NETCONF プロトコル

NETCONF プロトコルの概要

データモデルの概要：プログラムによる設定と各種の標準規格に準拠した設定

ネットワーク デバイスを管理する従来の方法は、階層的データ（設定コマンド）および運用データ（show コマンド）用のコマンドライン インターフェイス（CLI）を使用することです。ネットワーク管理の場合、特にさまざまなネットワーク デバイス間で管理情報を交換するために、Simple Network Management Protocol（SNMP）が広く使用されています。頻繁に使用されている CLI と SNMP ですが、これにはいくつかの制約事項があります。CLI は非常に独自的であり、テキストベースの仕様を理解し、解釈するには人間の介入が必要です。SNMP は、階層的データと運用データを区別しません。

これを解決するには、手作業で設定作業を行うのではなく、プログラムを使用したり、各種の標準規格に準拠してネットワーク デバイスの設定を記述します。Cisco IOS XE で動作するネットワーク デバイスは、データ モデルを使用するネットワーク上の複数のデバイスの設定の自動化をサポートしています。データ モデルは、業界で定義された標準的な言語で開発され、ネットワークの設定とステータス情報を定義できます。

Cisco IOS XE は、Yet Another Next Generation（YANG）データ モデリング言語をサポートしています。YANG をネットワーク設定プロトコル（NETCONF）で使用すると、自動化されたプログラミング可能なネットワーク操作の望ましいソリューションが実現します。NETCONF（RFC 6241）は、クライアント アプリケーションがデバイスからの情報を要求してデバイス

に設定変更を加えるために使用するXMLベースのプロトコルです。YANGは主に、NETCONF操作で使用される設定とステート データをモデル化するために使用されます。

Cisco IOS XE では、モデル ベースのインターフェイスは、既存のデバイス CLI、Syslog、およびSNMPインターフェイスと相互運用します。必要に応じて、これらのインターフェイスは、ネットワーク デバイスからノースバウンドに公開されます。YANG は、RFC 6020 に基づいて各プロトコルをモデル化するために使用されます。

（注） 開発者に分かりやすい方法でCisco YANGモデルにアクセスするには、GitHubリポジトリを複製し、vendor/ciscoサブディレクトリに移動します。ここでは、IOS XE、IOS-XR、およびNX-OSプラットフォームのさまざまなリリースのモデルを使用できます。

NETCONF

NETCONFは、ネットワーク デバイスの設定をインストール、操作、削除するためのメカニズムです。

コンフィギュレーション データとプロトコル メッセージにExtensible Markup Language（XML）ベースのデータ符号化を使用します。

NETCONFはシンプルなリモート プロシージャ コール（RPC）ベースのメカニズムを使用してクライアントとサーバ間の通信を促進します。クライアントはネットワーク マネージャの一部として実行されているスクリプトやアプリケーションです。通常、サーバはネットワーク デバイス（スイッチまたはルータ）です。サーバは、ネットワーク デバイス全体のトランスポート層としてセキュア シェル（SSH）を使用します。SSHポート番号830をデフォルトのポートとして使用します。ポート番号は、設定可能なオプションです。

NETCONFは、機能の検出およびモデルのダウンロードもサポートしています。サポート対象のモデルは、ietf-netconf-monitoring モデルを使用して検出されます。各モデルに対する改定日付は、機能の応答に示されています。データ モデルは、get-schema RPCを使用して、デバイスからオプションのダウンロードとして入手できます。これらのYANGモデルを使用して、データ モデルを理解したりエクスポートしたりできます。NETCONFの詳細については、RFC 6241を参照してください。

Cisco IOS XE Fuji 16.8.1 よりも前のリリースでは、運用データ マネージャ（ポーリングに基づく）が個別に有効になっていました。Cisco IOS XE Fuji 16.8.1 以降のリリースでは、運用データは、NETCONFを実行しているプラットフォームで動作し（設定データの仕組みと同様）、デフォルトで有効になっています。運用データのクエリまたはストリーミングに対応するコンポーネントの詳細については、GitHub リポジトリで命名規則の *-oper を参照してください。

NETCONF プロトコルの制約事項

- NETCONF 機能は、デュアル IOSd 設定またはソフトウェア冗長性を実行中のデバイスではサポートされていません。

- **no ip pim rp-address** コマンドを使用して NETCONF データストアから RP アドレスを削除すると、パーサーの制限により、データストアに不整合が生じる可能性があります。NETCONF データストアから RP アドレスエントリを削除するには、RPC を使用します。

YANG モデルバージョン 1.1

YANG バージョン 1.1 は、RFC 7950（YANG 1.1 データモデリング言語）で説明されています。YANG バージョン 1.1 は、YANG バージョン 1.0 仕様のあいまいさと欠陥に対処する YANG 言語のメンテナンスリリースです。

YANG バージョン 1.1 の YANG モジュールは、NETCONF hello メッセージの代わりに、*ietf-yang-library* を介してアドバタイズされます。

次の例は、デバイスでサポートされているすべての YANG モジュールのリストを取得する NETCONF <*get*> RPC を示しています。

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
  <get>
    <filter>
      <modules-state xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library"/>
    </filter>
  </get>
</rpc>
```

RPC 応答の出力には、各モジュールが使用する YANG バージョンに関係なく、すべての YANG モジュールのリストが含まれています。

Cisco IOS XE Cupertino 17.7.1 は YANG バージョン 1.0 を使用しますが、GitHub から YANG バージョン 1.1 もダウンロードできます（https://github.com/YangModels/yang/tree/master/vendor/cisco/xe）。

もしくは、NETCONF *get-schema* 操作を使用してデバイスから YANG モデルをダウンロードし、*migrate_yang_version.py* スクリプトを使用して、ダウンロードしたモデルをこのバージョンに移行できます。

次の例は、スクリプトを使用して YANG バージョン 1.0 から YANG バージョン 1.1 に移行する方法を示しています。

```
migrate_yang_version.py [-h] [--out OUT] path
```

help コマンドを使用して、スクリプトで使用可能なオプションを表示します。

```
python migrate_yang_version.py --help
usage: migrate_yang_version.py [-h] [--out OUT] path

positional arguments:
  path        Path to the YANG files

optional arguments:
  -h, --help  show this help message and exit
  --out OUT   Path to the output YANG file
```

次の例は、*out* 引数を使用して、ファイルを元の場所から別のフォルダに移動する方法を示しています。

```
python migrate_yang_version.py --out testdir/outdir testdir/indir
```

上記の例では、*testdir/outdir* は YANG モデルバージョン 1.1 が存在するディレクトリ、またはスクリプトの出力が置かれるディレクトリです。ディレクトリが使用できない場合は、このディレクトリが作成されます。

testdir/indir ディレクトリは、YANG モデルバージョン 1.0 が存在するディレクトリです。これはスクリプトの入力です。

YANG モデルバージョン 1.1 を作成したら、GitHub からダウンロードするか、*migrate_yang_version.py* スクリプトを使用してクライアントアプリケーションでコンパイルし、エンドツーエンドの YANG モデルテストを実行して、Cisco IOS XE デバイスに対して検証できます。

（注） デバイスの YANG モデルは、YANG バージョン 1.0 のままですが、クライアントテストケースの RPC ペイロードを変更する必要はありません。

migrate_yang_version.py スクリプトまたは Cisco IOS XE YANG 移行プロセスに関するお問い合わせは、*xe-yang-migration@cisco.com* にメールをお送りください。

Cisco IOS XE Cupertino 17.8.1 では、YANG バージョン 1.1 を使用しています。YANG バージョン 1.1 とバージョン 1.0 の違いは、https://tools.ietf.org/html/rfc7950#page-10 [英語] に記載されています。

Cisco IOS XE Dublin 17.10.1 の YANG バージョン

Cisco IOS XE Dublin 17.10.1 以降のリリースでは、シスコ定義の YANG モデルは、YANG バージョン 1.1 になっています。YANG バージョン 1.1 は、https://github.com/YangModels/yang/tree/master/vendor/cisco/xe の GitHub からダウンロードできます。

YANG バージョン 1.1 では、NETCONF を使用するクライアント アプリケーションに影響を与える重要な変更は、<*hello*> メッセージ内に示されます。サーバーは RFC 7950 に従って、<*hello*> メッセージ内で YANG 1.1 モジュールを機能としてリッスンするのではなく、*ietf-yang-library* を使用して YANG 1.1 モジュールのサポートをアドバタイズします。サポートされている YANG モジュールのリストを収集する場合、<*hello*> メッセージ内からリストを取得するのではなく、*ietf-yang-library* を使用することを推奨します。

NETCONF RESTCONF IPv6 のサポート

データ モデル インターフェイス（DMI）は IPv6 プロトコルの使用をサポートしています。DMI による IPv6 のサポートは、クライアント アプリケーションが、IPv6 アドレスを使用するサービスと通信する場合に役に立ちます。外部向けインターフェイスは、IPv4 と IPv6 の両方についてデュアルスタックをサポートします。

DMI は、ネットワーク要素の管理を容易にする一連のサービスです。NETCONF や RESTCONF などのアプリケーション層プロトコルは、ネットワークを介してこれらの DMI にアクセスします。

IPv6 アドレスが設定されていない場合でも、外部向けアプリケーションは IPv6 ソケットをリッスンし続けますが、これらのソケットは到達不能になります。

IOS コマンドの XML への変換

Cisco IOS XE Cupertino 17.7.1 以降のリリースでは、IOS コマンドを関連する NETCONF-YANG XML または RESTCONF-JSON 要求メッセージに自動的に変換できます。生成された設定メッセージを分析して、メッセージで使用される Xpath を理解できます。構造化された形式で生成された設定を使用して、ネットワーク内の他のデバイスをプロビジョニングできます。ただし、この設定は変更できません。

IOS コマンドに変換するには、**show running-config | format netconf-xml** コマンドまたは **show running-config | format restconf-json** コマンドを使用します。

netconf-xml キーワードを選択すると、IOS コマンドは NETCONF-YANG XML 形式に変換されます。**restconf-json** キーワードを選択すると、IOS コマンドは RESTCONF-JSON 形式に変換されます。

IOS コマンドの構造化形式への変換は、デフォルトで無効になっています。最初に NETCONF-YANG を設定する必要があります。データ モデル インターフェイス（DMI）が初期化されたら、適切なフォーマット オプションを使用してコマンドを変換します。

次に、**show running-config | format netconf-xml** コマンドの出力例を示します。

