8000-fib-platform-cfg                                |2019-04-05|
http://cisco.com/ns/yang/Cisco-IOS-XR-8000-lpts-oper                                       |2022-05-05|
http://cisco.com/ns/yang/Cisco-IOS-XR-8000-platforms-npu-resources-oper                    |2020-10-07|
http://cisco.com/ns/yang/Cisco-IOS-XR-8000-qos-oper                                        |2021-06-28|
http://cisco.com/ns/yang/Cisco-IOS-XR-Ethernet-SPAN-act                                    |2021-03-22|
http://cisco.com/ns/yang/Cisco-IOS-XR-Ethernet-SPAN-cfg                                    |2022-07-13|
http://cisco.com/ns/yang/Cisco-IOS-XR-Ethernet-SPAN-datatypes                              |2021-10-06|
http://cisco.com/ns/yang/Cisco-IOS-XR-Ethernet-SPAN-oper                                   |2022-09-05|
http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-aaacore-cfg                                      |2019-04-05|
http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-ldapd-cfg                                        |2022-06-22|
http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-ldapd-oper                                       |2022-05-20|
http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-lib-cfg                                          |2020-10-22|
http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-lib-datatypes
---------------------------------   Truncated for brevity  ---------------------------------------
```

By examining the capabilities, you can view the available data models for the software version installed on the router.

Access data models from Cisco Feature Navigator

To access data models from Cisco Feature Navigator, you can use these steps:

Procedure

Step 1 Go to Cisco Feature Navigator.

Step 2 If you have a Cisco.com account, click on the **Login** button and enter your credentials. If you don't have an account, you can click **Continue as Guest**.

You will be directed to the Cisco Feature Navigator main page.

Step 3 Click **YANG Data Models**.

Step 4 Select the **Product** and **Cisco IOS XR Release** based on your requirement.

The data models are listed based on type—Cisco XR native models, Unified models and OpenConfig models.

You can use the search field to search for specific data model of interest.

Step 5 Click the specific data model of interest to view more details.

The data model is displayed in a hierarchical tree structure making it easier to navigate and understand the relationships between different YANG modules, containers, leaves and leaf lists. You can apply filters to further narrow down the data model definitions for the selected platform and release based on status such as deprecated, obsolete and unsupported nodes.

You can also click the **Download** icon to export the data model information in Excel format.

This visual tree form helps you get insights into the nodes that you can use to automate your network.

The data models on Cisco Feature Navigator is regularly updated based on IOS XR release. If you encounter any problem or have suggestions for improvements, share your experience using Send us your feedback link.

Access data models from GitHub

To access the data models from GitHub repository, you can use these steps:

Procedure

Step 1 Go to the GitHub repository for data models.

On the repository page, you will find a list of folders based on IOS XR releases.

Step 2 Navigate to the release folder of interest to view the list of supported data models and their definitions. For example, if you want to access the data models for IOS XR release 7.10.1, click on the folder named `7.10.1`.

Inside the folder, you will find a list of YANG files representing different data models.

Step 3 Click on the YANG file you want to access to view its contents.

You can also click on the **Raw** button to see the raw code or use the **Download** button to download the file to your computer.

Each data model defines a complete and cohesive model, or augments an existing data model with additional XPaths. To view a comprehensive list of the data models supported in a release, navigate to the **Available-Content.md** file in the repository. The unsupported sensor paths are documented as deviations. For example, `openconfig-acl.yang` provides details about the supported sensor paths, whereas `cisco-xr-openconfig-acl-deviations.yang` shows the unsupported sensor paths for `openconfig-acl.yang` model.

Step 4 Repeat the above steps for other versions or data models of interest.

The GitHub repository for IOS XR data models is regularly updated based on release. You can also contribute to the repository by submitting pull requests, opening issues if you encounter any problems or have suggestions for improvements.

CLI to Yang mapping tools

CLI to Yang mapping tools are software utilities that

- bridge the gap between traditional CLI commands and modern programmatic approaches,
- associate CLI commands with YANG data models, and
- streamline network management tasks by providing clear mappings.

Note Starting from Release 7.11.1 , the command **yang-describe** in the Command Line Interface (CLI) is deprecated.

CLI commands are widely used for configuring and extracting the operational details of a router. But bulk configuration changes through CLIs are cumbersome and error-prone. These limitations restrict automation and scale. To overcome these limitations, you need an automated mechanism to manage your network. Cisco IOS XR supports a programmatic way of configuring and collecting operational data of a router using Yang data models. However, owing to the large number of CLI commands, it is cumbersome to determine the mapping between the CLI command and its associated data model.

The CLI to Yang describer tool is a component in the IOS XR software. It helps in mapping the CLI command with its equivalent data models. With this tool, network automation using data models can be adapted with ease.

The tool simulates the CLI command and displays the following data:

- Yang model mapping to the CLI command
- List of the associated sensor paths

To retrieve the Yang equivalent of a CLI, use the following command:

```
Router#yang-describe ?
            configuration  Describe configuration commands(cisco-support)
            operational    Describe operational commands(cisco-support)
```

The tool supports description of both operational and configurational commands.

Configuration data

In the following example, the Yang paths for configuring the MPLS label range with minimum and maximum static values are displayed:

```
Router#yang-describe configuration mpls label range table 0 34000 749999 static 34000 99999

            YANG Paths:
            Cisco-IOS-XR-um-mpls-lsd-cfg:mpls/label/range/table-0
            Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range

Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/minvalue

Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/max-value

Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/min-static-value


Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/max-static-value
```

In the following example, the Yang paths for configuring the gRPC address are displayed:

```
Router#yang-describe configuration grpc address-family ipv4
            YANG Paths:
            Cisco-IOS-XR-man-ems-cfg:grpc/enable
            Cisco-IOS-XR-man-ems-cfg:grpc/address-family
```

Operational data

The operational data includes support for the `show` CLI commands.

The example shows the Yang paths to retrieve the operational data for MPLS interfaces:

```
Router#yang-describe operational show mpls interfaces
            Mon May 10 12:34:05.198 UTC
            YANG Paths:
            Cisco-IOS-XR-mpls-lsd-oper:mpls-lsd/interfaces/interface
```

The example shows the Yang paths to retrieve the operational data for Virtual Router Redundancy Protocol (VRRP):

```
Router#yang-describe operational show vrrp brief
                Mon May 10 12:34:38.041 UTC
                YANG Paths:
                Cisco-IOS-XR-ipv4-vrrp-oper:vrrp/ipv4/virtual-routers/virtual-router
                Cisco-IOS-XR-ipv4-vrrp-oper:vrrp/ipv6/virtual-routers/virtual-router
```

CLI commands are widely used for configuring and extracting the operational details of a router. But bulk configuration changes through CLIs are cumbersome and error-prone. These limitations restrict automation and scale. To overcome these limitations, you need an automated mechanism to manage your network.

CLI to Yang mapping tool

The CLI to Yang describer tool is a component in the IOS XR software. It helps in mapping the CLI command with its equivalent data models. With this tool, network automation using data models can be adapted with ease.

Tool functionality

The tool simulates the CLI command and displays these data:

- Yang model mapping to the CLI command
- List of the associated sensor paths

Retrieve Yang equivalent of CLI

This example shows how to retrieve the Yang equivalent of a CLI:

```
Router#yang-describe ?
               configuration  Describe configuration commands(cisco-support)
               operational    Describe operational commands(cisco-support)
```

Example: Configuration data

In this example, the Yang paths for configuring the MPLS label range with minimum and maximum static values are displayed:

```
                  Router#
                yang-describe configuration mpls label range table 0 34000 749999 static
 34000 99999
Mon May 10 12:37:27.192 UTC
YANG Paths:
  Cisco-IOS-XR-um-mpls-lsd-cfg:mpls/label/range/table-0
  Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range
  Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/minvalue
  Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/max-value

Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/min-static-value


Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/max-static-value
```

Configuration data example for gRPC

In this example, the Yang paths for configuring the gRPC address are displayed:

```
Router#yang-describe configuration mpls label range table 0 34000 749999 static 34000 99999

               Mon May 10 12:37:27.192 UTC
               YANG Paths:
               Cisco-IOS-XR-um-mpls-lsd-cfg:mpls/label/range/table-0
             Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range


Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/minvalue

Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/max-value

Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/min-static-value


Cisco-IOS-XR-mpls-lsd-cfg:mpls-lsd/label-databases/label-database/label-range/max-static-value
```

YANG action

A YANG action is a network management operation that

- is defined in a YANG model using an RPC statement
- enables execution of specific commands or operations (such as ping or reload) on a network device via standardized management protocols like NETCONF or gRPC, and
- provides structured responses indicating the outcome of the requested operation.

YANG actions allow remote management systems to execute device-specific operations programmatically. Each action is modeled as an RPC (Remote Procedure Call) statement within a YANG module, making it accessible through automation tools and network controllers that communicate over NETCONF (using XML) or gRPC (using JSON). When an action request is received, the device executes the operation and returns a protocol-specific response. For example, common actions include "ping," "traceroute," "copy," and process restart commands.

Supported YANG actions and models

The following table lists common YANG actions and the associated YANG models. For a complete list of supported actions, refer to the YANG Data Models Navigator.

Table 2: YANG actions and models

Actions	YANG Models
logmsg	Cisco-IOS-XR-syslog-act
snmp	Cisco-IOS-XR-snmp-test-trap-act
rollback	Cisco-IOS-XR-cfgmgr-rollback-act
clear isis	Cisco-IOS-XR-isis-act
clear bgp	Cisco-IOS-XR-ipv4-bgp-act

PING NETCONF action

This example shows the IOS XR NETCONF action request to run the ping command on the router.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
 <ping xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act">
  <destination>
   <destination>1.2.3.4</destination>
  </destination>
 </ping>
</rpc>
```

This section shows the NETCONF action response from the router.

```
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
 <ping-response xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act">
  <ipv4>
```

```
   <destination>1.2.3.4</destination>
   <repeat-count>5</repeat-count>
   <data-size>100</data-size>
   <timeout>2</timeout>
   <pattern>0xabcd</pattern>
   <rotate-pattern>0</rotate-pattern>
   <reply-list>
    <result>!</result>
    <result>!</result>
    <result>!</result>
    <result>!</result>
    <result>!</result>
   </reply-list>
   <hits>5</hits>
   <total>5</total>
   <success-rate>100</success-rate>
   <rtt-min>1</rtt-min>
   <rtt-avg>1</rtt-avg>
   <rtt-max>1</rtt-max>
  </ipv4>
 </ping-response>
</rpc-reply>
```

XR process restart action

This example shows the process restart action sent to NETCONF agent.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <sysmgr-process-restart xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-sysmgr-act">
     <process-name>processmgr</process-name>
     <location>0/RP0/CPU0</location>
  </sysmgr-process-restart>
</rpc>
```

This example shows the action response received from the NETCONF agent.

```
<?xml version="1.0"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <ok/>
</rpc-reply>
```

Copy action

This example shows the RPC request and response for `copy` action:

RPC request:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <copy xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-shellutil-copy-act">
    <sourcename>//root:<location>/100MB.txt</sourcename>
    <destinationname>/</destinationname>
    <sourcefilesystem>ftp:</sourcefilesystem>
    <destinationfilesystem>harddisk:</destinationfilesystem>
    <destinationlocation>0/RSP1/CPU0</destinationlocation>
  </copy>
</rpc>
```

RPC response:

```
<?xml version="1.0"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
 <response xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-shellutil-copy-act">Successfully
completed copy operation</response>
</rpc-reply>

8.261830565s elapsed
```

Delete action

This example shows the RPC request and response for `delete` action:

RPC request:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
<delete xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-shellutil-delete-act">
    <name>harddisk:/netconf.txt</name>
  </delete>
</rpc>
```

RPC response:

```
<?xml version="1.0"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
 <response xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-shellutil-delete-act">Successfully
 completed delete operation</response>
</rpc-reply>

