Supported YANG Models in NCS 1004

The supported config and oper YANG models for NCS 1004 are listed below:

Config Yang Models	**Oper Yang Models**
Cisco-IOS-XR-osa-cfg.yang	Cisco-IOS-XR-osa-oper.yang
Cisco-IOS-XR-controller-optics-cfg.yang	Cisco-IOS-XR-controller-optics-oper.yang
Cisco-IOS-XR-pmengine-cfg.yang	Cisco-IOS-XR-pmengine-oper.yang
Cisco-IOS-XR-ethernet-lldp-cfg.yang	Cisco-IOS-XR-ethernet-lldp-oper.yang
Cisco-IOS-XR-ifmgr-cfg.yang	Cisco-IOS-XR-telemetry-model-driven-oper.yang
Cisco-IOS-XR-telemetry-model-driven-cfg.yang	Cisco-IOS-XR-fpd-infra-oper.yang
Cisco-IOS-XR-fpd-infra-cfg.yang	Cisco-IOS-XR-ikev2-oper.yang
Cisco-IOS-XR-ikev2-cfg.yang	Cisco-IOS-XR-otnsec-oper.yang

The supported versions of Open Config model are listed below:

- openconfig-platform.yang
- openconfig-platform-transceiver.yang
- openconfig-terminal-device.yang
- openconfig-interfaces.yang
- openconfig-system.yang

Note openconfig-platform-transceiver.yang model is the augmented model of openconfig-platform.yang model.

Configure Slice

Step 1 Use the Cisco-IOS-XR-osa-cfg.yang YANG model for provisioning the slice with traffic on the client and trunk ports.

All the five client ports of the slice need to be configured at the same bitrate except for mixed mode configuration. Both the trunk ports are always set with the same FEC mode. In mixed mode configuration, the client ports are configured at different bitrates.

YANG model	Example

Cisco-IOS-XR-osa-cfg.yang	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">`
	`  <edit-config>`
	`    <target>`
	`      <candidate/>`
	`    </target>`
	`    <config>`
	`      <active-nodes`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-config-mda-cfg">`
	`        <active-node>`
	`          <node-name>0/1</node-name>`
	`          <mxponder-slices`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-osa-cfg">`
	`            <mxponder-slice>`
	`              <slice-id>0</slice-id>`
	`              <trunk-rates>six-hundred-gig</trunk-rates>`
	`              <client-rates>hundred-gig-e</client-rates>`
	`            </mxponder-slice>`
	`          </mxponder-slices>`
	`        </active-node>`
	`      </active-nodes>`
	`    </config>`
	`  </edit-config>`
	`</rpc>`

YANG model	Example

Cisco-IOS-XR-osa-cfg.yang	`<?xml version="1.0"?>`
	`<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<edit-config>`
	`<target>`
	`<candidate/>`
	`</target>`
	`<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<hardware-module xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-osa-cfg">`
	
	`<node>`
	`<location>0_RP0_CPU0</location>`
	`<slice>`
	`<values>`
	`<client-rate>ten-and-hundred-gig</client-rate>`
	`<trunk-rate>two-hundred-gig</trunk-rate>`
	`<fec>sd7</fec>`
	`</values>`
	`<slice-id>0</slice-id>`
	`</slice>`
	`</node>`
	`</hardware-module>`
	`</config>`
	`</edit-config>`
	`</rpc>`

Step 2 Use the Cisco-IOS-XR-osa-oper.yang YANG model to verify the slice configuration.

YANG model	Example

Cisco-IOS-XR-osa-oper.yang	`<?xml version="1.0" ?>`
	`<rpc message-id="856612"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<get>`
	`<filter>`
	`<hw-module`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-osa-oper" >`
	`   <slice-all>`
	`    <slice-info>`
	`     <slice-id>0</slice-id>`
	`    </slice-info>`
	`   </slice-all>`
	
	`   <slice-all>`
	`    <slice-info>`
	`     <slice-id>1</slice-id>`
	`    </slice-info>`
	`   </slice-all>`
	
	`   <slice-all>`
	`    <slice-info>`
	`     <slice-id>2</slice-id>`
	`    </slice-info>`
	`   </slice-all>`
	
	`    <slice-all>`
	`    <slice-info>`
	`     <slice-id>3</slice-id>`
	`    </slice-info>`
	`   </slice-all>`

YANG model	Example

	` </hw-module>`
	`</filter>`
	`</get>`
	`</rpc>`

Configure Optics Controller

Step 1 Use the Cisco-IOS-XR-ifmgr-cfg.yang YANG model for configuring the optics controller.

YANG model	Example

Cisco-IOS-XR-ifmgr-cfg.yang	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">`
	`  <edit-config>`
	`    <target>`
	`      <candidate/>`
	`    </target>`
	`    <config>`
	`      <interface-configurations`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">`
	`        <interface-configuration>`
	`          <active>act</active>`
	`          <interface-name>Optics0/1/0/2</interface-name>`
	`          <shutdown xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`nc:operation="create"/>`
	`        </interface-configuration>`
	`      </interface-configurations>`
	`    </config>`
	`  </edit-config>`
	`</rpc>`

Step 2 Use the Cisco-IOS-XR-controller-optics-cfg.yang YANG model for configuring the wavelength on the trunk port.

YANG model	Example

Cisco-IOS-XR-controller-optics-cfg.yang	`<?xml version="1.0"?>`
	`<rpc message-id="102" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<edit-config>`
	`<target>`
	`<candidate/>`
	`</target>`
	`<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`   <interface-configurations`
	`      xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">`
	`      <interface-configuration>`
	`      <active>act</active>`
	`          <interface-name>Optics0/0/0/2</interface-name>`
	`          <optics`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-controller-optics-cfg">`
	`<optics-dwdm-carrier>`
	`<grid-type>50g-hz-grid</grid-type>`
	`<param-type>itu-ch</param-type>`
	`<param-value>1</param-value>`
	`</optics-dwdm-carrier>`
	`</optics>`

YANG model	Example

	`</interface-configuration>`
	`</interface-configurations>`
	
	`</config>`
	`</edit-config>`
	`</rpc>`

Step 3 Use the Cisco-IOS-XR-controller-optics-oper.yang YANG model to verify the wavelength and channel mapping for trunk optics controllers.

YANG model	Example

Cisco-IOS-XR-controller-optics-oper.yang	`<?xml version="1.0" ?>`
	`<rpc message-id="8566" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<get>`
	`<filter type="subtree">`
	
	`  <optics-oper`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-controller-optics-oper">`
	
	`   <optics-ports>`
	`    <optics-port>`
	`     <name>Optics0/0/0/13</name>`
	`     <optics-dwdm-carrrier-channel-map>`
	
	`     </optics-dwdm-carrrier-channel-map>`
	`     </optics-port>`
	`     </optics-ports>`
	`     </optics-oper>`
	
	`</filter>`
	`</get>`
	`</rpc>`

Configure Ethernet and Coherent DSP Controllers

Step 1 Use the Cisco-IOS-XR-ifmgr-cfg.yang YANG model to configure the Ethernet controller.

YANG model	Example

Cisco-IOS-XR-ifmgr-cfg.yang	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">`
	`  <edit-config>`
	`    <target>`
	`      <candidate/>`
	`    </target>`
	`    <config>`
	`      <interface-configurations`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">`
	`        <interface-configuration>`
	`          <active>act</active>`
	`          <interface-name>HundredGigECtrlr0/1/0/2</interface-name>`
	`          <shutdown xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`nc:operation="create"/>`
	`        </interface-configuration>`

YANG model	Example

```
      </interface-configurations>
    </config>
  </edit-config>
</rpc>
```

Step 2 Use the Cisco-IOS-XR-ifmgr-cfg.yang YANG model to configure the Coherent DSP controller.

YANG model	Example
Cisco-IOS-XR-ifmgr-cfg.yang	

```
<?xml version="1.0"?>
<rpc message-id="102" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
        <target>
        <candidate/>
        </target>
  <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
   <interface-configurations
      xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
      <interface-configuration>
      <active>act</active>
    <interface-name>CoherentDSP0/0/0/6</interface-name>
    <shutdown xc:operation="delete" />
   </interface-configuration>

         <interface-configuration>
      <active>act</active>
    <interface-name>CoherentDSP0/0/0/13</interface-name>
    <shutdown></shutdown>
   </interface-configuration>

         <interface-configuration>
      <active>act</active>
    <interface-name>CoherentDSP0/0/0/20</interface-name>
    <shutdown></shutdown>
   </interface-configuration>

         <interface-configuration>
      <active>act</active>
    <interface-name>CoherentDSP0/0/0/27</interface-name>
    <shutdown></shutdown>
   </interface-configuration>
      </interface-configurations>
    </config>
  </edit-config>
</rpc>
```

Step 3 Use the Cisco-IOS-XR-pfi-im-cmd-ctrlr-oper.yang YANG model to display the name, status, and port description of the Ethernet controller.