```
Device# show running-config | format netconf-xml

<config xmlns="http://tail-f.com/ns/config/1.0">
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
    <version>17.8</version>
    <boot-start-marker/>
    <boot>
      <system>
        <flash>
          <flash-list-ordered-by-user>

<flash-leaf>bootflash:c8000v-universalk9.BLD_POLARIS_DEV_LATEST_20211020_005209.SSA.bin</

              flash-leaf>
          </flash-list-ordered-by-user>
        </flash>
      </system>
    </boot>
    <boot-end-marker/>
    <memory>
      <free>
        <low-watermark>
          <processor>64219</processor>
        </low-watermark>
      </free>
    </memory>
    <call-home>
      <contact-email-addr xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-call-home">
        sch-smart-licensing@cisco.com</contact-email-addr>
```

```
  <tac-profile xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-call-home">
    <profile>
      <CiscoTAC-1>
        <active>true</active>
        <destination>
          <transport-method>http</transport-method>
        </destination>
      </CiscoTAC-1>
    </profile>
  </tac-profile>
</call-home>
<service>
  <timestamps>
    <debug-config>
      <datetime>
        <msec/>
        <localtime/>
        <show-timezone/>
      </datetime>
    </debug-config>
    <log-config>
      <datetime>
        <msec/>
        <localtime/>
        <show-timezone/>
      </datetime>
    </log-config>
  </timestamps>
  <call-home/>
</service>
<platform>
  <console xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-platform">
    <output>serial</output>
  </console>
  <qfp xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-platform">
    <utilization>
      <monitor>
        <load>80</load>
      </monitor>
    </utilization>
  </qfp>
  <punt-keepalive xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-platform">
    <disable-kernel-core>true</disable-kernel-core>
  </punt-keepalive>
</platform>
<hostname>pi-prog-csr1</hostname>
<enable>
  <password>
    <secret>lab</secret>
  </password>
</enable>
<username>
  <name>admin</name>
  <privilege>15</privilege>
  <password>
    <encryption>0</encryption>
    <password>lab</password>
  </password>
</username>
<vrf>
  <definition>
    <name>Mgmt-intf</name>
    <address-family>
      <ipv4>
```

```
      </ipv4>
      <ipv6>
      </ipv6>
    </address-family>
  </definition>
</vrf>
<ip>
  <domain>
    <name>cisco</name>
  </domain>
  <forward-protocol>
    <protocol>nd</protocol>
  </forward-protocol>
  <route>
    <ip-route-interface-forwarding-list>
      <prefix>10.0.0.0</prefix>
      <mask>255.255.0.0</mask>
      <fwd-list>
        <fwd>10.45.0.1</fwd>
      </fwd-list>
    </ip-route-interface-forwarding-list>
    <vrf>
      <name>Mgmt-intf</name>
      <ip-route-interface-forwarding-list>
        <prefix>0.0.0.0</prefix>
        <mask>0.0.0.0</mask>
        <fwd-list>
          <fwd>10.104.54.129</fwd>
        </fwd-list>
      </ip-route-interface-forwarding-list>
    </vrf>
  </route>
  <ssh>
    <ssh-version>2</ssh-version>
  </ssh>
  <tftp>
    <source-interface>
      <GigabitEthernet>1</GigabitEthernet>
    </source-interface>
    <blocksize>8192</blocksize>
  </tftp>
  <http xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-http">
    <authentication>
      <local/>
    </authentication>
    <server>true</server>
    <secure-server>true</secure-server>
  </http>
</ip>
<ipv6>
  <unicast-routing/>
</ipv6>
<interface>
  <GigabitEthernet>
    <name>1</name>
    <vrf>
      <forwarding>Mgmt-intf</forwarding>
    </vrf>
    <ip>
      <address>
        <primary>
          <address>10.104.54.222</address>
          <mask>255.255.255.128</mask>
        </primary>
```

```
      </address>
    </ip>
    <mop>
      <enabled>false</enabled>
      <sysid>false</sysid>
    </mop>
    <negotiation xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-ethernet">
      <auto>true</auto>
    </negotiation>
  </GigabitEthernet>
  <GigabitEthernet>
    <name>2</name>
    <ip>
      <address>
        <primary>
          <address>9.45.21.231</address>
          <mask>255.255.0.0</mask>
        </primary>
      </address>
    </ip>
    <mop>
      <enabled>false</enabled>
      <sysid>false</sysid>
    </mop>
    <negotiation xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-ethernet">
      <auto>true</auto>
    </negotiation>
  </GigabitEthernet>
  <GigabitEthernet>
    <name>3</name>
    <mop>
      <enabled>false</enabled>
      <sysid>false</sysid>
    </mop>
    <negotiation xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-ethernet">
      <auto>true</auto>
    </negotiation>
  </GigabitEthernet>
  <GigabitEthernet>
    <name>4</name>
    <mop>
      <enabled>false</enabled>
      <sysid>false</sysid>
    </mop>
    <negotiation xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-ethernet">
      <auto>true</auto>
    </negotiation>
  </GigabitEthernet>
  <GigabitEthernet>
    <name>5</name>
    <mop>
      <enabled>false</enabled>
      <sysid>false</sysid>
    </mop>
    <negotiation xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-ethernet">
      <auto>true</auto>
    </negotiation>
  </GigabitEthernet>
</interface>
<control-plane>
</control-plane>
<clock>
  <timezone>
    <zone>IST</zone>
```

```
        <hours>5</hours>
        <minutes>30</minutes>
      </timezone>
    </clock>
    <logging>
      <console-config>
        <console>false</console>
      </console-config>
    </logging>
    <aaa>
      <new-model xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-aaa"/>
      <authentication xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-aaa">
        <login>
          <name>default</name>
          <a1>
            <local/>
          </a1>
        </login>
      </authentication>
      <authorization xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-aaa">
        <exec>
          <name>default</name>
          <a1>
            <local/>
          </a1>
        </exec>
      </authorization>
      <common-criteria xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-aaa">
        <policy>enable_secret_policy</policy>
        <char-changes>4</char-changes>
        <lower-case>1</lower-case>
        <max-length>127</max-length>
        <min-length>10</min-length>
        <numeric-count>1</numeric-count>
        <upper-case>1</upper-case>
      </common-criteria>
      <session-id xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-aaa">common</session-id>
    </aaa>
    <login>
      <on-success>
        <log>
        </log>
      </on-success>
    </login>
    <multilink>
      <bundle-name
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-ppp">authenticated</bundle-name>
    </multilink>
    <redundancy>
    </redundancy>
    <spanning-tree>
      <extend xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-spanning-tree">
        <system-id/>
      </extend>
    </spanning-tree>
    <subscriber>
      <templating/>
    </subscriber>
    <crypto>
      <pki xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-crypto">
        <certificate>
          <chain>
            <name>SLA-TrustPoint</name>
            <certificate>
```

