NETCONF Protocol

- Restrictions for the NETCONF Protocol, on page 1
- Information About the NETCONF Protocol, on page 1
- How to Configure the NETCONF Protocol, on page 4
- Verifying the NETCONF Protocol Configuration, on page 7
- Additional References for NETCONF Protocol, on page 10
- Feature Information for NETCONF Protocol, on page 10

Restrictions for the NETCONF Protocol

The NETCONF feature is not supported on a device running dual IOSd configuration or software redundancy.

Information About the NETCONF Protocol

Introduction to Data Models - Programmatic and Standards-Based Configuration

The traditional way of managing network devices is by using Command Line Interfaces (CLIs) for configurational (configuration commands) and operational data (show commands). For network management, Simple Network Management Protocol (SNMP) is widely used, especially for exchanging management information between various network devices. Although CLIs and SNMP are heavily used, they have several restrictions. CLIs are highly proprietary, and human intervention is required to understand and interpret their text-based specification. SNMP does not distinguish between configurational and operational data.

The solution lies in adopting a programmatic and standards-based way of writing configurations to any network device, replacing the process of manual configuration. Network devices running on Cisco IOS XE support the automation of configuration for multiple devices across the network using data models. Data models are developed in a standard, industry-defined language, that can define configuration and state information of a network.

Cisco IOS XE supports the Yet Another Next Generation (YANG) data modeling language. YANG can be used with the Network Configuration Protocol (NETCONF) to provide the desired solution of automated and programmable network operations. NETCONF (RFC 6241) is an XML-based protocol that client applications...
use to request information from and make configuration changes to the device. YANG is primarily used to model the configuration and state data used by NETCONF operations.

In Cisco IOS XE, model-based interfaces interoperate with existing device CLI, Syslog, and SNMP interfaces. These interfaces are optionally exposed northbound from network devices. YANG is used to model each protocol based on RFC 6020.

---

**Note**

To access Cisco YANG models in a developer-friendly way, clone the [GitHub repository](https://github.com), and navigate to the `vendor/cisco` subdirectory. Models for various releases of IOS-XE, IOS-XR, and NX-OS platforms are available here.

---

**NETCONF**

NETCONF provides a mechanism to install, manipulate, and delete the configuration of network devices. It uses an Extensible Markup Language (XML)-based data encoding for the configuration data as well as the protocol messages.

NETCONF uses a simple Remote Procedure Call (RPC) based mechanism to facilitate communication between a client and a server. The client can be a script or application running as part of a network manager. The server is typically a network device (switch or router). It uses Secure Shell (SSH) as the transport layer across network devices. It uses SSH port number 830 as the default port. The port number is a configurable option.

NETCONF also supports capability discovery and model downloads. Supported models are discovered using the `ietf-netconf-monitoring` model. Revision dates for each model are shown in the capabilities response. Data models are available for optional download from a device using the `get-schema` RPC. You can use these YANG models to understand or export the data model. For more details on NETCONF, see [RFC 6241](https://tools.ietf.org/html/rfc6241).

In releases prior to Cisco IOS XE Fuji 16.8.1, an operational data manager (based on polling) was enabled separately. In Cisco IOS XE Fuji 16.8.1 and later releases, operational data works on platforms running NETCONF (similar to how configuration data works), and is enabled by default. For more information on the components that are enabled for operational data queries or streaming, see the GitHub repository, to view `*-oper` in the naming convention.

**NETCONF RESTCONF IPv6 Support**

Data model interfaces (DMIs) support the use of IPv6 protocol. DMI IPv6 support helps client applications to communicate with services that use IPv6 addresses. External facing interfaces will provide dual-stack support; both IPv4 and IPv6.

DMIs are a set of services that facilitate the management of network elements. Application layer protocols such as, NETCONF and RESTCONF access these DMIs over a network.

If IPv6 addresses are not configured, external-facing applications will continue to listen on IPv6 sockets; but these sockets will be unreachable.

**NETCONF Global Session Lock**

The NETCONF protocol provides a set of operations to manage device configurations and retrieve device state information. NETCONF supports a global lock, and the ability to kill non-responsive sessions are introduced in NETCONF.
To ensure consistency and prevent conflicting configurations through multiple simultaneous sessions, the owner of the session can lock the NETCONF session. The NETCONF lock RPC locks the configuration parser and the running configuration database. All other NETCONF sessions (that do not own the lock) cannot perform edit operations; but can perform read operations. These locks are intended to be short-lived and allow the owner to make changes without interaction with other NETCONF clients, non-NETCONF clients (such as, SNMP and CLI scripts), and human users.

A global lock held by an active session is revoked when the associated session is killed. The lock gives the session holding the lock exclusive write access to the configuration. When a configuration change is denied due to a global lock, the error message will specify that a NETCONF global lock is the reason the configuration change has been denied.