YANG model	Example
Cisco-IOS-XR-pfi-im-cmd-ctrlr-oper.yang	

```
<?xml version="1.0" ?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
<filter>
<controllers
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-pfi-im-cmd-ctrlr-oper">

     <controllers>
        <controller>
```

YANG model	Example

```
            <interafce-name>HundredGigECtrlr0/0/0/8
            </interafce-name>
         </controller>
      </controllers>
</controllers>
</filter>
</get>
</rpc>
```

Configure the GCC Interface

Use the Cisco-IOS-XR-controller-odu-cfg.yang YANG model to configure GCC interface.

YANG model	Example
Cisco-IOS-XR-controller-odu-cfg	

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <interface-configurations
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
        <interface-configuration>
          <active xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
nc:operation="create">act</active>
          <interface-name>ODU40/0/0/0/2</interface-name>
          <odu
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-controller-odu-cfg">
            <gcc-modes>
              <gcc-mode>
                <type>gcc2-mode</type>
                <mode>enable</mode>
              </gcc-mode>
            </gcc-modes>
          </odu>
        </interface-configuration>
      </interface-configurations>
    </config>
  </edit-config>
</rpc>
<rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
message-id="urn:uuid:34d98974-474a-4396-ad1a-6dd4ddfa20bc">
 <ok/>
</rpc-reply>
```

Configure idle insertion

Use the Cisco-IOS-XR-drivers-icpe-ethernet-cfg.yang YANG model to configure idle insertion.

YANG model	Example
Cisco-IOS-XR-drivers-icpe-ethernet-cfg	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101"> <edit-config> <target> <candidate/> </target> <config> <interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XF <interface-configuration> <active>act</active> <interface-name>HundredGigECtrlr0/1/0/9</interface-name> <holdoff-time xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-drivers-icpe-ethernet-cfg">30</ </interface-configuration> </interface-configurations> </config> </edit-config> </rpc> <rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:34d98974-474a-4396-ad1a-6dd4ddfa20bc"> <ok/> </rpc-reply>`

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XF
        <interface-configuration>
          <active>act</active>
          <interface-name>HundredGigECtrlr0/1/0/9</interface-name>
          <holdoff-time
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-drivers-icpe-ethernet-cfg">30</

        </interface-configuration>
      </interface-configurations>
    </config>
  </edit-config>
</rpc>
<rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
message-id="urn:uuid:34d98974-474a-4396-ad1a-6dd4ddfa20bc">
 <ok/>
</rpc-reply>
```

Configure Loopback

Step 1 Use the Cisco-IOS-XR-ifmgr-cfg.yang and Cisco-IOS-XR-controller-otu-cfg YANG models for configuring Loopback.

YANG model	Example
Cisco-IOS-XR-ifmgr-cfg.yang Cisco-IOS-XR-controller-otu-cfg.yang	see below

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
message-id="101">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <interface-configurations
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
        <interface-configuration>
          <active>act</active>
          <interface-name>CoherentDSP0/1/0/0</interface-name>
          <otu
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-controller-otu-cfg">
            <otn-send-tti>

<string-type>send-tti-full-ascii/full-ascii</string-type>
```

YANG model	Example

	`<full-ascii-string>test1234</full-ascii-string>`
	`</otn-send-tti>`
	`<otn-expected-tti>`
	
	`<string-type>exp-tti-full-ascii/full-ascii</string-type>`
	`<full-ascii-string>test1234</full-ascii-string>`
	`</otn-expected-tti>`
	`</otu>`
	`</interface-configuration>`
	`</interface-configurations>`
	`</config>`
	`</edit-config>`
	`</rpc>`

Step 2 Use the Cisco-IOS-XR-ifmgr-cfg.yang and Cisco-IOS-XR-drivers-media-eth-cfg.yang YANG models for configuring the maintenance mode and loopback on an Ethernet controller.

YANG model	Example

Cisco-IOS-XR-ifmgr-cfg.yang
 	
Cisco-IOS-XR-drivers-media-eth-cfg.yang
	`<rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`message-id="urn:uuid:34d98974-474a-4396-ad1a-6dd4ddfa20bc">`
	
	`<ok/>`
	`</rpc-reply>`

Configure Laser Squelch

Use the Cisco-IOS-XR-ifmgr-cfg.yang YANG model to configure laser squelch.

YANG model	Example

Cisco-IOS-XR-ifmgr-cfg.yang	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">`
	`<edit-config>`
	`<target>`
	`<candidate/>`
	`</target>`
	`<config>`
	`<interface-configurations`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">`
	`<interface-configuration>`
	`<active>act</active>`
	`<interface-name>HundredGigECtrlr0/1/0/9</interface-name>`
	`<laser-squelch`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-drivers-icpe-ethernet-cfg"/>`
	`</interface-configuration>`
	`</interface-configurations>`
	`</config>`
	`</edit-config>`
	`</rpc><rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`message-id="urn:uuid:34d98974-474a-4396-ad1a-6dd4ddfa20bc">`

YANG model	Example

	` <ok/>`
	`</rpc-reply>`

Configure OTNsec on ODU4 Controllers

Step 1 Use the Cisco-IOS-XR-otnsec-cfg.yang YANG model to configure OTNsec on ODU4 controllers.

YANG model	Example

Cisco-IOS-XR-otnsec-cfg	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">`
	`  <edit-config>`
	`    <target>`
	`      <candidate/>`
	`    </target>`
	`    <config>`
	`      <interface-configurations`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">`
	`        <interface-configuration>`
	`          <active xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`nc:operation="create">act</active>`
	`          <interface-name>ODU40/0/0/0/2</interface-name>`
	`         <odu-otnsec xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-otnsec-cfg">`
	
	`            <ipv4>`
	`              <session-id>1</session-id>`
	`              <destination-address>1.1.1.1</destination-address>`
	`              <source-address>1.1.1.2</source-address>`
	`            </ipv4>`
	`            <ik-ev2-profile>IP1</ik-ev2-profile>`
	`            <policy>OP1</policy>`
	`          </odu-otnsec>`
	`        </interface-configuration>`
	`      </interface-configurations>`
	`    </config>`
	`  </edit-config>`
	`</rpc>`

Step 2 Use the Cisco-IOS-XR-otnsec-cfg.yang YANG model to configure OTNsec on ODU4 controllers.

YANG model	Example

Cisco-IOS-XR-otnsec-cfg.yang	`<rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"`
	`message-id="urn:uuid:34d98974-474a-4396-ad1a-6dd4ddfa20bc">`
	` <ok/>`
	`</rpc-reply>`

Configure get keyring

Use the Cisco-IOS-XR-keyring-oper.yang YANG model for getting the configured keyring.

YANG model	Example

Cisco-IOS-XR-keyring-oper.yang	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">`
	`  <get>`
	`    <filter>`
	`      <keyrings`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-keyring-oper"/>`
	`    </filter>`
	`  </get>`
	`</rpc><?xml version="1.0"?>`
	`<rpc-reply message-id="101"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	` <data>`
	`  <keyrings xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-keyring-oper">`
	
	`   <keyring>`
	`    <name>KR1</name>`
	`    <keyring-name>KR1</keyring-name>`
	`    <total-peers>8</total-peers>`
	`    <peer>`
	`     <peer-name>SITE-A-1</peer-name>`
	`     <ip-address>1.1.1.1</ip-address>`
	`     <subnet>255.255.255.255</subnet>`
	`     <local-psk>Configured</local-psk>`
	`     <remote-psk>Configured</remote-psk>`
	`    </peer>`
	`    <peer>`
	`     <peer-name>SITE-A-2</peer-name>`
	`     <ip-address>1.1.2.1</ip-address>`
	`     <subnet>255.255.255.255</subnet>`
	`     <local-psk>Configured</local-psk>`
	`     <remote-psk>Configured</remote-psk>`
	`    </peer>`
	`    <peer>`
	`     <peer-name>SITE-A-3</peer-name>`
	`     <ip-address>1.1.3.1</ip-address>`
	`     <subnet>255.255.255.255</subnet>`
	`     <local-psk>Configured</local-psk>`
	`     <remote-psk>Configured</remote-psk>`
	`    </peer>`
	`    <peer>`
	`     <peer-name>SITE-A-4</peer-name>`
	`     <ip-address>1.1.4.1</ip-address>`
	`     <subnet>255.255.255.255</subnet>`
	`     <local-psk>Configured</local-psk>`
	`     <remote-psk>Configured</remote-psk>`
	`    </peer>`
	`    <peer>`
	`     <peer-name>SITE-A-5</peer-name>`
	`     <ip-address>1.1.5.1</ip-address>`
	`     <subnet>255.255.255.255</subnet>`
	`     <local-psk>Configured</local-psk>`
	`     <remote-psk>Configured</remote-psk>`
	`    </peer>`
	`    <peer>`
	`     <peer-name>SITE-A-6</peer-name>`

YANG model	Example
	```
	     <ip-address>1.1.6.1</ip-address>
	     <subnet>255.255.255.255</subnet>
	     <local-psk>Configured</local-psk>
	     <remote-psk>Configured</remote-psk>
	    </peer>
	    <peer>
	     <peer-name>SITE-A-7</peer-name>
	     <ip-address>1.1.7.1</ip-address>
	     <subnet>255.255.255.255</subnet>
	     <local-psk>Configured</local-psk>
	     <remote-psk>Configured</remote-psk>
	    </peer>
	    <peer>
	     <peer-name>SITE-A-8</peer-name>
	     <ip-address>1.1.8.1</ip-address>
	     <subnet>255.255.255.255</subnet>
	     <local-psk>Configured</local-psk>
	     <remote-psk>Configured</remote-psk>
	    </peer>
	   </keyring>