```
            <serial>01</serial>
            <certtype>ca</certtype>
          </certificate>
        </chain>
        <chain>
          <name>TP-self-signed-2685563505</name>
          <certificate>
            <serial>01</serial>
            <certtype>self-signed</certtype>
          </certificate>
        </chain>
      </certificate>
      <trustpoint>
        <id>SLA-TrustPoint</id>
        <enrollment>
          <pkcs12/>
        </enrollment>
        <revocation-check>crl</revocation-check>
      </trustpoint>
      <trustpoint>
        <id>TP-self-signed-2685563505</id>
        <enrollment>
          <selfsigned/>
        </enrollment>
        <revocation-check>none</revocation-check>
        <rsakeypair>
          <key-label>TP-self-signed-2685563505</key-label>
        </rsakeypair>
        <subject-name>cn=IOS-Self-Signed-Certificate-2685563505</subject-name>
      </trustpoint>
    </pki>
  </crypto>
  <license>
    <udi>
      <pid>C8000V</pid>
      <sn>93SHKMJKOC6</sn>
    </udi>
    <boot>
      <level>
        <network-advantage>
          <addon>dna-advantage</addon>
        </network-advantage>
      </level>
    </boot>
  </license>
  <line>
    <aux>
      <first>0</first>
    </aux>
    <console>
      <first>0</first>
      <exec-timeout>
        <minutes>0</minutes>
        <seconds>0</seconds>
      </exec-timeout>
      <stopbits>1</stopbits>
    </console>
    <vty>
      <first>0</first>
      <last>4</last>
      <exec-timeout>
        <minutes>0</minutes>
        <seconds>0</seconds>
      </exec-timeout>
```

```
        <password>
          <secret>lab</secret>
        </password>
        <transport>
          <input>
            <all/>
          </input>
          <output>
            <all/>
          </output>
        </transport>
      </vty>
      <vty>
        <first>5</first>
        <last>31</last>
        <transport>
          <input>
            <all/>
          </input>
          <output>
            <all/>
          </output>
        </transport>
      </vty>
    </line>
    <diagnostic xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-diagnostics">
      <bootup>
        <level>minimal</level>
      </bootup>
    </diagnostic>
  </native>
</config>
pi-prog-csr1#
pi-prog-csr1#
pi-prog-csr1#show running-config | format restconf-json
{
  "data": {
    "Cisco-IOS-XE-native:native": {
      "version": "17.8",
      "boot-start-marker": [null],
      "boot": {
        "system": {
          "flash": {
            "flash-list-ordered-by-user": [
              {
                "flash-leaf":
"bootflash:c8000v-universalk9.BLD_POLARIS_DEV_LATEST_20211020_005209.SSA.bin"
              }
            ]
          }
        }
      },
      "boot-end-marker": [null],
      "memory": {
        "free": {
          "low-watermark": {
            "processor": 64219
          }
        }
      },
      "call-home": {
        "Cisco-IOS-XE-call-home:contact-email-addr": "sch-smart-licensing@cisco.com",
        "Cisco-IOS-XE-call-home:tac-profile": {
          "profile": {
```

```
      "CiscoTAC-1": {
        "active": true,
        "destination": {
          "transport-method": "http"
        }
      }
    }
  }
},
"service": {
  "timestamps": {
    "debug-config": {
      "datetime": {
        "msec": [null],
        "localtime": [null],
        "show-timezone": [null]
      }
    },
    "log-config": {
      "datetime": {
        "msec": [null],
        "localtime": [null],
        "show-timezone": [null]
      }
    }
  },
  "call-home": [null]
},
"platform": {
  "Cisco-IOS-XE-platform:console": {
    "output": "serial"
  },
  "Cisco-IOS-XE-platform:qfp": {
    "utilization": {
      "monitor": {
        "load": 80
      }
    }
  },
  "Cisco-IOS-XE-platform:punt-keepalive": {
    "disable-kernel-core": true
  }
},
"hostname": "pi-prog-csr1",
"enable": {
  "password": {
    "secret": "lab"
  }
},
"username": [
  {
    "name": "admin",
    "privilege": 15,
    "password": {
      "encryption": "0",
      "password": "lab"
    }
  }
],
"vrf": {
  "definition": [
    {
      "name": "Mgmt-intf",
      "address-family": {
```

```
        "ipv4": {
        },
        "ipv6": {
        }
      }
    }
  ]
},
"ip": {
  "domain": {
    "name": "cisco"
  },
  "forward-protocol": {
    "protocol": "nd"
  },
  "route": {
    "ip-route-interface-forwarding-list": [
      {
        "prefix": "10].0.0.0",
        "mask": "255.255.0.0",
        "fwd-list": [
          {
            "fwd": "9.45.0.1"
          }
        ]
      }
    ],
    "vrf": [
      {
        "name": "Mgmt-intf",
        "ip-route-interface-forwarding-list": [
          {
            "prefix": "0.0.0.0",
            "mask": "0.0.0.0",
            "fwd-list": [
              {
                "fwd": "10.104.54.129"
              }
            ]
          }
        ]
      }
    ]
  },
  "ssh": {
    "ssh-version": "2"
  },
  "tftp": {
    "source-interface": {
      "GigabitEthernet": "1"
    },
    "blocksize": 8192
  },
  "Cisco-IOS-XE-http:http": {
    "authentication": {
      "local": [null]
    },
    "server": true,
    "secure-server": true
  }
},
"ipv6": {
  "unicast-routing": [null]
},
```

```
"interface": {
  "GigabitEthernet": [
    {
      "name": "1",
      "vrf": {
        "forwarding": "Mgmt-intf"
      },
      "ip": {
        "address": {
          "primary": {
            "address": "10.104.54.222",
            "mask": "255.255.255.128"
          }
        }
      },
      "mop": {
        "enabled": false,
        "sysid": false
      },
      "Cisco-IOS-XE-ethernet:negotiation": {
        "auto": true
      }
    },
    {
      "name": "2",
      "ip": {
        "address": {
          "primary": {
            "address": "10.45.21.231",
            "mask": "255.255.0.0"
          }
        }
      },
      "mop": {
        "enabled": false,
        "sysid": false
      },
      "Cisco-IOS-XE-ethernet:negotiation": {
        "auto": true
      }
    },
    {
      "name": "3",
      "mop": {
        "enabled": false,
        "sysid": false
      },
      "Cisco-IOS-XE-ethernet:negotiation": {
        "auto": true
      }
    },
    {
      "name": "4",
      "mop": {
        "enabled": false,
        "sysid": false
      },
      "Cisco-IOS-XE-ethernet:negotiation": {
        "auto": true
      }
    },
    {
      "name": "5",
      "mop": {
```

```
        "enabled": false,
        "sysid": false
      },
      "Cisco-IOS-XE-ethernet:negotiation": {
        "auto": true
      }
    }
  ]
},
"control-plane": {
},
"clock": {
  "timezone": {
    "zone": "IST",
    "hours": 5,
    "minutes": 30
  }
},
"logging": {
  "console-config": {
    "console": false
  }
},
"aaa": {
  "Cisco-IOS-XE-aaa:new-model": [null],
  "Cisco-IOS-XE-aaa:authentication": {
    "login": [
      {
        "name": "default",
        "a1": {
          "local": [null]
        }
      }
    ]
  },
  "Cisco-IOS-XE-aaa:authorization": {
    "exec": [
      {
        "name": "default",
        "a1": {
          "local": [null]
        }
      }
    ]
  },
  "Cisco-IOS-XE-aaa:common-criteria": [
    {
      "policy": "enable_secret_policy",
      "char-changes": 4,
      "lower-case": 1,
      "max-length": 127,
      "min-length": 10,
      "numeric-count": 1,
      "upper-case": 1
    }
  ],
  "Cisco-IOS-XE-aaa:session-id": "common"
},
"login": {
  "on-success": {
    "log": {
    }
  }
},
```

```
"multilink": {
  "Cisco-IOS-XE-ppp:bundle-name": "authenticated"
},
"redundancy": {
},
"spanning-tree": {
  "Cisco-IOS-XE-spanning-tree:extend": {
    "system-id": [null]
  }
},
"subscriber": {
  "templating": [null]
},
"crypto": {
  "Cisco-IOS-XE-crypto:pki": {
    "certificate": {
      "chain": [
        {
          "name": "SLA-TrustPoint",
          "certificate": [
            {
              "serial": "01",
              "certtype": "ca"
            }
          ]
        },
        {
          "name": "TP-self-signed-2685563505",
          "certificate": [
            {
              "serial": "01",
              "certtype": "self-signed"
            }
          ]
        }
      ]
    },
    "trustpoint": [
      {
        "id": "SLA-TrustPoint",
        "enrollment": {
          "pkcs12": [null]
        },
        "revocation-check": ["crl"]
      },
      {
        "id": "TP-self-signed-2685563505",
        "enrollment": {
          "selfsigned": [null]
        },
        "revocation-check": ["none"],
        "rsakeypair": {
          "key-label": "TP-self-signed-2685563505"
        },
        "subject-name": "cn=IOS-Self-Signed-Certificate-2685563505"
      }
    ]
  }
},
"license": {
  "udi": {
    "pid": "C8000V",
    "sn": "93SHKMJKOC6"
  },
```