The <lock> operation takes a mandatory parameter, <target> that is the name of the configuration datastore that is to be locked. When a lock is active, the <edit-config> and <copy-config> operations are not allowed.

If the clear configuration lock command is specified while a NETCONF global lock is being held, a full synchronization of the configuration is scheduled and a warning syslog message is produced. This command clears only the parser configuration lock.

The following is a sample RPC that shows the <lock> operation:

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

## NETCONF Kill Session

During a session conflict or client misuse of the global lock, NETCONF sessions can be monitored via the show netconf-yang sessions command, and non-responsive sessions can be cleared using the clear netconf-yang session command. The clear netconf-yang session command clears both the NETCONF lock and the configuration lock.

A <kill-session> request will force a NETCONF session to terminate. When a NETCONF entity receives a <kill-session> request for an open session, it stops all operations in process, releases all locks and resources associated with the session, and closes any associated connections.

A <kill-session> request requires the session-ID of the NETCONF session that is to be terminated. If the value of the session-ID is equal to the current session ID, an invalid-value error is returned. If a NETCONF session is terminated while its transaction is still in progress, the data model infrastructure will request a rollback, apply it to the network element, and trigger a synchronization of all YANG models.

If a session kill fails, and a global lock is held, enter the clear configuration lock command via the console or vty. At this point, the data models can be stopped and restarted.
How to Configure the NETCONF Protocol

NETCONF-YANG uses the primary trustpoint of a device. If a trustpoint does not exist, when NETCONF-YANG is configured, it creates a self-signed trustpoint. For more information, see the Public Key Infrastructure Configuration Guide, Cisco IOS XE Gibraltar 16.10.x.

Providing Privilege Access to Use NETCONF

To start working with NETCONF APIs, you must be a user with privilege level 15. To provide this, perform the following task:

SUMMARY STEPS

1. enable
2. configure terminal
3. username name privilege level password password
4. aaa authentication login default local and aaa authorization exec default local
5. end

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> enable</td>
<td>Enables privileged EXEC mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> Device# enable</td>
<td>Enter your password if prompted.</td>
</tr>
<tr>
<td><strong>Step 2</strong> configure terminal</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> Device# configure terminal</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> username name privilege level password password</td>
<td>Establishes a user name-based authentication system. Configure the following keywords:</td>
</tr>
</tbody>
</table>
| **Example:** Device(config)# username example-name privilege 15 password example_password | • privilege level —Sets the privilege level for the user. For the programmability feature, it must be 15.  
• password password—Sets a password to access the CLI view. |
| **Step 4** aaa authentication login default local and aaa authorization exec default local | [Optional] If aaa new-model is configured, AAA authentication and authorization is required. For a remote AAA server, replace local with your AAA server. |
| **Example:** Device (config)# aaa authentication login default local  
Device (config)# aaa authorization exec default local | |
### Summary Steps

1. `enable`
2. `configure terminal`
3. `netconf-yang`
4. `netconf-yang feature candidate-datastore`
5. `exit`

### Detailed Steps

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| 1    | `enable`         | Enables privileged EXEC mode.  
* Example:  
  Device> enable |  
| 2    | `configure terminal` | Enters global configuration mode.  
* Example:  
  Device# configure terminal |  
| 3    | `netconf-yang`   | Enables the NETCONF interface on your network device.  
* Example:  
  Device (config)# netconf-yang |  
|      |                  | **Note**: After the initial enablement through the CLI, network devices can be managed subsequently through a model based interface. The complete activation of model-based interface processes may require up to 90 seconds. |
| 4    | `netconf-yang feature candidate-datastore` | Enables candidate datastore.  
* Example:  
  Device(config)# netconf-yang feature candidate-datastore |  
| 5    | `exit`           | Exits global configuration mode.  
* Example:  
  Device (config)# exit |  

---

### Configuring NETCONF-YANG

If the legacy NETCONF protocol is enabled on your device, the RFC-compliant NETCONF protocol does not work. Disable the legacy NETCONF protocol by using the `no netconf legacy` command.
Configuring NETCONF Options

Configuring SNMP

Enable the SNMP Server in IOS to enable NETCONF to access SNMP MIB data using YANG models generated from supported MIBs, and to enable supported SNMP traps in IOS to receive NETCONF notifications from the supported traps.

Perform the following steps:

SUMMARY STEPS

1. Enable SNMP features in IOS.
2. After NETCONF-YANG starts, enable SNMP Trap support by sending the following RPC <edit-config> message to the NETCONF-YANG port.
3. Send the following RPC message to the NETCONF-YANG port to save the running configuration to the startup configuration.