	  </keyrings>
	 </data>
	</rpc-reply>
	```

Configure get ikev2 proposal/policy/profile

Use the Cisco-IOS-XR-osa-cfg.yang YANG model for getting the ikev2 proposal/policy/profile.

YANG model	Example

Cisco-IOS-XR-osa-cfg.yang	
	```
	<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
	  <get>
	    <filter>
	      <ikev2 xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ikev2-oper"/>
	    </filter>
	  </get>
	</rpc><rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">

	  <get>
	    <filter>
	      <ikev2 xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ikev2-oper"/>
	    </filter>
	  </get>
	</rpc>

	...........
	Response - Wed Jul 31 2019 14:16:35

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

	 <data>
	  <ikev2 xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ikev2-oper">
	   <nodes>
	```

YANG model	Example

```
    <node>
     <node-name>0/RP0/CPU0</node-name>
     <stats>
      <ike-sa-init-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>508</tx-res>
       <rx-req>508</rx-req>
      </ike-sa-init-cnt>
      <ike-auth-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>417</tx-res>
       <rx-req>417</rx-req>
      </ike-auth-cnt>
      <create-child-sa-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>60219</tx-res>
       <rx-req>60219</rx-req>
      </create-child-sa-cnt>
      <create-child-sa-ipsec-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>1149</tx-res>
       <rx-req>1149</rx-req>
      </create-child-sa-ipsec-cnt>
      <create-child-sa-ipsec-rekey-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>37445</tx-res>
       <rx-req>37445</rx-req>
      </create-child-sa-ipsec-rekey-cnt>
      <create-child-sa-ike-rekey-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>21625</tx-res>
       <rx-req>21625</rx-req>
      </create-child-sa-ike-rekey-cnt>
      <gsk-auth-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </gsk-auth-cnt>
      <gsk-reg-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </gsk-reg-cnt>
      <gsk-rekey-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </gsk-rekey-cnt>
      <gsk-rekey-ack-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
```

YANG model	Example
	(continued below)

```
      </gsk-rekey-ack-cnt>
      <informational-cnt>
       <tx-req>119863</tx-req>
       <rx-res>117976</rx-res>
       <tx-res>177059</tx-res>
       <rx-req>177298</rx-req>
      </informational-cnt>
      <unsupported-critical-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </unsupported-critical-cnt>
      <invalid-ike-spi-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </invalid-ike-spi-cnt>
      <invalid-major-version-ikev2-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </invalid-major-version-ikev2-cnt>
      <invalid-syntax-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </invalid-syntax-cnt>
      <invalid-msg-id-ikev2-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </invalid-msg-id-ikev2-cnt>
      <invalid-spi-ikev2-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </invalid-spi-ikev2-cnt>
      <no-proposal-chosen-ikev2-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </no-proposal-chosen-ikev2-cnt>
      <invalid-ke-pyld-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </invalid-ke-pyld-cnt>
      <auth-failed-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </auth-failed-cnt>
      <single-pair-required-cnt>
```

YANG model	Example

```
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </single-pair-required-cnt>
       <no-additional-sas-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </no-additional-sas-cnt>
       <internal-addr-failure-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </internal-addr-failure-cnt>
       <failed-cp-required-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </failed-cp-required-cnt>
       <ts-unacceptable-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </ts-unacceptable-cnt>
       <invalid-selectors-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </invalid-selectors-cnt>
       <initial-contact-ikev2-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>417</rx-req>
       </initial-contact-ikev2-cnt>
       <set-window-size-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>22042</tx-res>
        <rx-req>22042</rx-req>
       </set-window-size-cnt>
       <additional-ts-poss-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </additional-ts-poss-cnt>
       <ipcomp-supported-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
        <tx-res>0</tx-res>
        <rx-req>0</rx-req>
       </ipcomp-supported-cnt>
       <nat-detection-src-cnt>
        <tx-req>0</tx-req>
        <rx-res>0</rx-res>
```

YANG model	Example
	```
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </nat-detection-src-cnt>
	     <nat-detection-dst-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </nat-detection-dst-cnt>
	     <cookie-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </cookie-cnt>
	     <use-transport-mode-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </use-transport-mode-cnt>
	     <http-cert-lookup-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </http-cert-lookup-cnt>
	     <rekey-sa-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>37445</rx-req>
	     </rekey-sa-cnt>
	     <esp-tfc-padding-not-supported-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </esp-tfc-padding-not-supported-cnt>
	     <delete-reason-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </delete-reason-cnt>
	     <custom-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </custom-cnt>
	     <redirect-supported-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	     </redirect-supported-cnt>
	     <redirect-cnt>
	      <tx-req>0</tx-req>
	      <rx-res>0</rx-res>
	      <tx-res>0</tx-res>
	      <rx-req>0</rx-req>
	```

YANG model	Example
	(continued below)

```
      </redirect-cnt>
      <redirected-from-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </redirected-from-cnt>
      <ikev2-dpd-cnt>
       <tx-req>119699</tx-req>
       <rx-res>117972</rx-res>
       <tx-res>117989</tx-res>
       <rx-req>117989</rx-req>
      </ikev2-dpd-cnt>
      <cfg-request-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </cfg-request-cnt>
      <cfg-reply-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>417</rx-req>
      </cfg-reply-cnt>
      <cfg-set-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </cfg-set-cnt>
      <cfg-ack-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </cfg-ack-cnt>
      <nat-inside-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </nat-inside-cnt>
      <nat-outside-cnt>
       <tx-req>0</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </nat-outside-cnt>
      <no-nat-cnt>
       <tx-req>508</tx-req>
       <rx-res>0</rx-res>
       <tx-res>0</tx-res>
       <rx-req>0</rx-req>
      </no-nat-cnt>
      <tx-succ>360612</tx-succ>
      <rx-succ>356689</rx-succ>
      <tx-write-fail>0</tx-write-fail>
      <tx-send-fail>0</tx-send-fail>
      <tx-set-qos-fail>0</tx-set-qos-fail>
      <tx-socket-not-conn>0</tx-socket-not-conn>
      <tx-unknown-src-port>0</tx-unknown-src-port>
```

YANG model	Example
	(continued below)

```
     <rx-get-if-fail>0</rx-get-if-fail>
     <rx-null-ifhndl>0</rx-null-ifhndl>
     <rx-get-ntwrk-offset>0</rx-get-ntwrk-offset>
     <rx-read-ip-head>0</rx-read-ip-head>
     <rx-get-transport-offset>0</rx-get-transport-offset>
     <rx-read-upd-head>0</rx-read-upd-head>
     <rx-unexpected-marker>0</rx-unexpected-marker>
     <rx-get-vrf-id>0</rx-get-vrf-id>
     <rx-read-pyld>0</rx-read-pyld>
    </stats>