```
    "boot": {
      "level": {
        "network-advantage": {
          "addon": "dna-advantage"
        }
      }
    }
  },
  "line": {
    "aux": [
      {
        "first": "0"
      }
    ],
    "console": [
      {
        "first": "0",
        "exec-timeout": {
          "minutes": 0,
          "seconds": 0
        },
        "stopbits": "1"
      }
    ],
    "vty": [
      {
        "first": 0,
        "last": 4,
        "exec-timeout": {
          "minutes": 0,
          "seconds": 0
        },
        "password": {
          "secret": "lab"
        },
        "transport": {
          "input": {
            "all": [null]
          },
          "output": {
            "all": [null]
          }
        }
      },
      {
        "first": 5,
        "last": 31,
        "transport": {
          "input": {
            "all": [null]
          },
          "output": {
            "all": [null]
          }
        }
      }
    ]
  },
  "Cisco-IOS-XE-diagnostics:diagnostic": {
    "bootup": {
      "level": "minimal"
    }
  }
}
```

```
  }
}
```

NETCONF グローバル セッションのロック

NETCONF プロトコルは、デバイス設定を管理し、デバイスの状態情報を取得するための一連の操作を提供します。NETCONF はグローバル ロックをサポートしており、NETCONF では応答しなくなったセッションを kill する機能が導入されています。

複数の同時セッションの全体にわたって一貫性を確保し、設定の競合を防ぐために、セッションのオーナーは NETCONF セッションをロックできます。NETCONF lock RPC は、コンフィギュレーション パーサーと実行コンフィギュレーション データベースをロックします。その他のすべての NETCONF セッション（ロックを所有していない）は、編集操作を実行できません。ただし、読み取り操作は実行できます。これらのロックは存続時間が短いことを意図しており、オーナーは、他の NETCONF クライアント、NETCONF 以外のクライアント（SNMP、CLI スクリプトなど）、および人間のユーザとやり取りをせずに変更を加えることができます。

アクティブセッションによって保持されているグローバル ロックは、関連付けられたセッションが kill されたときに無効になります。ロックによって、ロックを保持しているセッションが、設定に対して排他的な書き込みアクセスを行えるようになります。グローバル ロックにより設定の変更が拒否された場合は、エラー メッセージによって、NETCONF グローバル ロックが原因で設定の変更が拒否されたことが示されます。

<lock> 操作は必須パラメータ <target> を受け取ります。これは、ロックしようとするコンフィギュレーション データストアの名前です。ロックがアクティブな場合、<edit-config> 操作と <copy-config> 操作は許可されません。

NETCONF のグローバル ロックの保持中に **clear configuration lock** コマンドが指定された場合は、設定の完全な同期がスケジュールされ、警告の syslog メッセージが生成されます。このコマンドは、パーサー コンフィギュレーション ロックのみをクリアします。

次に、<lock> 操作を示す RPC の例を示します。

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

NETCONF Kill セッション

セッションの競合時、またはクライアントによるグローバル ロックの誤用が生じたときは、**show netconf-yang sessions** コマンドを使用して NETCONF セッションをモニタできます。また、**clear netconf-yang session** コマンドを使用して応答しなくなったセッションをクリアする

こともできます。**clear netconf-yang session** コマンドは、NETCONF ロックとコンフィギュレーション ロックの両方をクリアします。

<kill-session> 要求は、NETCONF セッションを強制的に終了します。NETCONF エンティティは、オープン セッションの <kill-session> 要求を受信すると、プロセス内のすべての操作を停止し、セッションに関連付けられているすべてのロックとリソースを解放して、関連付けられた接続をすべて閉じます。

<kill-session> 要求には、終了する NETCONF セッションのセッション ID が必要です。セッション ID の値が現在のセッション ID と同じ場合は、無効な値を示すエラーが返されます。NETCONF セッションのトランザクションがまだ進行中に NETCONF セッションが終了した場合は、データ モデル インフラストラクチャによってロールバックが要求され、ネットワーク要素にロールバックが適用されて、すべての YANG モデルの同期がトリガーされます。

セッションの kill が失敗し、グローバル ロックが保持されている場合は、コンソールまたは vty を使用して **clear configuration lock** コマンドを入力します。この時点で、データ モデルを停止して再起動することができます。

NETCONF-YANG SSH サーバのサポート

NETCONF-YANG はパスワードベースの認証に代わる方法として、IOS セキュアシェル（SSH）リベスト、シャミア、エーデマル（RSA）公開キーを使用したユーザの認証をサポートします。

NETCONF-YANG で公開キー認証を機能させるには、IOS SSH サーバを設定する必要があります。SSH サーバに対してユーザを認証するには、**ip ssh pubkey-chain** および **user** コマンドを使用して設定された RSA キーのいずれかを使用します。

NACM は、グループベースのアクセス制御メカニズムです。ユーザが認証されると、設定された権限レベルに基づいて、NACM 権限グループに自動的に配置されます。ユーザを他のユーザ定義グループに手動で配置することもできます。デフォルトの特権レベルは 1 です。PRIV00 ー PRIV15 の 16 の特権レベルがあります。

ユーザが公開キーを介して認証する場合、対応する認証、許可、アカウンティング（AAA）設定がないと、このユーザは拒否されます。ユーザが公開キーを介して認証する場合、NETCONF の AAA 設定がローカル以外の AAA ソースを使用していると、このユーザも拒否されます。ローカルおよび TACACS + AAA 認証がサポートされます。

トークンベースの RESTCONF 認証はサポートされていません。SSH ユーザ証明書はサポートされていません。

候補コンフィギュレーションのサポート

候補コンフィギュレーション機能を使用すると、シンプルなコミットオプションを使用して RFC 6241 を実装することによって、候補機能をサポートできます。

候補データストアは、デバイスの実行コンフィギュレーションのコピーを保存する一時的な作業領域となります。実行コンフィギュレーションをデバイスにコミットする前に、実行コンフィギュレーションを作成して変更することができます。候補機能は、NETCONF 機能

urn:ietf:params:netconf:capability:candidate:1.0 により示されます。この NETCONF 機能は、デバイスが候補データストアをサポートしていることを示します。

ユーザはこの共有データストアを使用して、デバイスの実行コンフィギュレーションに影響を与えることなく、デバイスのコンフィギュレーションを作成、追加、削除、変更できます。コミット操作では、デバイスのコンフィギュレーションが候補から実行のコンフィギュレーションにプッシュされます。候補データストアが有効になっていると、実行のデータストアには NETCONF セッションを介して書き込むことができず、すべてのコンフィギュレーションは候補を通じてのみコミットされます。つまり、稼働中の設定を直接変更できる NETCONF 機能は、候補コンフィギュレーションでは有効になりません。

（注） 候補データストアは共有データストアであることに留意してください。複数の NETCONF セッションが内容を同時に変更する可能性があります。したがって、内容を変更する前にデータストアをロックして、コミットが競合しないようにし、最終的にコンフィギュレーションの変更が失われる可能性を防ぐことが重要になります。

候補の NETCONF 操作

候補データストアでは次の操作を実行できます。

（注） この項の情報は RFC 6241 の 8.3.4 項を参考にしています。詳細と正確な RPC については、RFC を参照してください。

ロック

<lock> RPC は、ターゲットのデータストアをロックするために使用します。これにより、他のユーザはロックされたデータストアのコンフィギュレーションを変更できなくなります。ロック操作では候補データと実行データの両方をロックできます。

（注） 候補データストアのロックは、Cisco IOS のコンフィギュレーションのロックや実行コンフィギュレーションのロックに影響を与えません。逆も同様です。

コミット

<commit> RPC は、候補コンフィギュレーションをデバイスの実行コンフィギュレーションにコピーします。「コミット」操作は、候補コンフィギュレーションを更新してコンフィギュレーションをデバイスにプッシュした後に実行する必要があります。

実行または候補のデータストアのいずれかが別の NETCONF セッションによってロックされている場合、<commit> RPC は RPC エラー応答で失敗します。<error-tag> は <in-use> となり、<error-info> にはロックを保持している NETCONF セッションのセッション ID が示されます。conf t lock モードに移行してグローバルロックを使用し、「実行」コンフィギュレーションを

ロックすることもできますが、コミット操作は RPC エラー応答で失敗し、error-tag の値は <in-use>、セッション ID は「0」になります。

コンフィギュレーション編集

候補コンフィギュレーションは、コンフィギュレーションを変更するための edit-config（コンフィギュレーション編集）操作のターゲットとして使用できます。デバイスの実行コンフィギュレーションに影響を与えることなく、候補コンフィギュレーションを変更できます。

廃棄

候補コンフィギュレーションに加えられた変更を削除するには、discard（廃棄）操作を実行して候補コンフィギュレーションを実行コンフィギュレーションに戻します。

たとえば、NETCONF セッション A によって候補データストアの内容が変更されている場合、セッション B が候補データストアをロックしようとするとロックは失敗します。NETCONF セッション B では候補をロックする前に、他の NETCONF セッションから候補データストアの未解決のコンフィギュレーションの変更を削除するために <discard> 操作を実行する必要があります。

ロック解除

ロック、edit-config（コンフィギュレーション編集）、コミットなどで候補コンフィギュレーションを操作した後、ロック解除 RPC でターゲットとして candidate を指定することによって、データストアをロック解除できます。これで、他のセッションでのすべての操作に候補データストアを使用できるようになります。

候補データストアに対する未解決の変更で不具合が発生した場合、コンフィギュレーションの回復が困難になり、他のセッションで問題が生じる可能性があります。問題を回避するため、未解決の変更は、「NETCONF セッションの障害」で暗黙的にロックが解除されたとき、またはロック解除操作を使用して明示的にロックが解除されたときに廃棄される必要があります。

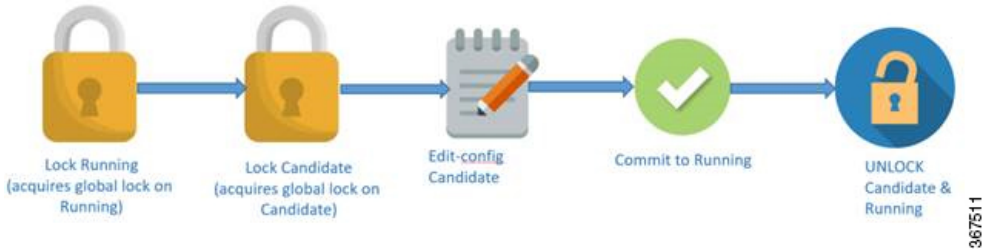
コンフィギュレーション取得、コンフィギュレーションコピー、コンフィギュレーション検証

候補データストアは、get-config（コンフィギュレーション取得）、copy-config（コンフィギュレーション コピー）、または validate（コンフィギュレーション検証）のどの操作でも、ソースまたはターゲットとして使用できます。候補データストアの変更をデバイスにコミットせずに、コンフィギュレーションの検証のみを行う場合は、<validate> RPC の後に discard の操作を付けることで使用できます。

候補データストアの変更

次の図は、候補データストアを介してデバイスコンフィギュレーションを変更する場合に推奨されるベスト プラクティスを示しています。

図 *1:* 候補データストアの変更手順



1. 実行データストアをロックします。
2. 候補データストアをロックします。
3. edit-config RPC とターゲットの候補を使用して、候補コンフィギュレーションを変更します。
4. 候補コンフィギュレーションを、実行コンフィギュレーションにコミットします。
5. 候補データストアと実行データストアをロック解除します。

確認済み候補コンフィギュレーションのコミット

候補コンフィギュレーションは、confirmed-commit 機能をサポートします。この実装では、confirmed-commit 機能に関する RFC 6241 で指定されているとおり、発行されると、実行コンフィギュレーションが候補コンフィギュレーションの現在の内容に設定され、confirmed-commit タイマーが開始されます。commit がタイムアウト期間内に発行されない場合、confirmed-commit 操作はロールバックされます。デフォルトのタイムアウト期間は 600 秒（10分）です。

候補コンフィギュレーションをコミットする場合、コミットを永続的にするための明示的な確認を要求できます。確認済みコミット操作は、コンフィギュレーションの変更が正しく機能し、デバイスへの管理アクセスを妨げないことを確認するのに役立ちます。変更によってアクセスが妨げられたり、その他のエラーが発生したりすると、ロールバックの期限が過ぎた後、以前のコンフィギュレーションへの自動ロールバックによってアクセスが復元されます。指定した時間内にコミットが確認されない場合、デバイスはデフォルトで、以前にコミットされたコンフィギュレーションを自動的に取得してコミット（ロールバック）します。

（注） RESTCONF は確認済みコミットをサポートしていません。

NETCONF セッションでは、候補コンフィギュレーションをコミットし、コミットが永続的になることを明示的に確認するために、クライアントアプリケーションは空の <confirmed/> タグを <commit> および <rpc> タグ要素内に囲みます。