DETAILED STEPS

Step 1  Enable SNMP features in IOS.

Example:

```
configure terminal
logging history debugging
logging snmp-trap emergencies
logging snmp-trap alerts
logging snmp-trap critical
logging snmp-trap errors
logging snmp-trap warnings
logging snmp-trap notifications
logging snmp-trap informational
logging snmp-trap debugging
!
snmp-server community public RW
snmp-server trap link ietf
snmp-server enable traps snmp authentication linkdown linkup
snmp-server enable traps syslog
snmp-server manager
exit
```

Step 2  After NETCONF-YANG starts, enable SNMP Trap support by sending the following RPC <edit-config> message to the NETCONF-YANG port.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <netconf-yang xmlns="http://cisco.com/yang/cisco-self-mgmt">
        <cisco-ia xmlns="http://cisco.com/yang/cisco-ia">
          <snmp-trap-control>
            <trap-list>
            </trap-list>
          </snmp-trap-control>
        </cisco-ia>
      </netconf-yang>
    </config>
  </edit-config>
</rpc>
```
Step 3  Send the following RPC message to the NETCONF-YANG port to save the running configuration to the startup configuration.

Example:
```xml
<?xml version="1.0" encoding="utf-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="">
</rpc>
```

Verifying the NETCONF Protocol Configuration

Use the following commands to verify your NETCONF configuration.

**SUMMARY STEPS**

1. show netconf-yang datastores
2. show netconf-yang sessions
3. show netconf-yang sessions detail
4. show netconf-yang statistics
5. show platform software yang-management process

**DETAILED STEPS**

**Step 1  show netconf-yang datastores**

Displays information about NETCONF-YANG datastores.

Example:
```
Device# show netconf-yang datastores
Device# show netconf-yang datastores
Datastore Name : running
Globally Locked By Session : 42
Globally Locked Time : 2018-01-15T14:25:14-05:00
```

**Step 2  show netconf-yang sessions**
Displays information about NETCONF-YANG sessions.

**Example:**

Device# show netconf-yang sessions

R: Global-lock on running datastore
C: Global-lock on candidate datastore
S: Global-lock on startup datastore
Number of sessions : 10

<table>
<thead>
<tr>
<th>session-id</th>
<th>transport</th>
<th>username</th>
<th>source-host</th>
<th>global-lock</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>42</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>44</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>46</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>48</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>50</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>52</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>54</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>56</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
<tr>
<td>58</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>10.85.70.224</td>
<td>None</td>
</tr>
</tbody>
</table>

**Step 3**

**show netconf-yang sessions detail**

Displays detailed information about NETCONF-YANG sessions.

**Example:**

Device# show netconf-yang sessions detail

R: Global-lock on running datastore
C: Global-lock on candidate datastore
S: Global-lock on startup datastore

Number of sessions : 1

<table>
<thead>
<tr>
<th>session-id</th>
<th>transport</th>
<th>username</th>
<th>source-host</th>
<th>login-time</th>
<th>in-rpcs</th>
<th>in-bad-rpcs</th>
<th>out-rpc-errors</th>
<th>out-notifications</th>
<th>global-lock</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>netconf-ssh</td>
<td>admin</td>
<td>2001:db8::1</td>
<td>2018-10-26T12:37:22+00:00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>None</td>
</tr>
</tbody>
</table>

**Step 4**

**show netconf-yang statistics**

Displays information about NETCONF-YANG statistics.

**Example:**

Device# show netconf-yang statistics

netconf-start-time : 2018-01-15T12:51:14-05:00
in-rpcs : 0
in-bad-rpcs : 0
out-rpc-errors : 0
out-notifications : 0
in-sessions : 10
dropped-sessions : 0
Step 5  show platform software yang-management process

Displays the status of the software processes required to support NETCONF-YANG.

Example:

```
Device# show platform software yang-management process

confd : Running
nesd  : Running
syncfd: Running
ncsshd: Running
dmiauthd: Running
vtyserverutil: Running
opdatamgrd: Running
nginx : Running
ndbmand: Running
```

Note  The process nginx runs if `ip http secure-server` or `ip http server` is configured on the device. This process is not required to be in the `running` state for NETCONF to function properly. However, the nginx process is required for RESTCONF.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>confd</td>
<td>Configuration daemon</td>
</tr>
<tr>
<td>nesd</td>
<td>Network element synchronizer daemon</td>
</tr>
<tr>
<td>syncfd</td>
<td>Sync from daemon</td>
</tr>
<tr>
<td>ncsshd</td>
<td>NETCONF Secure Shell (SSH) daemon</td>
</tr>
<tr>
<td>dmiauthd</td>
<td>Device management interface (DMI) authentication daemon</td>
</tr>
<tr>
<td>vtyserverutil</td>
<td>VTY server util daemon</td>
</tr>
<tr>
<td>opdatamgrd</td>
<td>Operational Data Manager daemon</td>
</tr>
<tr>
<td>nginx</td>
<td>NGINX web server</td>
</tr>
<tr>
<td>ndbmand</td>
<td>NETCONF database manager</td>
</tr>
</tbody>
</table>
Additional References for NETCONF Protocol

Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>YANG data models for various release of IOS-XE, IOS-XR, and NX-OS platforms</td>
<td>To access Cisco YANG models in a developer-friendly way, please clone the GitHub repository, and navigate to the vendor/cisco subdirectory. Models for various releases of IOS-XE, IOS-XR, and NX-OS platforms are available here.</td>
</tr>
</tbody>
</table>

Standards and RFCs

<table>
<thead>
<tr>
<th>Standard/RFC</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 6020</td>
<td>YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)</td>
</tr>
<tr>
<td>RFC 6241</td>
<td>Network Configuration Protocol (NETCONF)</td>
</tr>
<tr>
<td>RFC 6536</td>
<td>Network Configuration Protocol (NETCONF) Access Control Model</td>
</tr>
<tr>
<td>RFC 8040</td>
<td>RESTCONF Protocol</td>
</tr>
</tbody>
</table>

Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.</td>
<td><a href="http://www.cisco.com/support">http://www.cisco.com/support</a></td>
</tr>
</tbody>
</table>

Feature Information for NETCONF Protocol

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to [www.cisco.com/go/cfh](http://www.cisco.com/go/cfh). An account on Cisco.com is not required.
Table 2: Feature Information for NETCONF Protocol

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Release</th>
<th>Feature Information</th>
</tr>
</thead>
</table>
| Candidate Config Support | Cisco IOS XE Fuji 16.9.1 | The Candidate Config Support feature enables support for candidate capability by implementing RFC 6241 with a simple commit option. This feature was implemented on the following platforms:  
  - Cisco ASR 1000 Series Aggregated Services Routers  
  - Cisco ASR 900 Series Aggregated Services Routers  
  - Cisco Catalyst 3650 Series Switches  
  - Cisco Catalyst 3850 Series Switches  
  - Cisco Catalyst 9300 Series Switches  
  - Cisco Catalyst 9400 Series Switches  
  - Cisco Catalyst 9500 Series Switches  
  - Cisco CBR-8 Series Routers  
  - Cisco Cloud Services Router 1000V Series  
  - Cisco ISR 4000 Series Integrated Services Routers  
  The following command was introduced: `netconf-yang feature candidate-datastore`.
<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Release</th>
<th>Feature Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>NETCONF Protocol</td>
<td>Cisco IOS XE Denali 16.3.1</td>
<td>The NETCONF Protocol feature facilitates a programmatic and standards-based way of writing configurations and reading operational data from network devices. The following command was introduced: <code>netconf-yang</code>.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS XE Everest 16.5.1a</td>
<td>This feature was implemented on Cisco Catalyst 9300 Series Switches and Cisco Catalyst 9500 Series Switches.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS XE Everest 16.6.2</td>
<td>This feature was implemented on Cisco Catalyst 9400 Series Switches.</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS XE Fuji 16.7.1</td>
<td>This feature was implemented on the following platforms:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ASR 900 Series Aggregated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ASR 920 Series Aggregated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Network Convergence System 4200 Series</td>
</tr>
<tr>
<td></td>
<td>Cisco IOS XE Fuji 16.8.1a</td>
<td>This feature was implemented on Cisco Catalyst 9500-High Performance Series Switches.</td>
</tr>
<tr>
<td>Feature Name</td>
<td>Release</td>
<td>Feature Information</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NETCONF and RESTCONF IPv6 Support</td>
<td>Cisco IOS XE Fuji 16.8.1a</td>
<td>This feature was implemented on the following platforms:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ASR 1000 Series Aggregated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ASR 900 Series Aggregated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 3650 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 3850 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 9300 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 9400 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 9500 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco CBR-8 Series Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco CSR 1000v Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ISR 1100 Series Integrated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ISR 4000 Series Integrated Services Routers</td>
</tr>
<tr>
<td>NETCONF Global Lock and Kill Session</td>
<td>Cisco IOS XE Fuji 16.8.1a</td>
<td>This feature was implemented on the following platforms:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ASR 1000 Series Aggregated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ASR 900 Series Aggregated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 3650 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 3850 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 9300 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 9400 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco Catalyst 9500 Series Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco CBR-8 Series Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco CSR 1000v Switches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ISR 1100 Series Integrated Services Routers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Cisco ISR 4000 Series Integrated Services Routers</td>
</tr>
</tbody>
</table>