    <summary>
     <total-sa>0</total-sa>
     <total-sa-active>0</total-sa-active>
     <total-sa-negotiating>0</total-sa-negotiating>
     <total-outgoing-sa>0</total-outgoing-sa>
     <outgoing-sa-active>0</outgoing-sa-active>
     <outgoing-sa-negotiating>0</outgoing-sa-negotiating>
     <total-incoming-sa>0</total-incoming-sa>
     <incoming-sa-active>0</incoming-sa-active>
     <incoming-sa-negotiating>0</incoming-sa-negotiating>
    </summary>

    <policies>
     <policy>
      <name>default</name>
      <policy-name>default</policy-name>
      <total-local-addr>1</total-local-addr>
      <addr>0.0.0.0</addr>
      <total-proposal>1</total-proposal>
      <proposal>default</proposal>
     </policy>

    </policies>
    <proposals>
     <proposal>
      <name>default</name>
      <proposal-name>default</proposal-name>
      <total-enc-alg>1</total-enc-alg>
      <encryption-alg>CBC-AES-256</encryption-alg>
      <total-hash-alg>2</total-hash-alg>
      <hash-alg>SHA 512</hash-alg>
      <hash-alg>SHA 384</hash-alg>
      <total-prf-alg>2</total-prf-alg>
      <prf-alg>SHA 512</prf-alg>
      <prf-alg>SHA 384</prf-alg>
      <total-group-alg>3</total-group-alg>
      <group-alg>Group 19</group-alg>
      <group-alg>Group 20</group-alg>
      <group-alg>Group 21</group-alg>
      <status>Complete</status>
     </proposal>

    </proposals>
    <profiles>
     <profile>
      <name>IP1</name>
      <profile-name>IP1</profile-name>
      <keyring-name>KR1</keyring-name>
      <match-any>false</match-any>
      <total-match-remote-peers>8</total-match-remote-peers>
      <addr>1.1.1.1</addr>
```

YANG model	Example
	`<addr>1.1.2.1</addr>`
	`<addr>1.1.3.1</addr>`
	`<addr>1.1.4.1</addr>`
	`<addr>1.1.5.1</addr>`
	`<addr>1.1.6.1</addr>`
	`<addr>1.1.7.1</addr>`
	`<addr>1.1.8.1</addr>`
	`<mask>255.255.255.255</mask>`
	`<mask>255.255.255.255</mask>`
	`<mask>255.255.255.255</mask>`
	`<mask>255.255.255.255</mask>`
	`<mask>255.255.255.255</mask>`
	`<mask>255.255.255.255</mask>`
	`<mask>255.255.255.255</mask>`
	`<mask>255.255.255.255</mask>`
	`<lifetime-in-sec>120</lifetime-in-sec>`
	`<dpd-interval-in-sec>10</dpd-interval-in-sec>`
	`<dpd-retry-in-sec>2</dpd-retry-in-sec>`
	`</profile>`
	`</profiles>`
	`</node>`
	`</nodes>`
	`</ikev2>`
	`</data>`
	`</rpc-reply>`

Configure Performance Monitoring

Step 1 Use the Cisco-IOS-XR-ifmgr-cfg.yang and Cisco-IOS-XR-pmengine-cfg.yang YANG models for configuring the performance monitoring parameters for the Optics, Ethernet, and coherentDSP controllers.

Step 2 Use the Cisco-IOS-XR-pmengine-oper.yang YANG models to view the performance monitoring parameters for the Optics, Ethernet, and coherentDSP controllers.

The table below shows an example that displays all the PM parameters for the optics controller. You can use specific filters for the required the output.

YANG model	Example
Cisco-IOS-XR-pmengine-oper.yang	`<?xml version="1.0" ?>`
	`<rpc message-id="856612"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<get>`
	`<filter type="subtree">`
	`<performance-management`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-pmengine-oper">`
	`<optics>`
	`<optics-ports>`
	`<optics-port>Optics0/0/0/1</optics-port>`
	`</optics-ports>`
	`</optics>`
	`</performance-management>`
	`</filter>`

YANG model	Example

	`</get>`
	`</rpc>`

The table below shows an example that displays current 15 minute FEC PM for the Coherent DSP controller.

YANG model	Example

Cisco-IOS-XR-pmengine-oper.yang	`<?xml version="1.0" ?>`
	`<rpc message-id="856612"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<get>`
	`<filter type="subtree">`
	`<performance-management`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-pmengine-oper">`
	`<otu>`
	`<otu-ports>`
	`<otu-port>`
	`<name>CoherentDSP0/0/0/12</name>`
	`<otu-current>`
	`<otu-minute15>`
	`<otu-minute15fecs/>`
	`</otu-minute15>`
	`</otu-current>`
	`</otu-port>`
	`</otu-ports>`
	`</otu>`
	`</performance-management>`
	`</filter>`
	`</get>`
	`</rpc>`

NETCONF Operations

NETCONF defines one or more configuration datastores and allows configuration operations on the datastores. A configuration datastore is a complete set of configuration data that is required to get a device from its initial default state into a desired operational state. The configuration datastore does not include state data or executive commands.

The base protocol includes the following NETCONF operations:

```
|  +--Get-config
|  +--Edit-Config
|     +--Merge
|     +--Replace
|     +--Create
|     +--Delete
|     +--Remove
|     +--Default-Operations
|        +--Merge
|        +--Replace
|        +--None
|  +--Get
|  +--Lock
|  +--UnLock
|   +--Close-Session
|   +--Kill-Session
```

These NETCONF operations are described in the following table:

NETCONF Operation	**Description**	**Example**

<get-config>	Retrieves all or part of a specified configuration from a named data store	Retrieve specific interface configuration details from running configuration using filter option
		
		`<rpc message-id="101"`
		`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`<get-config>`
		`<source>`
		`<running/>`
		`</source>`
		`<filter>`
		`<interface-configurations`
		`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg"\>`
		`<interface-configuration>`
		`<active>act</active>`
		`<interface-name>TenGigE0/0/0/2/0</interface-name>`
		`</interface-configuration>`
		`</interface-configurations>`
		`</filter>`
		`</get-config>`
		`</rpc>`

<get>	Retrieves running configuration and device state information	Retrieve all acl configuration and device state information.
		
		`Request:`
		`<get>`
		`<filter>`
		`<ipv4-acl-and-prefix-list`
		`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-acl-oper"/>`
		`</filter>`
		`</get>`

NETCONF Operation	Description	Example

<edit-config>	Loads all or part of a specified configuration to the specified target configuration	Configure ACL configs using **Merge** operation
		
		`<rpc message-id="101"`
		`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`<edit-config>`
		`<target><candidate/></target>`
		`<config`
		`xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`<ipv4-acl-and-prefix-list`
		`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-acl-cfg"`
		`  xc:operation=”merge”>`
		`<accesses>`
		`<access>`
		`<access-list-name>aclv4-1</access-list-name>`
		`<access-list-entries>`
		`<access-list-entry>`
		`<sequence-number>10</sequence-number>`
		`<remark>GUEST</remark>`
		`</access-list-entry>`
		`<access-list-entry>`
		`<sequence-number>20</sequence-number>`
		`<grant>permit</grant>`
		`<source-network>`
		`<source-address>172.0.0.0</source-address>`
		`<source-wild-card-bits>0.0.255.255</source-wild-card-bits>`
		`</source-network>`
		`</access-list-entry>`
		`</access-list-entries>`
		`</access>`
		`</accesses>`
		`</ipv4-acl-and-prefix-list>`
		`</config>`
		`</edit-config>`
		`</rpc>`
		
		`Commit:`
		`<rpc message-id="101"`
		`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`<commit/>`
		`</rpc>`

<lock>	Allows the client to lock the entire configuration datastore system of a device	Lock the running configuration.
		`Request:`
		`<rpc message-id="101"`
		`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		`<lock>`
		`<target>`
		`<running/>`
		`</target>`
		`</lock>`
		`</rpc>`
		
		`Response :`
		`<rpc-reply message-id="101"`
		`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
		
		` <ok/>`
		`</rpc-reply>`

NETCONF Operation	Description	Example
<Unlock>	Releases a previously locked configuration. An <unlock> operation will not succeed if either of the following conditions is true: • The specified lock is not currently active. • The session issuing the <unlock> operation is not the same session that obtained the lock.	Lock and unlock the running configuration from the same session. `Request:` `rpc message-id="101"` `xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">` `<unlock>` `<target>` `<running/>` `</target>` `</unlock>` `</rpc>` `Response -` `<rpc-reply message-id="101"` `xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">` `<ok/>` `</rpc-reply>`
<close-session>	Closes the session. The server releases any locks and resources associated with the session and closes any associated connections.	Close a NETCONF session. `Request :` `<rpc message-id="101"` `xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">` `<close-session/>` `</rpc>` `Response:` `<rpc-reply message-id="101"` `xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">` `<ok/>` `</rpc-reply>`
<kill-session>	Cancels operations currently in process, releases locks and resources associated with the session, and closes any associated connections.	Cancel a session if the ID is other session ID. `Request:` `<rpc message-id="101"` `xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">` `<kill-session>` `<session-id>4</session-id>` `</kill-session>` `</rpc>` `Response:` `<rpc-reply message-id="101"` `xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">` `<ok/>` `</rpc-reply>`

NETCONF Operation to Get Configuration

This example shows how a NETCONF <get-config> request works for CDP feature.

The client initiates a message to get the current configuration of CDP running on the router. The router responds with the current CDP configuration.

Netconf Request (Client to Router)	Netconf Response (Router to Client)

`<rpc message-id="101"`
`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
 	
`<get-config>`
`<source><running/></source>`
`<filter>`
`<cdp`
`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-cdp-cfg"/>`
`</filter>`
`</get-config>`
`</rpc>`
	`<?xml version="1.0"?>`
	`<rpc-reply message-id="101"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	
	` <data>`
	`  <cdp`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-cdp-cfg">`
	
	`   <timer>10</timer>`
	`   <enable>true</enable>`
	`   <log-adjacency></log-adjacency>`
	`   <hold-time>200</hold-time>`
	`   <advertise-v1-only></advertise-v1-only>`
	`  </cdp>`
	`#22`
	` </data>`
	`</rpc-reply>`

The <rpc> element in the request and response messages enclose a NETCONF request sent between the client and the router. The `message-id` attribute in the <rpc> element is mandatory. This attribute is a string chosen by the sender and encodes an integer. The receiver of the <rpc> element does not decode or interpret this string but simply saves it to be used in the <rpc-reply> message. The sender must ensure that the `message-id` value is normalized. When the client receives information from the server, the <rpc-reply> message contains the same `message-id`.

CLI Over NETCONF

A new yang model, Cisco-IOS-XR-cli-cfg.yang is defined, which consists of a leaf node called 'cli'. The leaf node can be used to either send or receive the CLI configurations.

Limitations:

- Process restart and sysadmin mode is not supported .
- Rollback of configuration changes is not supported.
- Copying of images and logs to and from the box is not supported.

Edit Configuration Request

Edit-Config request with the sample CLI configurations is as follows. It must be followed by a commit rpc request for the configurations to be applied on the router.

Note The operation attribute, default operation parameter in an Edit-Config request can only be "Merge". Other operation parameters are not supported.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<edit-config>
<target>
<candidate />
</target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0" >
```

```
<cli xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-cli-cfg">
interface MgmtEth0/RP0/CPU0/0
no shutdown
ipv4 address dhcp
router static
address-family ipv4 unicast
0.0.0.0/0 MgmtEth0/RP0/CPU0/0 192.168.122.1
ssh server v2
ssh server netconf
netconf-yang agent ssh
</cli>
</config>
</edit-config>
</rpc>

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

Copy Configuration Request

Copy-Config request with a sample CLI configuration is as follows. It must be followed by a commit rpc request for the configurations to be applied on the router.

Note A Copy-Config request replaces the configurations on the router with the configurations sent in the request. So, any reachability configuration (related to netconf, ssh, management ip) must be sent in the Copy-Config request.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<copy-config>
<target>
<candidate />
</target>
<source>
<config>
<cli xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-cli-cfg">
interface MgmtEth0/RP0/CPU0/0
no shutdown
ipv4 address dhcp
router static
address-family ipv4 unicast
0.0.0.0/0 MgmtEth0/RP0/CPU0/0 192.168.122.1
ssh server v2
ssh server netconf
netconf-yang agent ssh
</cli>
</config>
</source>
</copy-config>
</rpc>

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

Get Configurations Request

Get-Config request to retrieve the configurations on the router is as follows.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get-config>
<source>
```

```
<running/>
</source>
<filter type="subtree">
<cli xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-cli-cfg">
</cli>
</filter>
</get-config>
</rpc>
```

Get request to retrieve the configurations on the router.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<get>
<filter>
<cli xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-cli-cfg"/>
</filter>
</get>
</rpc>
```

Note Get requests always return the running configurations of the router and show cli is not supported in NETCONF clear text.

CLIDIFF Over NETCONF

A new RPC is implemented in the IOS-XR NETCONF agent, which can be used to retrieve the difference in the configuration changes before and after committing. The output is similar to the output of the command **show commit changes diff**. It shows the configurations added or removed after the commit is done. The added configurations will have a "+" sign and the removed configurations will have a "-" sign.

The sample output of **show commit changes diff** command is as follows:

```
RP/0/RSP0/CPU0:ASR9001-1(config)#sh commit changes diff
Fri Sep 23 08:03:07.485 UTC
Building configuration...
!! IOS XR Configuration 5.3.3
- interface Loopback1000
- description test
- ipv4 address 10.10.0.1 255.255.255.255
!
+ multicast-routing
+     address-family ipv4
+         interface Loopback0
+             enable
               !
           !
        !
+ multicast-routing
!
end
```

RPC Request

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

RPC Reply

```
<?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-cli-diff-act">Building
configuration...
!! IOS XR Configuration 6.5.1
+  vrf RED
+   address-family ipv6 unicast
     import route-target
+     65172:1
     !
     export route-target
+     65172:1
     !
    !
   !
end

</response>
</rpc-reply>
```

Note The above RPC reply is seen after adding the following CLI configuration:

```
vrf RED
 address-family ipv6 unicast
  import route-target
   65172:1
  !
  export route-target
   65172:1
  !
 !
!
```

Configure LLDP Drop

Step 1 Use the Cisco-IOS-XR-osa-cfg.yang YANG model to configure LLDP drop.

YANG model	Example

Cisco-IOS-XR-osa-cfg.yang	`<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">`
	`  <edit-config>`
	`    <target>`
	`      <candidate/>`
	`    </target>`
	`    <config>`
	`      <active-nodes xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-config-mda-cfg">`
	`        <active-node>`
	`          <node-name>0/1</node-name>`
	`          <mxponder-slices xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-osa-cfg">`
	`            <mxponder-slice>`
	`              <slice-id>0</slice-id>`
	`              <lldp xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" nc:operation="merge">true</lldp>`
	`            </mxponder-slice>`

YANG model	Example

	`          </mxponder-slices>`
	`        </active-node>`
	`      </active-nodes>`
	`    </config>`
	`  </edit-config>`
	`</rpc>`

Step 2 Use the Cisco-IOS-XR-osa-cfg.yang YANG model to delete LLDP drop configuration.

YANG model	Example

Cisco-IOS-XR-osa-cfg.yang	`<?xml version="1.0"?>`
	`<rpc message-id="102" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<edit-config>`
	`<target>`
	`<candidate/>`
	`</target>`
	`<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<hardware-module xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-osa-cfg">`
	`<node>`
	`<location>0_RP0_CPU0</location>`
	`<slice>`
	`<slice-id>0</slice-id>`
	`<lldp>false</lldp>`
	`</slice>`
	`</node>`
	`</hardware-module>`
	`</config>`
	`</edit-config>`
	`</rpc>`

Step 3 Use the Cisco-IOS-XR-osa-cfg.yang YANG model to retrieve operational data for LLDP drop.

YANG model	Example

Cisco-IOS-XR-osa-cfg.yang	`<?xml version="1.0"?>`
	`<rpc message-id="856615"`
	`xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">`
	`<get>`
	`<filter>`
	`<lldp-snoop-data`
	`xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-osa-oper.yang"/>`
	`</filter>`
	`</get>`
	`</rpc>`

IPv4 PING Over NETCONF

Use the Cisco-IOS-XR-ping-act YANG model to do the ping test to the destination IPv4 addresses.The following example shows the RPC request and RPC response messages for a successful ping test. The destination host is reachable and the success rate is 100%.

YANG Model	Example
Cisco-IOS-XR-ping-act.yang	<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:28170002-365f-45be-a8e1-e1f54d8b64b5"><ping xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <destination> <destination>10.127.60.1</destination> </destination> </ping> </nc:rpc> <rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:28170002-365f-45be-a8e1-e1f54d8b64b5"> <ping-response xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <ipv4> <destination>10.127.60.1</destination> <data-size>100</data-size> <timeout>2</timeout> <pattern>abcd</pattern> <rotate-pattern>false</rotate-pattern> <replies> <reply> <reply-index>1</reply-index> <result>!</result> </reply> <reply> <reply-index>2</reply-index> <result>!</result> </reply> <reply> <reply-index>3</reply-index> <result>!</result> </reply> <reply> <reply-index>4</reply-index> <result>!</result>

YANG Model	Example
	</reply> <reply> <reply-index>5</reply-index> <result>!</result> </reply> </replies> <hits>5</hits> <total>5</total> <success-rate>100</success-rate> <rtt-min>1</rtt-min> <rtt-avg>1</rtt-avg> <rtt-max>2</rtt-max> </ipv4> </ping-response> </rpc-reply>

The following example shows the RPC request and RPC response messages for a failure ping test. The destination host is not reachable and the success rate is 0%.

YANG model	Example
Cisco-IOS-XR-ping-act.yang	<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:28170002-365f-45be-a8e1-e1f54d8b64b5"><ping xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <destination> <destination>10.127.60.1</destination> </destination> </ping> </nc:rpc> <rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:02800209-6ebf-4955-8588-f6cdfd6f2750"> <ping-response xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <ipv4> <destination>10.127.60.171</destination> <data-size>100</data-size> <timeout>2</timeout>

YANG model	Example
	<pattern>abcd</pattern> <rotate-pattern>false</rotate-pattern> <replies> <reply> <reply-index>1</reply-index> <result>.</result> </reply> <reply> <reply-index>2</reply-index> <result>.</result> </reply> <reply> <reply-index>3</reply-index> <result>.</result> </reply> <reply> <reply-index>4</reply-index> <result>.</result> </reply> <reply> <reply-index>5</reply-index> <result>.</result> </reply> </replies> <hits>0</hits> <total>5</total> <success-rate>0</success-rate> </ipv4> </ping-response> </rpc-reply>

IPv6 PING Over NETCONF

Table 1: Feature History

Feature Name	Release	Description
NETCONF Support for READ, WRITE, and Execute or Administrative Commands.	Cisco IOS XR Release 7.3.1	Support for IPv4 and IPv6 Ping test using the Cisco-IOS-XR-ping-act YANG model, instead of using CLI commands, is available. RPC (Remote Procedure Call) Request and Response messages are used to do the ping test, which is automated using scripts. This enables you to perform the ping test in a less time-consuming manner and to enhance network scalability.

Use the Cisco-IOS-XR-ping-act YANG model to do the ping test to the destination IPv6 addresses. The following example shows the RPC request and RPC response messages for a successful ping test. The destination host is reachable and the success rate is 100%.

YANG model	Example
Cisco-IOS-XR-ping-act.yang	<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:28170002-365f-45be-a8e1-e1f54d8b64b5"><ping xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <destination> <destination>2001:420:5446:2014::281:178</destination> </destination> </ping> </nc:rpc> <rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:15798adc-f9f9-41b2-9aa5-a1c88dd788e8"> <ping-response xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <ipv6> <destination>2001:420:5446:2014::281:178</destination> <repeat-count>50</repeat-count> <data-size>100</data-size> <timeout>2</timeout> <pattern>abcd</pattern>

YANG model	Example
	<rotate-pattern>false</rotate-pattern> <replies> <reply> <reply-index>1</reply-index> <result>!</result> </reply> <reply> <reply-index>2</reply-index> <result>!</result> </reply> <reply> <reply-index>3</reply-index> <result>!</result> </reply> <reply> <reply-index>4</reply-index> <result>!</result> </reply> <reply> <reply-index>5</reply-index> <result>!</result> </reply> </replies> <hits>5</hits> <total>5</total> <success-rate>100</success-rate> <rtt-min>1</rtt-min> <rtt-avg>1</rtt-avg> <rtt-max>2</rtt-max> </ipv6> </ping-response> </rpc-reply>

The following example shows the RPC request and RPC response messages for a failure ping test. The destination host is not reachable and the success rate is 0%.

YANG model	Example
Cisco-IOS-XR-ping-act.yang	<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:28170002-365f-45be-a8e1-e1f54d8b64b5"><ping xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <destination> <destination>2001:420:5446:2014::281:178</destination> </destination> </ping> </nc:rpc> <rpc-reply xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:02800209-6ebf-4955-8588-f6cdfd6f2750"> <ping-response xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ping-act"> <ipv6> <destination>2001:420:5446:2014::281:178</destination> <data-size>100</data-size> <timeout>2</timeout> <pattern>abcd</pattern> <replies> <reply> <reply-index>1</reply-index> <result>.</result> </reply> <reply> <reply-index>2</reply-index> <result>.</result> </reply> <reply> <reply-index>3</reply-index> <result>.</result> </reply> <reply> <reply-index>4</reply-index> <result>.</result>

YANG model	Example
	</reply> <reply> <reply-index>5</reply-index> <result>.</result> </reply> </replies> <hits>0</hits> <total>5</total> <success-rate>0</success-rate> </ipv6> </ping-response> </rpc-reply>

Configure LLDP on Management Port

Table 2: Feature History

Feature Name	Release	Description
OC (Open Configuration) Support for LLDP (Link Layer Discovery Protocol) on Management Port	Cisco IOS XR Release 7.3.1	The OC support for configuring LLDP on a management port is available. This feature enables you to perform the configuration using scripts, which is less time-consuming. Also, the Open Configuration model supports the use of vendor-neutral data models to configure and manage the network.

Step 1 You can use the following script to configure LLDP on managent port.

```
"openconfig-lldp:lldp": {
"interfaces": {
  "interface": [
   {
    "name": "MgmtEth0/RP0/CPU0/0",
    "config": {
     "name": "MgmtEth0/RP0/CPU0/0",
     "enabled": true
    }
```

```
       },
       {
        "name": "MgmtEth0/RP0/CPU0/1",
        "config": {
         "name": "MgmtEth0/RP0/CPU0/1",
         "enabled": true
        }
       },
       {
        "name": "MgmtEth0/RP0/CPU0/2",
        "config": {
         "name": "MgmtEth0/RP0/CPU0/2",
         "enabled": true
        }
       }
      ]
    }
    }
```

Step 2 You can get the operational data through GNMI

```
"MgmtEth0/RP0/CPU0/1": {
                          "neighbors": {
                             "neighbor": {
                                 "SW-VEGA-CORBU-C4#Gi1/0/13": {
                                      "capabilities": {
                                         "capability": {
                                             "openconfig-lldp-types:MAC_BRIDGE": {
                                                  "state": {
                                                      "enabled": true,
                                                      "name": "openconfig-lldp-types:MAC_BRIDGE"
                                                  }
                                              },
                                              "openconfig-lldp-types:ROUTER": {
                                                  "state": {
                                                      "enabled": false,
                                                      "name": "openconfig-lldp-types:ROUTER"
                                                  }
                                              }
                                          }
                                     },
                                      "custom-tlvs": {
                                          "tlv": {
                                              "32962": {
                                                  "1": {
                                                     "295": {
                                                          "state": {
                                                              "oui": "32962",
                                                             "oui-subtype": "1",
                                                              "type": 127,
                                                              "value": "Aa8="
                                                          }
                                                      }
                                                  }
                                             },
                                             "4623": {
                                                  "1": {
                                                     "295": {
                                                         "state": {
                                                             "oui": "4623",
                                                              "oui-subtype": "1",
                                                            "type": 127,
                                                            "value": "A2wBAB4="
                                                        }
                                                    }
                                                }
                                            }
                                        }
                                    },
                                    "state": {
                                       "chassis-id": "6899.cd9f.f480",
                                        "chassis-id-type": "MAC_ADDRESS",
                                        "id": "SW-VEGA-CORBU-C4",
                                        "management-address": "4.31.25.25",
                                        "management-address-type": "ipv4",
                                        "port-description": "GigabitEthernet1/0/13",
                                        "port-id": "Gi1/0/13",
                                        "port-id-type": "INTERFACE_NAME",
                                        "system-description": "Cisco IOS Software, Catalyst L3
Switch Software (CAT3K_CAA-UNIVERSALK9-M), Version 15.0(1)EX3, RELEASE SOFTWARE (fc2)\nTechnical
Support: http://www.cisco.com/techsupport\nCopyright (c) 1986-2013 by Cisco Systems, Inc.\nCompiled
 Mon 23-Sep-13 18:24 by prod_r",
                                        "system-name": "SW-VEGA-CORBU-C4"
                                    }
                                }
                            }
                        },
                        "state": {
                            "enabled": true,
                            "name": "MgmtEth0/RP0/CPU0/1"
                        }
                    },
```

Open Configuration Model for Client FEC

Before you begin

Table 3: Feature History

Feature Name	Release	Description
OC (Open Configuration) Model for Client FEC and Laser Squelch	Cisco IOS XR Release 7.3.1	The OC model for configuring client FEC and Laser Squelch is available. This feature enables you to perform the configuration using scripts, which is less time-consuming. Also, the Open Configuration model supports the use of vendor-neutral data models to configure and manage the network.

Step 1 You can enable FEC (Forward Error Correction) on clients using the following scripts:

```
"openconfig-platform:components": {
 "component": [
  {
   "name": "0/0-Optics0/0/0/2",
   "config": {
    "name": "0/0-Optics0/0/0/2"
   },
   "openconfig-platform-transceiver:transceiver": {
    "config": {
     "fec-mode": "openconfig-platform-types:FEC_ENABLED"
    }
   }
  }
```

Step 2 You can get operational data using GNMI.