```
<rpc>
  <commit>
    <confirmed/>
  </commit>
```

```
</rpc>
```

次に、デフォルトのロールバックタイマーを変更する RPC の例を示します。

```
<rpc>
    <commit>
        <confirmed/>
        <confirm-timeout>nnn</confirm-timeout> !nnn is the rollback-delay in seconds.
    </commit>
</rpc>
```

次のサンプル RPC は、NETCONF サーバが候補コンフィギュレーションが一時的にコミットされたことを確認することを示しています。

```
<rpc-reply xmlns="URN" xmlns:junos="URL">
    <ok/>
</rpc-reply>
```

NETCONF サーバが候補コンフィギュレーションをコミットできない場合、<rpc-reply> 要素で失敗の理由を説明する <rpc-error> 要素を囲みます。最も一般的な原因は、候補コンフィギュレーションのセマンティックまたは構文エラーです。

ロールバックを現在のロールバックタイマーよりも後の時間に遅らせるために、クライアントアプリケーションは、期限が切れる前に <commit> タグ要素内にある <confirmed/> タグを再度送信します。オプションで、<confirm-timeout> 要素を含めることで、次のロールバックを遅らせる時間を指定します。クライアントアプリケーションは、<confirmed/> タグを繰り返し送信することでロールバックを無制限に遅らせることができます。

コンフィギュレーションを永続的にコミットするには、ロールバック期限が過ぎる前に、クライアントアプリケーションが <rpc> タグ要素で囲まれた <commit/> タグを送信します。ロールバックがキャンセルされ、候補コンフィギュレーションがただちにコミットされます。候補コンフィギュレーションが、一時的にコミットされたコンフィギュレーションと同じ場合、一時的にコミットされたコンフィギュレーションが再コミットされます。

別のアプリケーションが <kill-session/> タグ要素を使用して、確認済みコミットが保留中の間にこのアプリケーションのセッションを終了する場合（このアプリケーションは変更をコミットしましたが、まだ確認していません）、このセッションを使用している NETCONF サーバは、確認済みコミット命令が発行される前の状態にコンフィギュレーションを復元します。

候補データストアは、 **no netconf-yang feature candidate-datastore** コマンドを使用することで無効になります。候補データストアが有効の場合に候補データストアの確認済みコミットが有効になるため、候補データストアが無効の場合は確認済みコミットが無効になります。進行中のすべてのセッションが終了し、confd プログラムが再起動されます。

候補サポートの設定

候補データストア機能は、 **netconf-yang feature candidate-datastore** コマンドを使用して有効にすることができます。データストアの状態が「実行」から「候補」、またはその逆に変わると、変更を有効にするために NETCONF または RESTCONF の再起動が行われることをユーザに通知する警告メッセージが表示されます。

NETCONF-YANG または RESTCONF confd プロセスの開始時に候補または実行のデータストアの選択がコンフィギュレーションで指定されている場合は、次のような警告が表示されます。

```
Device(config)# netconf-yang feature candidate-datastore

netconf-yang initialization in progress - datastore transition not allowed, please try
again after 30 seconds
```

NETCONF-YANG または RESTCONF confd プロセスの開始後に候補または実行の選択が行われた場合は、次のように適用されます。

- **netconf-yang feature candidate-datastore** コマンドが設定されている場合は、コマンドによって候補データストアが有効になり、次の警告が出力されます。

  ```
  "netconf-yang and/or restconf is transitioning from running to candidate netconf-yang
   and/or restconf will now be restarted,
  and any sessions in progress will be terminated".
  ```

- **netconf-yang feature candidate-datastore** コマンドが削除された場合は、コマンドによって候補データストアが無効になり、実行データストアが有効になり、次の警告が出力されます。

  ```
  netconf-yang and/or restconf is transitioning from candidate to running netconf-yang
   and/or restconf will now be restarted,
  and any sessions in progress will be terminated".
  ```

- NETCONF-YANG または RESTCONF が再起動すると、進行中のセッションは失われます。

コンフィギュレーション データベースの副次的同期

データ モデル インターフェイス（DMI）の設定変更中に、コマンドまたは RPC の設定時にトリガーされる変更の部分的な同期が行われます。これは副次的同期と呼ばれ、同期時間と NETCONF のダウンタイムを短縮します。副次的同期の前に、コンフィギュレーションデータベースの時間のかかる完全な同期をトリガーするため、設定変更が使用されます。

副次的同期は、**netconf-yang feature side-effect-sync** コマンドによって有効になります。

一部のコマンドは、設定されると、すでに設定されている一部のコマンドの変更をトリガーします。たとえば、次に NETCONF edit-config RPC が設定される前のデバイスの設定を示します。

```
hostname device123
```

NETCONF edit-config RPC：

```
<native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
 <hostname xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" nc:operation="delete"/>
</native>
```

次に、NETCONF edit-config RPC を設定した後のデバイスの設定を示します。