395.099948ms elapsed
```

Supported protocols and operational restrictions

- NETCONF actions use XML formatting; gRPC actions use JSON.
- Some actions—such as installing software—may have restrictions (for example, deprecated commands or maximum parameter limits).
- System admin models support only <get> and <get-config> operations; <edit-config> supports only merge. Delete, remove, and replace are not supported for system admin models.

YANG actions and data elements

A YANG action acts like a remote control button: when pressed (invoked through a protocol), the device performs the specified operation and returns the result, similar to how an ATM processes withdrawal commands remotely.

Configuration containers or leaves in YANG are not actions—they only represent data to store or retrieve, not executable operations.

YANG input validators and Get requests

A YANG input validator is a system component that:

- checks the validity of configuration data against OpenConfig YANG models
- verifies Get requests processed through protocols such as NETCONF or gNMI
- ensures that only explicitly configured OpenConfig leaves are returned in response to Get requests.

A Get request is an operation that:

- retrieves configuration or operational data from a network device,
- uses management protocols like NETCONF or gNMI
- returns either OpenConfig leaves or Cisco native model items based on system settings and operational modes.

YANG input validators and Get requests ensure configuration integrity and accurate state retrieval in Cisco IOS XR devices. Recent enhancements to these components result in stricter input validation, with OpenConfig models providing consistent handling of configuration queries. By default, Get requests now return only leaves explicitly configured via OpenConfig, improving data accuracy and compliance.

Legacy mode remains available for limited releases, preserving previous behaviors where Get requests could return all convertible Cisco native items.

Table 3: Feature History Table

Feature Name	Release Information	Description
Improved YANG Input Validator and Get Requests	Release 7.10.1	The OpenConfig data models provide a structure for managing networks via YANG protocols. With this release, enhancements to the configuration architecture improve input validations and ensure that the Get requests made through gNMI or NETCONF protocols return only explicitly configured OpenConfig leaves. Previously, Get requests returned all the items in the Cisco native data models that the system could convert into OpenConfig items, regardless of whether they were initially configured via OpenConfig. We have added a new legacy mode option for a limited number of releases which helps you preserve this behaviour.

Legacy Mode Usage

- NETCONF: Add a legacy mode attribute to the **<get-config>** request **tag:get-config xmlns:xr-md="http://cisco.com/ns/yang/cisco-xr-metadata" xr-md:mode="legacy"**
- gNMI: Set the origin to **openconfig-legacy** in the request.

By default, OpenConfig leaves now return default values consistently using Explicit Basic Mode (see RFC6243).

YANG input validators usage guidelines and limitations

Use these best practices and follow the warnings when working with YANG input validators and OpenConfig in Cisco IOS XR:

Usage guidelines

Consider the following usage guidelines:

- After upgrading to Cisco IOS XR Release 7.10.1 or later, always commit OpenConfig changes to ensure OpenConfig leaves appear in Get requests.
- Use either gNMI or NETCONF as the management agent for OpenConfig, but do not use both simultaneously. The first successful commit determines the active agent.
- Fully configure each feature via OpenConfig, native model, or CLI. Items overridden in native models do not appear in the OpenConfig view.

Operational limitations

- Observe the same commitment requirements when downgrading or upgrading the software.
- The non-active agent can only configure native model items or perform Get requests. It cannot modify OpenConfig items until all OpenConfig leaves are first removed by the active agent.
- Changes made using CLI or Config Scripts are not reflected in OpenConfig system views.
- During commits, you can issue Get requests only on the running datastore. Edits, commits, or Get requests on candidate datastores belonging to other clients are rejected.

Warnings

- The command `show running-config | (xml | json) openconfig` displays the running OpenConfig configuration but cannot be filtered using XR CLI configuration keywords. Starting from Cisco IOS XR Release 2.4.4.1, this command is not supported.
- Do not use `load rollback changes` or `load commit changes` for rollbacks or commits that include OpenConfig leaves; these operations are not supported for OpenConfig data.
- System events, such as install operations or startup configuration failures, may remove configurations from the system, leaving OpenConfig views temporarily unsynchronized. Watch for syslog messages and reapply OpenConfig configurations when needed.
- Performing a `Commit Replace` operation through CLI removes all OpenConfig entries from the system.

Pseudo-atomic commit capabilities

A pseudo-atomic commit capability is a Cisco IOS XR configuration feature that

- ensures configuration changes are either fully applied or not applied at all,
- maintains synchronization between the system database and the OpenConfig datastore, and
- prevents partially committed configuration states in the event of commit failures.

Pseudo-atomic commit capabilities were introduced to address issues where, during the commit process, failures could result in the system database and OpenConfig datastore becoming out of sync.

From IOS XR Software Release 7.10.1 onwards, pseudo-atomic commit ensures that, if a commit failure occurs, all configuration changes are rolled back in both the system database and OpenConfig datastore. This prevents situations where only some changes take effect, thereby avoiding partial or inconsistent configurations.

Table 4: Feature history

Feature Name	Release Information	Description
Prevent Partial Pseudo-Atomic Committed Configurations	Release 7.10.1	You can now prevent the partially-committed configurations on the router and thus ensure the system database and OpenConfig datastore stay in sync. This feature changes how the internal rollback error is handled when a pseudo-atomic commit fails. In such cases, the system database always rolls back the configuration in its datastore thereby ensuring that there is no partially-committed configuration. If there is still inconsistency, the system displays error messages to notify you of various internal rollback failure scenarios based on which you must take rectification action to re-synchronize the data.

Existing and enhanced pseudo-atomic commit behavior

Behavior	Existing (before 7.10.1)	Enhanced (7.10.1 and later)
Commit success criteria	All changes in the commit must succeed for operation to complete. Errors result in complete rollback.	Same as existing behavior.
Partial rollback failure	If rollback fails (e.g., verifier rejects changes), the system may end up with some changes committed, resulting in partial configuration.	The internal rollback process always reverts changes, or issues error messages for rectification, thus avoiding partial configuration states.
Synchronization of datastores	System database and OpenConfig datastore may become out of sync if a rollback fails.	System database and OpenConfig datastore remain in sync, even after internal rollback issues.
User notification and action	Errors may become untraceable, leading to inconsistent operation unless user takes corrective action based on system messages.	System provides specific error messages, enabling the user to re-synchronize data.

Examples: Internal rollback error scenarios

Verifier rejection:

When the verifier process rejects configuration during internal rollback, the system database displays an error message such as:

```
%MGBL-VERIFIER-4-COMMIT_ROLLBACK_REJECTED
```

```
%MGBL-VERIFIER-4-COMMIT_ROLLBACK_REJECTED : verify_process incorrectly rejected rollback
of a failed commit to a previously accepted state. The rollback change has been made anyway.
 (/cfg/gl/test/item1, 0x40828400)
```

Verifier timeout:

When the verifier does not respond within 300 seconds, a timeout occurs and is indicated by:

```
%MGBL-VERIFIER-3-COMMIT_ROLLBACK_TIMEOUT
```

```
%MGBL-VERIFIER-3-COMMIT_ROLLBACK_TIMEOUT : verify_process (jid 68368, 0/0/CPU0) took too
long to verify the rollback of a failed commit
(cfg/if/act/GigabitEthernet0_0_0_2/a/test/item3). The rollback change has been made anyway.
```

Verifier failure:

When the verifier fails to apply the rollback or a timeout occurs, the following message is shown:

```
%MGBL-VERIFIER-3-COMMIT_ROLLBACK_FAILED
```

```
%MGBL-VERIFIER-3-COMMIT_ROLLBACK_FAILED : verify_process failed to apply the rollback of a
 failed commit (/cfg/gl/test/item1, 0x40828400) and may no longer operate as configured.
The process need to be restarted to rectify the error.
```

Additional reference information

You can check for partial configurations by using the following command:

```
show config commit changes [commit_id]
```

If error messages are displayed, follow the recommended actions to re-synchronize the system database and the verifier process to maintain system consistency.

Automatic resynchronization of OpenConfig configurations

An OpenConfig resynchronization event is a Cisco IOS XR automation feature that

- automatically restores consistency between OpenConfig and running configurations after system events,
- generates system logs to notify administrators about synchronization issues and the status of resynchronization attempts, and
- eliminates the need for manual replacement of OpenConfig configurations when discrepancies occur.

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Automatic resynchronization of OpenConfig configuration	Release 7.11.1	OpenConfig infrastructure can now reapply all the OpenConfig configurations automatically if there are any discrepancies in the running configuration. With this feature, there is no need for manual replacement of the OpenConfig configuration using Netconf or gNMI. The re-sync operation is triggered if the running configurations and the OpenConfig configuration go out of sync after any system event that removes some running configurations from the system. A corresponding system log gets generated to indicate the re-sync status.

Automated OpenConfig synchronisation in Cisco IOS XR

In Cisco IOS XR, OpenConfig and running configurations can go out of sync during certain operations, such as interface breakout, software installation, or new line card insertion. Previously, restoring sync required a full replacement of the OpenConfig configuration using Netconf or gNMI.

From the Cisco IOS XR Software Release 7.11.1, if the OpenConfig configurations and running configurations go out of sync, or any activities takes place which may result in the two configurations to go out of sync, the system automatically reapplies all the OpenConfig configurations and resolve the sync issue. If there is a synchronization issue between the running configuration and the OpenConfig configuration, a corresponding system log is generated to indicate it. Similarly, a corresponding system log is generated indicating the status of the re-synchronization attempt.

This feature is enabled by default. This process is completely automated.

From the Cisco IOS XR Software Release 24.1.1, the new `Cisco-IOS-XR-yiny-oper` YANG model displays the OpenConfig configuration which is out of sync with the running configuration, including the error associated with each out of sync configuration.

The `Cisco-IOS-XR-yiny-oper` operational data is a snapshot of the current system status, rather than a record of all past failures. That is, if an item of configuration is out of sync and is later resolved, such as through a resynchronization or another configuration operation, then this configuration is no longer considered out of sync and is removed from the snapshot.

Automatic re-synchronization of OpenConfig and running configurations

Starting with Cisco IOS XR Software Release 7.11.1, if OpenConfig and running configurations go out of sync, the system automatically reapplies all OpenConfig configurations to resolve the issue. System logs are generated to indicate any synchronization issues and re-synchronization attempts, which are performed automatically by default.

Viewing out-of-sync OpenConfig configurations

From Cisco IOS XR Release 24.1.1, the Cisco-IOS-XR-yiny-oper YANG model allows viewing of any OpenConfig configurations that are out of sync with the running configuration, including the associated errors. The model provides a snapshot of the current system status, displaying only present inconsistencies rather than a record of past sync issues.

Operations that remove running configuration

Here are three types of operation that can have the effect of removing running configuration from the system. Running configurations are either affected because they directly remove configuration in the system or because they result in configuration failing to be accepted by the system during start-up.

- Install operations: Running configuration can be removed during non-reload and reload install operations. During non-reload install, running configuration is removed when it is incompatible with the new software. In this case, it is directly removed by the Install infra. The configuration is removed during reload install operations if the attempt to restore the startup configuration is partially successful.
- Breakout interfaces configuration: When breakout interfaces are configured or de-configured, all the existing configuration on interfaces is affected. The affect may be creation or deletion of the parent and child interfaces. This results in an inconsistency between the running configuration and the OpenConfig datastore for any of the removed configurations that was mapped from OpenConfig configuration.

  The automatic restoration of OpenConfig configuration resolves this inconsistency by re-adding that removed configuration.
- New line card insertion: On insertion of a new line card into the system, any pre-configuration for that card is verified for the first time and may be rejected, causing it to be removed. This results in an inconsistency between the running configuration and the OpenConfig datastore.

In any of the above scenarios, if there is a sync issue, system logs are generated and the system tries to reapply all the OpenConfig configurations. If the re-sync attempt is successful, the configurations which were removed earlier, are re-applied. If the re-sync attempt fails, this means that some of the OpenConfig configuration is no longer valid.

Note The above scenarios are invalid if there are no OpenConfig configuration present in the system.

System logs Indicating out-of-sync configuration

System log messages are generated due to the above operations that can lead to discrepancies in configurations on the router. Listed are examples of system log messages raised if any such discrepancies occur.

Table 6: Examples of system log messages generated due to Out-of-Sync Configurations :

Event Name Displayed in the System Log	Description
unexpected commit errors	When an unexpected commit errors in case of a SysDB server crash.
config rollback (to a commit ID created using a different software version)	When a configuration rollbacks back to a commit ID created using a different software version.

Event Name Displayed in the System Log	Description
inconsistent configuration	This system log is generated when an inconsistency alarm is raised due to failure in restoring the start-up configurations after activities like system reload or insertion of a new line card. Re-synchronization of the configuration is triggered only after the alarm is cleared.
configuration removal (triggered on 0/2/CPU0 by the last config operation for interface GigabitEthernet0/2/0/0 and 6 other interfaces)	When interface configuration is removed in response to a change in interface breakout configuration.
configuration removal (to prepare for an install operation)	Configuration is removed from the system during a non-reload install operation due to incompatibility with the new software.

Alarms for out-of-sync OpenConfig configuration

- Inconsistency alarm: When a there is a failure in restoring the start-up configurations after a system reload or insertion of a new line card, inconsistency alarm is raised. If the inconsistency alarm is raised, you can see an informational system log is generated which indicates that the OpenConfig configuration and running configuration may be out of sync. A re-sync attempt will be made when the configuration inconsistency alarm is cleared. This system log is an early warning that the system is potentially out of sync.

  Inconsistency alarm message:

  ```
  NMI OpenConfig configuration is potentially out of sync with the running configuration
  (details: system configuration become inconsistent during OIR restore on 0/0/CPU0). An
  automatic reapply of the OpenConfig configuration will be performed when the inconsistency
  alarm is cleared.
  ```

- Missing item in the OpenConfig datastore alarm: If there are missing items in the configurations which could not be added to the OpenConfig datastore while loading in a snapshot from disk, you can see an error system log is raised which indicates that there are some items which are absent in the running OpenConfig configuration. This scenario occurs when the yang schema is changed from the time the snapshot was created.

  Item missing alarm message:

  ```
  gNMI OpenConfig configuration is potentially out of sync with the running configuration:
  3 failed to be applied to the system (details: snapshot 2 was created with a different
  schema version). The system may contain config items mapped from OC that no longer exist
  in the OC datastore. Automatic attempts to reapply OC will not remove these items, even
  if they otherwise succeed. Config should be replaced manually using a GNMI Replace
  operation.
  ```

System logs from configuration resynchronization

When an attempt to re-apply OpenConfig (resynchronization) is complete, the following informational system logs are generated to indicate the user that the OpenConfig and running configuration were out of sync, and whether the attempt to resolve this was successful.

- Successful re-sync:

  ```
  As a result of configuration removal (to prepare for an install operation), the gNMI
  OpenConfig configuration has been successfully reapplied.
  ```

- Unsuccessful re-sync:

  ```
  As a result of configuration removal (to prepare for an install operation), an attempt
  to reapply the gNMI configuration was made, but some items remain out of sync with the
  running configuration. The configuration should be reapplied manually using a GNMI
  Replace operation.
  ```

- Re-sync failure during mapping of OpenConfig configurations to XR configurations:

  ```
  As a result of configuration removal (to prepare for an install operation), the attempt
  to reapply the gNMI OpenConfig configuration failed, and the out of sync configuration
  could not be updated. gNMI OpenConfig configuration is potentially out of sync with the
  running configuration. Configuration should be reapplied manually using a GNMI Replace
  operation
  ```

Re-sync failure during mapping of OpenConfig configurations to XR configurations is a rare scenario. When there is a failure in the re-sync process while mapping the OpenConfig configuration to XR items, it causes the re-sync request to aborted. This scenario is only possible after an install which changes the OpenConfig mappings such that some configuration is no longer supported.