```
"state": {
    "connector-type": "openconfig-transport-types:LC_CONNECTOR",
    "date-code": "2019-08-05T00:00:00Z+00:00",
    "fault-condition": false,
    "fec-mode": "openconfig-platform-types:FEC_ENABLED",
    "fec-uncorrectable-words": 0,
    "form-factor": "openconfig-transport-types:QSFP28",
    "otn-compliance-code": "openconfig-transport-types:OTN_UNDEFINED",
    "present": "PRESENT",
    "serial-no": "INL23321878",
    "sonet-sdh-compliance-code": "openconfig-transport-types:SONET_UNDEFINED",
    "vendor": "CISCO-INNOLIGHT",
    "vendor-part": "10-3220-02",
    "vendor-rev": "1C"
}
```

Open Configuration Model for Laser-Squelch

Step 1 You can enable laser-squelch using the following scripts:

```
"openconfig-terminal-device:terminal-device": {
 "logical-channels": {
  "channel": [
   {
    "index": 30002,
    "config": {
     "index": 30002,
     "rate-class": "openconfig-transport-types:TRIB_RATE_100G",
     "admin-state": "ENABLED",
     "description": "ETH Logical Channel
                         ",
     "loopback-mode": "NONE",
     "trib-protocol": "openconfig-transport-types:PROT_100G_MLG",
     "logical-channel-type": "openconfig-transport-types:PROT_ETHERNET"
    },
    "ethernet": {
     "config": {
      "als-delay": 3000,
      "client-als": "LASER_SHUTDOWN"
      }
     },
     "ingress": {
      "config": {
       "transceiver": "0/0-Optics0/0/0/2"
      }
     },
     "logical-channel-assignments": {
      "assignment": [
       {
        "index": 1,
        "config": {
         "index": 1,
         "allocation": "100",
         "assignment-type": "LOGICAL_CHANNEL",
         "description": "ETH to ODU4 assignemnt
                                ",
         "logical-channel": 30020
        }
       }
      ]
     }
    },
```

Step 2 You can get operational data using GNMI.

```
"state": {
                                 "als-delay": 5,
                                  "client-als": "LASER_SHUTDOWN",
                                  "in-crc-errors": 18446744073709551473,
                                  "in-fragment-frames": 0,
                                 "in-jabber-frames": 2,
                                  "in-mac-pause-frames": 0,
                                  "in-oversize-frames": 0,
                                  "in-pcs-bip-errors": 0,
                                  "in-pcs-errored-seconds": 0,
                                 "in-pcs-severely-errored-seconds": 0,
                                  "in-pcs-unavailable-seconds": 10,
                                  "out-mac-pause-frames": 0
                              }
                        },
```

Configure the Line Card in Regen Mode

Table 4: Feature History

Feature Name	Release	Description
OC (Open Configuration) Support for Regen option	Cisco IOS XR Release 7.3.1	The OC support for configuring the Line Card in Regen mode is available. This enables you to perform the configuration using scripts, which is less time-consuming. Also, the Open Configuration model supports the use of vendor-neutral data models to configure and manage the network.

Procedure

	Command or Action	Purpose
Step 1	You can configure Regen mode for the Line Card on slot 1 using the following scripts:	(see script below)

```
{

"openconfig-terminal-device:terminal-device": {

                              "logical-channels":
 {

"channel": [{

                          "index": 10000,

                          "config": {

                                     "index":
 10000,

"admin-state": "ENABLED",

"description": "Coherent Logical Channel",

"logical-channel-type":
"openconfig-transport-types:PROT_OTN"

                          },
```

	Command or Action	Purpose
		```"logical-channel-assignments": {``` ```"assignment": [{``` ```"index": 1,``` ```"config": {``` ```"index": 1,``` ```"allocation": 300,``` ```"assignment-type": "OPTICAL_CHANNEL",``` ```"description": "Coherent to optical assignemnt",``` ```"optical-channel": "0/1-OpticalChannel0/1/0/0"``` ```}``` ```},``` ```{``` ```"index": 2,``` ```"config": {```

	Command or Action	Purpose
		`"index": 2,` `"allocation": 300,` `"assignment-type": "LOGICAL_CHANNEL",` `"description": "Coherent to optical assignemnt index junk",` `"logical-channel": 10001` `}` `}` `]` `}` `},` `{` `"index": 10001,` `"config": {` `"index": 10001,` `"admin-state": "ENABLED",` `"description": "Coherent Logical Channel",`

	Command or Action	Purpose
		` "logical-channel-type": "openconfig-transport-types:PROT_OTN" }, "logical-channel-assignments": { "assignment": [{ "index": 1, "config": { "index": 1, "allocation": 300, "assignment-type": "OPTICAL_CHANNEL", "description": "Coherent to optical assignemnt", "optical-channel": "0/1-OpticalChannel0/1/0/1" } }, { "index": 2,`

	Command or Action	Purpose
		```
		"config": {
		"index": 2,
		"allocation": 300,
		"assignment-type": "LOGICAL_CHANNEL",
		"description": "Coherent to optical assignemnt",
		"logical-channel": 10000
		}
		}
		]
		}
		}
		]
		}
		},
		"openconfig-platform:components": {
		"component": [{
		"name": "0/1-OpticalChannel0/1/0/0",
		```

	Command or Action	Purpose

		`"openconfig-terminal-device:optical-channel": {`
		`"config": {`
		`"line-port": "0/1-Optics0/1/0/0"`
		`}`
		`}`
		`},`
		`{`
		`"name": "0/1-OpticalChannel0/1/0/1",`
		`"openconfig-terminal-device:optical-channel": {`
		`"config": {`
		`"line-port": "0/1-Optics0/1/0/1"`
		`}`
		`}`
		`}`
		`]`
		`}`
		`}`

Examples Using gRPC

Example—Verify the Slice Configuration Using gRPC

Set-up:

- Client—client_v3
- Client IP address and configured grpc port—1.74.27.25:57500

```
./client_v3 -server 1.74.27.25:57500 -oper show-cmd-text -cli_input_file show-hw-module
```

The slice configuration is displayed.

```
{
  "Response": "{\"ResReqId\":753690684504425618,\"output\":\"\\n--------------------------
 show hw-module slice all ---------------------------\\nSlice ID:               1\\nStatus:
                  Provisioned\\nClient Bitrate:          100\\nTrunk Bitrate:
100\\nDP FPGA Version:         H201 (NEED UPG)\\n\\nClient Port -   Trunk Port\\t
CoherentDSP0/0/0/12\\t CoherentDSP0/0/0/13\\nTraffic Split
Percentage\\n\\nHundredGigECtrlr0/0/0/7  \\t                 100
0\\nHundredGigECtrlr0/0/0/11 \\t                  0                 100\\n\\n\\n\"}",
  "FatalErrors": ""
}
```

Example—View the Optics Controller Configuration Using gRPC and Yang

Set-up:

- Client—client_v3
- Client IP address and configured grpc port—1.74.27.25:57500
- Yang model—Cisco-IOS-XR-ifmgr-cfg

```
./client -server_addr=1.74.27.25:57500 -username=root -password=lab -oper=get-config
-yang_path='{"Cisco-IOS-XR-ifmgr-cfg:interface-configurations": [null]}'
```