```
hostname Switch
```

ここで、NETCONF edit-config RPC の副作用は、RPC が直接意図していない実行コンフィギュレーションへの変更です。edit-config 要求はホスト名を削除することになっていますが、ホス

ト名は削除されずに Switch に戻ります。副次的同期では、設定全体を同期することなく、この設定変更を NETCONF データベースと同期するため、パフォーマンスが向上します。

副次的同期はCLIモードツリーの概念に基づいており、コマンドはモードとサブモード構造で維持されます。このCLIモードツリーのデータ構造は、次の3つのメインノードで構成されています。

- 同じレベルのノード：このノードは、同じ親に属し、同じレベルにあるCLIノードのリストを指します。
- 親ノード：このノードは、CLIノードの親、そのモード、およびサブモードノードを指します。
- 子ノード：このノードは子 CLI（現在のモードまたはサブモードでの CLI）を指します。ノードに複数の子ノードがある場合、それらの子ノードは同じレベルのノードポインタの一部としてリンクされます。

NETCONF プロトコルの設定方法

NETCONF-YANG は、デバイスのプライマリ トラストポイントを使用します。トラストポイントが存在しない場合に NETCONF-YANG が設定されると、自己署名トラストポイントが作成されます。詳細については、『Public Key Infrastructure Configuration Guide, Cisco IOS XE Gibraltar 16.10.x』を参照してください。

NETCONF を使用するための権限アクセスの提供

NETCONF API の使用を開始するには、権限レベル 15 を持つユーザである必要があります。

手順の概要

1. **enable**
2. **configure terminal**
3. **username** *name* **privilege** *level* **password** *password*
4. **aaa new-model**
5. **aaa authentication login default local**
6. **aaa authorization exec default local**
7. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： `Device# enable`	特権 EXEC モードを有効にします。 パスワードを入力します（要求された場合）。

	コマンドまたはアクション	目的
ステップ 2	**configure terminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**username** *name* **privilege** *level* **password** *password* 例： `Device(config)# username example-name privilege 15 password example_password`	ユーザ名をベースとした認証システムを確立します。次のキーワードを設定します。 • **privilege** *level*：ユーザの権限レベルを設定します。NETCONF プロトコルの場合は、15 にする必要があります。 • **password** *password*：CLI ビューにアクセスするためのパスワードを設定します。
ステップ 4	**aaa new-model** 例： `Device(config)# aaa new-model`	（任意）許可、認証、アカウンティング（AAA）を有効にします。 **aaa new-model** コマンドを設定する場合は、AAA 認証および許可が必要です。
ステップ 5	**aaa authentication login default local** 例： `Device(config)# aaa authentication login default local`	ローカルユーザ名データベースを使用するログイン認証を設定します。 （注） NETCONF プロトコルでは、デフォルトの AAA 認証ログイン方式のみがサポートされます。 • リモート AAA サーバの場合は、local を AAA サーバに置き換えます。 **default** キーワードにより、ローカルユーザデータベース認証がすべてのポートに適用されます。
ステップ 6	**aaa authorization exec default local** 例： `Device(config)# aaa authorization exec default local`	ユーザの AAA 許可を設定し、ローカルデータベースを確認して、そのユーザに EXEC シェルの実行を許可します。 （注） NETCONF プロトコルでは、デフォルトの AAA 認証方式のみがサポートされます。 • リモート AAA サーバの場合は、local を AAA サーバに置き換えます。

	コマンドまたはアクション	目的
		• **default** キーワードにより、ローカルユーザデータベース認証がすべてのポートに適用されます。
ステップ 7	**end** 例： `Device(config)# end`	グローバル コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。

NETCONF-YANG の設定

レガシー NETCONF プロトコルがデバイスで有効になっている場合、RFC 準拠の NETCONF プロトコルは機能しません。**no netconf legacy** コマンドを使用してレガシー NETCONF プロトコルを無効にしてください。

手順の概要

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

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： `Device> enable`	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**netconf-yang** 例： `Device (config)# netconf-yang`	ネットワーク デバイスで NETCONF インターフェイスを有効にします。 （注） CLI による最初のイネーブル化の後、ネットワーク デバイスをモデル ベースのインターフェイスを通じて管理できるようになります。モデル ベースのインターフェイス プロセスの完全なアクティベーションには、最大 90 秒かかることがあります。

	コマンドまたはアクション	目的
ステップ 4	**netconf-yang feature candidate-datastore** 例： `Device(config)# netconf-yang feature candidate-datastore`	候補データストアを有効にします。
ステップ 5	**exit** 例： `Device (config)# exit`	グローバル コンフィギュレーション モードを終了します。

NETCONF オプションの設定

SNMP の設定

NETCONF を有効にして、サポートされている MIB から生成された YANG モデルを使用して SNMP MIB データにアクセスしたり、IOS でサポートされている SNMP トラップを有効にして、サポートされているトラップから NETCONF 通知を受信するには、IOS で SNMP サーバを有効にします。

次の操作を行ってください。

手順の概要

1. IOS で SNMP 機能を有効にします。
2. NETCONF-YANG が起動した後、次の RPC <edit-config> メッセージを NETCONF-YANG ポートに送信して、SNMP トラップのサポートを有効にします。
3. 次の RPC メッセージを NETCONF-YANG ポートに送信して、実行コンフィギュレーションをスタートアップ コンフィギュレーションに保存します。

手順の詳細

ステップ 1 IOS で SNMP 機能を有効にします。

例：

```
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
```

ステップ 2 NETCONF-YANG が起動した後、次の RPC <edit-config> メッセージを NETCONF-YANG ポートに送信して、SNMP トラップのサポートを有効にします。

例：

```
<?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-oid>1.3.6.1.4.1.9.9.41.2.0.1</trap-oid>
            </trap-list>
            <trap-list>
              <trap-oid>1.3.6.1.6.3.1.1.5.3</trap-oid>
            </trap-list>
            <trap-list>
              <trap-oid>1.3.6.1.6.3.1.1.5.4</trap-oid>
            </trap-list>
          </snmp-trap-control>
        </cisco-ia>
      </netconf-yang>
    </config>
  </edit-config>
</rpc>
```

ステップ 3 次の RPC メッセージを NETCONF-YANG ポートに送信して、実行コンフィギュレーションをスタートアップ コンフィギュレーションに保存します。

例：

```
<?xml version="1.0" encoding="utf-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="">
  <cisco-ia:save-config xmlns:cisco-ia="http://cisco.com/yang/cisco-ia"/>
</rpc>
```

RSA ベースのユーザ認証を実行するための SSH サーバの設定

NETCONF-YANG がユーザを認証するための SSH 公開キーを設定するには、次の作業を実行します。

手順の概要

1. **enable**
2. **configure terminal**
3. **ip ssh pubkey-chain**
4. **username** *username*

5. **key-string**
6. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： `Device> enable`	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**ip ssh pubkey-chain** 例： `Device(config)# ip ssh pubkey-chain`	SSH サーバ上のユーザおよびサーバ認証用に SSH-RSA キーを設定し、公開キー コンフィギュレーション モードを開始します。 • サーバに保存されている RSA 公開キーが、クライアントに保存されている公開キーと秘密キーのペアを使用して検証されると、ユーザ認証は成功です。
ステップ 4	**username** *username* 例： `Device(conf-ssh-pubkey)# username user1`	SSH ユーザ名を設定し、公開キー ユーザ コンフィギュレーション モードを開始します。
ステップ 5	**key-string** 例： `Device(conf-ssh-pubkey-user)# key-string`	リモート ピアの RSA 公開キーを指定し、公開キー データ コンフィギュレーション モードを開始します。 （注） オープン SSH クライアントから（言い換えると .ssh/id_rsa.pub ファイルから）公開キー値を取得できます。
ステップ 6	**end** 例： `Device(conf-ssh-pubkey-data)# end`	公開キー データ コンフィギュレーション モードを終了し、特権 EXEC モードに戻ります。 • デフォルト ホストに戻るには、**no hostname** コマンドを使用します。

CLI を使用した NETCONF プロトコルのコンフィギュレーションの確認

NETCONF コンフィギュレーションを確認するには次のコマンドを使用します。

手順の概要

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

手順の詳細

ステップ 1 show netconf-yang datastores

NETCONF-YANG データストアに関する情報を表示します。

例：

```
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
```

ステップ 2 show netconf-yang sessions

NETCONF-YANG セッションに関する情報を表示します。

例：

```
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
session-id transport username source-host global-lock
--------------------------------------------------------------------------------
40 netconf-ssh admin 10.85.70.224 None
42 netconf-ssh admin 10.85.70.224 None
44 netconf-ssh admin 10.85.70.224 None
46 netconf-ssh admin 10.85.70.224 None
48 netconf-ssh admin 10.85.70.224 None
50 netconf-ssh admin 10.85.70.224 None
52 netconf-ssh admin 10.85.70.224 None
54 netconf-ssh admin 10.85.70.224 None
56 netconf-ssh admin 10.85.70.224 None
```

```
58 netconf-ssh admin 10.85.70.224 None
```

ステップ 3 show netconf-yang sessions detail

NETCONF-YANG セッションに関する詳細情報を表示します。

例：

```
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

session-id             : 19
transport              : netconf-ssh
username               : admin
source-host            : 2001:db8::1
login-time             : 2018-10-26T12:37:22+00:00
in-rpcs                : 0
in-bad-rpcs            : 0
out-rpc-errors         : 0
out-notifications      : 0
global-lock            : None
```

ステップ 4 show netconf-yang diagnostics summary

NETCONF-YANG 診断情報の概要を表示します。

例：

```
Device# show netconf-yang diagnostics summary

Diagnostic Debugging is ON
Diagnostic Debugging Level: Maximum
Total Log Size (bytes): 20097
Total Transactions: 1
message username session-id transaction-id start-time        end-time          log size
-------------------------------------------------------------------------------------------
1       admin    35          53             03/12/21 14:31:03 03/12/21 14:31:04 20097
```

ステップ 5 show netconf-yang statistics

NETCONF-YANG 統計に関する情報を表示します。

例：

```
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
in-bad-hellos : 0
```

ステップ 6　show platform software yang-management process

NETCONF-YANG のサポートに必要なソフトウェア プロセスのステータスを表示します。

例：

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

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

（注）　プロセス nginx は、**ip http secure-server** または **ip http server** がデバイスで設定されている場合に実行されます。このプロセスが「実行」状態でなくても NETCONF は正常に機能します。ただし、RESTCONF には nginx プロセスが必要です。

***表 1 : show platform software yang-management process* のフィールドの説明**

フィールド	説明
confd	コンフィギュレーション デーモン
nesd	ネットワーク要素シンクロナイザ デーモン
syncfd	デーモンからの同期
ncsshd	NETCONF セキュア シェル（SSH）デーモン
dmiauthd	デバイス管理インターフェイス（DMI）認証デーモン
vtyserverutild	VTY サーバ ユーティリティ デーモン
opdatamgrd	運用データ マネージャ デーモン
nginx	NGINX Web サーバ
ndbmand	NETCONF データベース マネージャ

RPC による NETCONF-YANG 診断の表示

show netconf-yang diagnostics コマンドまたは次の RPC を使用して、診断情報を表示できます。

次に、NETCONF-YANG 診断を有効にする RPC の例と、ホストから受信した RPC 応答を示します。

```
#308
<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="urn:uuid:b0f45ac0-3fe2-4e1d-a3a1-f57985965be6">
  <enable-netconf-diag xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-rpc">
    <diag-level>diag-maximum</diag-level>
  </enable-netconf-diag>
</nc:rpc>

##

Received message from host
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:b0f45ac0-3fe2-4e1d-a3a1-f57985965be6"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

次に、現在のステータスを示す RPC の例と、ホストから受信した RPC 応答を示します。

```
#294
<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="urn:uuid:c6c986ac-fc44-45e2-9390-f8a5968dc8d4">
  <nc:get>
    <nc:filter>
      <netconf-diag-oper-data
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-oper"/>
    </nc:filter>
  </nc:get>
</nc:rpc>

#

Received message from host
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:c6c986ac-fc44-45e2-9390-f8a5968dc8d4"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf-diag-oper-data
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-oper">
      <diag-summary>
        <level>diag-maximum</level>
        <log-size>0</log-size>
        <trans-count>0</trans-count>
      </diag-summary>
    </netconf-diag-oper-data>
  </data>
</rpc-reply>
```

次に、ホスト名を変更するための RPC の例と、ホストから受信した RPC 応答を示します。

```
#
#364
<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="urn:uuid:f3005ee6-8a11-4146-b616-dd95a92b97d1">
  <nc:edit-config>
    <nc:target>
      <nc:running/>
    </nc:target>
    <nc:config>
      <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
        <hostname>new-ott-c9300-35</hostname>
      </native>
    </nc:config>
  </nc:edit-config>
</nc:rpc>

##

Received message from host
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:f3005ee6-8a11-4146-b616-dd95a92b97d1"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

次に、現在のステータスを表示するための RPC の例と、ホストから受信した RPC 応答を示します。

```
 #294
<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="urn:uuid:9bffb8d5-3866-48ef-b59d-0486e508fbc4">
  <nc:get>
    <nc:filter>
      <netconf-diag-oper-data
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-oper"/>
    </nc:filter>
  </nc:get>
</nc:rpc>

##

Received message from host
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:9bffb8d5-3866-48ef-b59d-0486e508fbc4"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <netconf-diag-oper-data
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-oper">
      <diag-summary>
        <level>diag-maximum</level>
        <log-size>20775</log-size>
        <trans-count>1</trans-count>
      </diag-summary>
      <diag-trans>
        <message>1</message>
        <username>lab</username>
        <session-id>31</session-id>
        <trans-id>50</trans-id>
        <start-time>2021-03-12T14:08:26.830334+00:00</start-time>
```

```
        <end-time>2021-03-12T14:08:28.279414+00:00</end-time>
        <log-size>20775</log-size>
      </diag-trans>
    </netconf-diag-oper-data>
  </data>
</rpc-reply>
```

次に、収集されたシステムエラーメッセージをアーカイブするための PRC の例と、ホストから受信した RPC 応答を示します。

```
#
#256
<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="urn:uuid:1dbc795c-f594-4194-a89b-fd4d88446b69">
  <archive-netconf-diag-logs
xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-rpc"/>
</nc:rpc>

##

Received message from host
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:1dbc795c-f594-4194-a89b-fd4d88446b69"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <log-file xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-rpc">
    bootflash:netconf-yang-diag.20210312141009.tar.gz</log-file>


</rpc-reply>
```

次に、NETCONF-YANG 診断を無効にする RPC の例と、ホストから受信した RPC 応答を示します。

```
#309
<nc:rpc xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="urn:uuid:d253a313-4aec-42bc-80a2-672e9bb9ad56">
  <enable-netconf-diag xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-netconf-diag-rpc">
    <diag-level>diag-disabled</diag-level>
  </enable-netconf-diag>
</nc:rpc>

##

Received message from host
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:d253a313-4aec-42bc-80a2-672e9bb9ad56"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

NETCONF プロトコルの関連資料

関連資料

関連項目	マニュアル タイトル
IOS-XE、IOS-XR、および NX-OS プラットフォームのさまざまなリリースの YANG データ モデル	開発者に分かりやすい方法で Cisco YANG モデルにアクセスするには、*GitHub リポジトリ*を複製し、*vendor/cisco* サブディレクトリに移動します。ここでは、IOS XE、IOS-XR、および NX-OS プラットフォームのさまざまなリリースのモデルを使用できます。

標準および RFC

標準/RFC	タイトル
RFC 6020	*YANG：Network Configuration Protocol（NETCONF）向けデータ モデリング言語*
RFC 6241	*ネットワーク設定プロトコル（NETCONF）*
RFC 6536	*ネットワーク設定プロトコル（NETCONF）アクセス制御モデル*
RFC 7950	*YANG 1.1 データ モデリング言語*
RFC 8040	*RESTCONF プロトコル*

シスコのテクニカル サポート

説明	リンク
シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンライン リソースを提供しています。 お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service（Field Notice からアクセス）、Cisco Technical Services Newsletter、Really Simple Syndication（RSS）フィードなどの各種サービスに加入できます。 シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。	http://www.cisco.com/support

NETCONF プロトコルの機能情報

表 *2 : NETCONF* プロトコルの機能情報

機能名	リリース	機能情報
NETCONF プロトコル	Cisco IOS XE Denali 16.3.1	NETCONFプロトコル機能によって、プログラムによる各種の標準規格に準拠した方法で、設定の記述やネットワーク デバイスからの運用データの読み取りが容易になります。 次のコマンドが導入されました：**netconf-yang** この機能は、次のプラットフォームに実装されていました。 • Cisco 4000 シリーズ サービス統合型ルータ • Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ • Cisco Cloud Services Router 1000V シリーズ
	Cisco IOS XE Everest 16.5.1a	この機能は、次のプラットフォームに実装されていました。 • Cisco Catalyst 3650 シリーズ スイッチ • Cisco Catalyst 3850 シリーズ スイッチ
	Cisco IOS XE Everest 16.6.2	この機能は、次のプラットフォームに実装されていました。 • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ
	Cisco IOS XE Fuji 16.8.1a	