The optics controller configuration is displayed.

```
{
 "Cisco-IOS-XR-ifmgr-cfg:interface-configurations": {
  "interface-configuration": [
   {
    "active": "act",
    "interface-name": "Optics0/0/0/5",
    "shutdown": [null]
   },
   {
    "active": "act",
    "interface-name": "Optics0/0/0/6",
    "Cisco-IOS-XR-controller-optics-cfg:optics": {
     "optics-dwdm-carrier": {
      "grid-type": "100mhz-grid",
      "param-type": "frequency",
      "param-value": 1927000
     }
    },
    "secondary-admin-state": "maintenance"
   },
   {
    "active": "act",
    "interface-name": "Optics0/0/0/12",
    "shutdown": [
     null
    ]
   },
   {
    "active": "act",
    "interface-name": "Optics0/0/0/13",
    "Cisco-IOS-XR-controller-optics-cfg:optics": {
     "optics-dwdm-carrier": {
      "grid-type": "100mhz-grid",
      "param-type": "frequency",
      "param-value": 1927000
     }
    },
    "secondary-admin-state": "maintenance"
   },
   {
    "active": "act",
    "interface-name": "Optics0/0/0/14",
    "Cisco-IOS-XR-controller-optics-cfg:optics": {
     "rx-thresholds": {
      "rx-threshold": [
       {
        "rx-threshold-type": "low",
        "rx-threshold": -120
       },
       {
        "rx-threshold-type": "high",
        "rx-threshold": 49
       }]}}}
    ,
   {
    "active": "act",
    "interface-name": "Optics0/0/0/18",
    "Cisco-IOS-XR-controller-optics-cfg:optics": {
     "rx-thresholds": {
      "rx-threshold": [
       {
        "rx-threshold-type": "low",
        "rx-threshold": -120
       },
       {
        "rx-threshold-type": "high",
        "rx-threshold": 49
       }]}}}
      ,
   {
    "active": "act",
    "interface-name": "Optics0/0/0/19",
    "shutdown": [
     null
    ],
    "Cisco-IOS-XR-controller-optics-cfg:optics": {
     "optics-dwdm-carrier": {
      "grid-type": "50g-hz-grid",
      "param-type": "frequency",
      "param-value": 19270
     }}}
   ,
   {
    "active": "act",
    "interface-name": "Optics0/0/0/20",
    "Cisco-IOS-XR-controller-optics-cfg:optics": {
     "optics-dwdm-carrier": {
      "grid-type": "50g-hz-grid",
      "param-type": "frequency",
      "param-value": 19270
     },
     "rx-thresholds": {
      "rx-threshold": [
       {
        "rx-threshold-type": "low",
        "rx-threshold": -120
       },
      {
       "rx-threshold-type": "high",
       "rx-threshold": 49
      }]}}}
     ],
   {
    "active": "act",
    "interface-name": "Optics0/0/0/26",
    "shutdown": [
     null
    ]
   },
   {
    "active": "act",
    "interface-name": "Optics0/0/0/27",
    "shutdown": [
     null
    ]
   },
   {
    "active": "act",
    "interface-name": "MgmtEth0/RP0/CPU0/0",
    "Cisco-IOS-XR-ipv4-io-cfg:ipv4-network": {
     "addresses": {
      "primary": {
       "address": "10.77.132.165",
       "netmask": "255.255.255.0"
      }}}}
     ,
   {
    "active": "act",
    "interface-name": "TenGigECtrlr0/0/0/0/1",
    "Cisco-IOS-XR-pmengine-cfg:performance-management": {
     "ethernet-minute15": {
      "minute15-ether": {
       "minute15-ether-reports": {
        "minute15-ether-report": [
         {
          "ether-report": "report-fcs-err"
         }
        ]
       },
       "minute15-ether-thresholds": {
        "minute15-ether-threshold": [
         {
          "ether-threshold": "thresh-fcs-err",
          "ether-threshold-value": 1000
         }
        ]
       }
      }
     }
    }
   },
   {
    "active": "act",
    "interface-name": "TenGigECtrlr0/0/0/0/2",
    "Cisco-IOS-XR-pmengine-cfg:performance-management": {
     "ethernet-minute15": {
      "minute15-ether": {
       "minute15-ether-reports": {
        "minute15-ether-report": [
         {
          "ether-report": "report-fcs-err"
      }
     ]
    },
    "minute15-ether-thresholds": {
     "minute15-ether-threshold": [
      {
       "ether-threshold": "thresh-fcs-err",
       "ether-threshold-value": 1000
      }
     ]
    }
   }
  }
 }
},
{
 "active": "act",
 "interface-name": "TenGigECtrlr0/0/0/0/3",
 "Cisco-IOS-XR-pmengine-cfg:performance-management": {
  "ethernet-minute15": {
   "minute15-ether": {
    "minute15-ether-reports": {
     "minute15-ether-report": [
      {
       "ether-report": "report-fcs-err"
      }
     ]
    },
    "minute15-ether-thresholds": {
     "minute15-ether-threshold": [
      {
       "ether-threshold": "thresh-fcs-err",
       "ether-threshold-value": 1000
      }
     ]
    }
   }
  }
 }
},
{
 "active": "act",
 "interface-name": "TenGigECtrlr0/0/0/0/4",
 "Cisco-IOS-XR-pmengine-cfg:performance-management": {
  "ethernet-minute15": {
   "minute15-ether": {
    "minute15-ether-reports": {
     "minute15-ether-report": [
      {
       "ether-report": "report-fcs-err"
      }
     ]
    },
    "minute15-ether-thresholds": {
     "minute15-ether-threshold": [
      {
       "ether-threshold": "thresh-fcs-err",
       "ether-threshold-value": 1000
      }
     ]
    }
   }
  }
 }
```

```
},
{
 "active": "act",
 "interface-name": "TenGigECtrlr0/0/0/11/1",
 "Cisco-IOS-XR-pmengine-cfg:performance-management": {
  "ethernet-minute15": {
   "minute15-ether": {
    "minute15-ether-reports": {
     "minute15-ether-report": [
      {
       "ether-report": "report-fcs-err"
      }
     ]
    },
    "minute15-ether-thresholds": {
     "minute15-ether-threshold": [
      {
       "ether-threshold": "thresh-fcs-err",
       "ether-threshold-value": 1000
      }
     ]
    }
   }
  }
 }
},
{
 "active": "act",
 "interface-name": "TenGigECtrlr0/0/0/11/2",
 "Cisco-IOS-XR-pmengine-cfg:performance-management": {
  "ethernet-minute15": {
   "minute15-ether": {
    "minute15-ether-reports": {
     "minute15-ether-report": [
      {
       "ether-report": "report-fcs-err"
      }
     ]
    },
    "minute15-ether-thresholds": {
     "minute15-ether-threshold": [
      {
       "ether-threshold": "thresh-fcs-err",
       "ether-threshold-value": 1000
      }
     ]
    }
   }
  }
 }
},
{
 "active": "act",
 "interface-name": "TenGigECtrlr0/0/0/11/3",
 "Cisco-IOS-XR-pmengine-cfg:performance-management": {
  "ethernet-minute15": {
   "minute15-ether": {
    "minute15-ether-reports": {
     "minute15-ether-report": [
      {
       "ether-report": "report-fcs-err"
      }
     ]
    },
        "minute15-ether-thresholds": {
         "minute15-ether-threshold": [
          {
           "ether-threshold": "thresh-fcs-err",
           "ether-threshold-value": 1000
          }
         ]
        }
       }
      }
     }
    },
    {
     "active": "act",
     "interface-name": "TenGigECtrlr0/0/0/11/4",
     "Cisco-IOS-XR-pmengine-cfg:performance-management": {
      "ethernet-minute15": {
       "minute15-ether": {
        "minute15-ether-reports": {
         "minute15-ether-report": [
          {
           "ether-report": "report-fcs-err"
          }
         ]
        },
        "minute15-ether-thresholds": {
         "minute15-ether-threshold": [
          {
           "ether-threshold": "thresh-fcs-err",
           "ether-threshold-value": 1000
          }
         ]
        }
       }
      }
     }
    }
   ]
  }
}
emsGetConfig: ReqId 1, byteRecv: 7455

----------------- gRPC Summary ----------------------

Operation: get-config
Number of iterations: 1
Total bytes transferred: 7455
Number of bytes per second: 124482
Ave elapsed time in seconds: 0.059888
Min elapsed time in seconds: 0.059888
Max elapsed time in seconds: 0.059888

--------------- End gRPC Summary ---------------------
```