機能名	リリース	機能情報
		Cisco IOS XE Fuji 16.8.1a では、この機能は次のプラットフォームに実装されていました。 • Cisco 1000 シリーズ サービス統合型ルータ • Cisco ASR 900 シリーズアグリゲーション サービス ルータ • Cisco ASR 920 シリーズアグリゲーション サービス ルータ • Cisco Catalyst 9500 ハイ パフォーマンス シリーズ スイッチ • Cisco CBR-8 シリーズ ルータ • Cisco Network Convergence System 4200 シリーズ
	Cisco IOS XE Fuji 16.9.2	この機能は、次のプラットフォームに実装されていました。 • Cisco Catalyst 9200 および 9200L シリーズ スイッチ • Cisco Catalyst 9300L SKU
	Cisco IOS XE Gibraltar 16.10.1	Cisco IOS XE Gibraltar 16.10.1 では、この機能は次のプラットフォームに実装されていました。 • Cisco Catalyst 9800-40 ワイヤレスコントローラ • Cisco Catalyst 9800-80 ワイヤレスコントローラ • Cisco Network Convergence System 520 シリーズ
	Cisco IOS XE Gibraltar 16.11.1	Cisco IOS XE Gibraltar 16.11.1 では、この機能は Cisco Catalyst 9600 シリーズ スイッチに実装されていました。
	Cisco IOS XE Gibraltar 16.12.1	この機能は、Cisco IOS XE Gibraltar 16.12.1 で、Cisco Catalyst 9800-L ワイヤレスコントローラに実装されました。

機能名	リリース	機能情報
	Cisco IOS XE Amsterdam 17.3.1	この機能は、Cisco IOS XE Amsterdam 17.3.1 で次のプラットフォームに実装されました。 • Cisco Catalyst 8200 シリーズ エッジ プラットフォーム • Cisco Catalyst 8300 シリーズ エッジ プラットフォーム • Cisco Catalyst 8500 および 8500L シリーズ エッジ プラットフォーム

機能名	リリース	機能情報
NETCONF および RESTCONF IPv6 のサポート	Cisco IOS XE Fuji 16.8.1a	NETCONF および RESTCONF プロトコルの IPv6 のサポート。この機能は、次のプラットフォームに実装されていました。 • Cisco 4000 シリーズ サービス統合型ルータ • Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ • Cisco ASR 900 シリーズ アグリゲーション サービス ルータ • Cisco Catalyst 3650 シリーズ スイッチ • Cisco Catalyst 3850 シリーズ スイッチ • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ • Cisco CBR-8 シリーズ ルータ • Cisco Cloud Services Router 1000V シリーズ
	Cisco IOS XE Gibraltar 16.11.1	Cisco IOS XE Gibraltar 16.11.1 では、この機能は Cisco Catalyst 9500 ハイ パフォーマンス シリーズ スイッチに実装されていました。

機能名	リリース	機能情報
NETCONF グローバル ロックおよびセッションの kill	Cisco IOS XE Fuji 16.8.1a	NETCONF プロトコルは、グローバル ロックおよび応答しなくなったセッションを kill する機能をサポートしています。この機能は、次のプラットフォームに実装されています。 • Cisco 1100 シリーズ サービス統合型ルータ • Cisco 4000 シリーズ サービス統合型ルータ • Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ • Cisco ASR 900 シリーズアグリゲーション サービス ルータ • Cisco Catalyst 3650 シリーズ スイッチ • Cisco Catalyst 3850 シリーズ スイッチ • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ • Cisco CBR-8 シリーズ ルータ • Cisco Cloud Services Router 1000V シリーズ

機能名	リリース	機能情報
NETCONF：候補コンフィギュレーション サポート	Cisco IOS XE Fuji 16.9.1	候補コンフィギュレーション サポート機能を使用すると、シンプルなコミット オプションを使用して RFC 6241 を実装することによって、候補機能をサポートできます。 この機能は、次のプラットフォームに実装されていました。 • Cisco 4000 シリーズ サービス統合型ルータ • Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ • Cisco ASR 900 シリーズ アグリゲーション サービス ルータ • Cisco Catalyst 3650 シリーズ スイッチ • Cisco Catalyst 3850 シリーズ スイッチ • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ • Cisco CBR-8 シリーズ ルータ • Cisco Cloud Services Router 1000V シリーズ 次のコマンドが導入されました：**netconf-yang feature candidate-datastore**

機能名	リリース	機能情報
NETCONF：候補コンフィギュレーションのコミットの確認	Cisco IOS XE Amsterdam 17.1.1	候補コンフィギュレーションは、confirmed-commit 機能をサポートします。 この機能は、次のプラットフォームに実装されていました。 • Cisco 1000 シリーズ サービス統合型ルータ • Cisco 4000 シリーズ サービス統合型ルータ • Cisco ASR 1000 シリーズ アグリゲーション サービス ルータ • Cisco ASR 900 シリーズアグリゲーション サービス ルータ • Cisco Catalyst 3650 シリーズ スイッチ • Cisco Catalyst 3850 シリーズ スイッチ • Cisco Catalyst 9200 シリーズ スイッチ • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ • Cisco cBR-8 コンバージドブロードバンド ルータ • Cisco Cloud Services Router 1000V シリーズ • Cisco Network Convergence System 520 シリーズ • Cisco Network Convergence System 4200 シリーズ

機能名	リリース	機能情報
NETCONF-YANG SSH サーバのサポート	Cisco IOS XE Gibraltar 16.12.1	

機能名	リリース	機能情報
		この機能は、次のプラットフォームに実装されていました。 • Cisco 1000 シリーズ サービス統合型ルータ • Cisco 4000 シリーズ サービス統合型ルータ • Cisco ASR 900 シリーズアグリゲーション サービス ルータ • Cisco ASR 920 シリーズアグリゲーション サービス ルータ • Cisco ASR 1000 アグリゲーション サービス ルータ • Cisco Catalyst 3650 シリーズ スイッチ • Cisco Catalyst 3850 シリーズ スイッチ • Cisco Catalyst 9200 シリーズ スイッチ • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ • Cisco Catalyst 9600 シリーズ スイッチ • Cisco Catalyst 9800 シリーズ ワイヤレス コントローラ • Cisco cBR-8 コンバージドブロードバンド ルータ • Cisco Cloud Services Router 1000V シリーズ • Cisco Network Convergence System 520 シリーズ

機能名	リリース	機能情報
		• Cisco Network Convergence System 4200 シリーズ
コンフィギュレーションデータベースの副次的同期	Cisco IOS XE Bengaluru 17.4.1	DMI の設定変更中に、コマンドまたは RPC の設定時にトリガーされる変更の部分的な同期が行われます。これは副次的同期と呼ばれ、同期時間と NETCONF のダウンタイムを短縮します。 この機能は、次のプラットフォームに実装されていました。 • Cisco ASR 1000 アグリゲーションサービス ルータ • Cisco Catalyst 8500 および 8500L シリーズ エッジ プラットフォーム • Cisco Catalyst 9200 シリーズ スイッチ • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ • Cisco Catalyst 9600 シリーズ スイッチ

機能名	リリース	機能情報
YANG モデルバージョン 1.1	Cisco IOS XE Cupertino 17.7.1	

機能名	リリース	機能情報
		Cisco IOS XE Cupertino 17.7.1 は YANG バージョン 1.0 を使用しますが、YANG バージョン 1.1 も使用できます。YANG バージョン 1.1 は、https://github.com/YangModels/yang/tree/master/vendor/cisco/xe フォルダの GitHub からダウンロードできます。 この機能は、次のプラットフォームに実装されていました。 • Cisco 1000 シリーズ サービス統合型ルータ • Cisco 4000 シリーズ サービス統合型ルータ • Cisco ASR 900 アグリゲーション サービス ルータ • Cisco ASR 920 アグリゲーション サービス ルータ • Cisco ASR 1000 アグリゲーション サービス ルータ • Cisco Catalyst 9200 および 9200L シリーズ スイッチ • Cisco Catalyst 9300 および 9300L シリーズ スイッチ • Cisco Catalyst 9400 シリーズ スイッチ • Cisco Catalyst 9500 および 9500 ハイパフォーマンス シリーズ スイッチ • Cisco Catalyst 9600 シリーズ スイッチ • Cisco Catalyst 9800-40 ワイヤレスコントローラ • Cisco Catalyst 9800-80 ワイヤレスコントローラ • Cisco cBR-8 コンバージド ブロードバンド ルータ • Cisco Cloud Services Router 1000V

機能名	リリース	機能情報
		シリーズ • Cisco Network Convergence System 520 シリーズ • Cisco Network Convergence System 4200 シリーズ
	Cisco IOS XE Cupertino 17.8.1	Cisco IOS XE Cupertino 17.8.1 では、YANG バージョン 1.1 を使用しています。YANG バージョン 1.1 とバージョン 1.0 の違いは、https://tools.ietf.org/html/rfc7950#page-10 [英語] に記載されています。
	Cisco IOS XE Dublin 17.10.1	Cisco IOS XE Dublin 17.10.1 以降のリリースでは、シスコ定義の YANG モデルは、YANG バージョン 1.1 になっています。
IOS コマンドの XML への変換	Cisco IOS XE Cupertino 17.7.1	この機能は、IOS コマンドを関連する NETCONF-YANG XML または RESTCONF-JSON 要求メッセージに自動的に変換するために役立ちます。 この機能は、NETCONF-YANG をサポートするすべてのプラットフォームでサポートされています。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。