RESTCONF Protocol

This chapter describes how to configure the HTTP-based Representational State Transfer Configuration Protocol (RESTCONF). RESTCONF provides a programmatic interface based on standard mechanisms for accessing configuration data, state data, data-model-specific Remote Procedure Call (RPC) operations and events, defined in the YANG model.

Prerequisites for the RESTCONF Protocol

- Enable the Cisco IOS-HTTP services for RESTCONF. For more information, see Examples for RESTCONF RPCs

Restrictions for the RESTCONF Protocol

The following restrictions apply to the RESTCONF protocol:

- Notifications and event streams
- YANG patch
- Optional query parameters, such as, filter, start-time, stop-time, replay, and action
- The RESTCONF feature is not supported on a device running dual IOSd configuration or software redundancy.

Information About RESTCONF Programmable Interface

Overview of RESTCONF

This section describes the protocols and modelling languages that enable a programmatic way of writing configurations to a network device.

- RESTCONF—Uses structured data (XML or JSON) and YANG to provide a REST-like APIs, enabling you to programmatically access different network devices. RESTCONF APIs use HTTPs methods.
- YANG—A data modelling language that is used to model configuration and operational features . YANG determines the scope and the kind of functions that can be performed by NETCONF and RESTCONF APIs.

RESTCONF and NETCONF in IOS

Protocols and Data Models for Programmatic Device

This section describes the protocols and modelling languages that enable a programmatic way of writing configurations to a network device.

- RESTCONF— Uses structured data (XML or JSON) and YANG to provide a REST-like APIs, enabling you to programmatically access different network devices. RESTCONF APIs use HTTPs methods.
- YANG—A data modelling language that is used to model configuration and operational features . YANG determines the scope and the kind of functions that can be performed by NETCONF and RESTCONF APIs.

If a RESTCONF server is co-located with a NETCONF server, then there are protocol interactions with the NETCONF protocol. The RESTCONF server provides access to specific datastores using operation resources, however, the RESTCONF protocol does not specify any mandatory operation resources, each operation resource determine if and how datastores are accessed.

For more information, refer to the Protocols and Data Models for Programmatic Device section in the Catalyst 4500 Series Software Configuration Guide.

HTTPs Methods

The https-based protocol-RESTCONF (RFC 8040), which is a stateless protocol, uses secure HTTP methods to provide CREATE, READ, UPDATE and DELETE (CRUD) operations on a conceptual datastore containing YANG-defined data, which is compatible with a server that implements NETCONF datastores.

The following table shows how the RESTCONF operations relate to NETCONF protocol operations:

OPTIONS	SUPPORTED METHODS
GET	Read
PATCH	Update
PUT	Create or Replace

OPTIONS	SUPPORTED METHODS
POST	Create or Operations (reload, default)
DELETE	Deletes the targeted resource
HEAD	Header metadata (no response body)

RESTCONF Root Resource

- A RESTCONF device determines the root of the RESTCONF API through the link element: /.well-known/host-meta resource that contains the RESTCONF attribute.
- The RESTCONF device uses the restconf api root resource as the initial part of the path in the request URI.

For example:

```
Example returning /restconf:

   The client might send the following:

      GET /.well-known/host-meta HTTP/1.1
      Host: example.com
      Accept: application/xrd+xml

   The server might respond as follows:

      HTTP/1.1 200 OK
      Content-Type: application/xrd+xml
      Content-Length: nnn

      <XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'>
          <Link rel='restconf' href='/restconf'/>
      </XRD>
```

Example of URIs:

- GigabitEthernet0/0/2 - http://10.104.50.97/restconf/data/Cisco-IOS-XE-native:native/interface/GigabitEthernet=0%2F0%2F2
- fields=name – http://10.104.50.97/restconf/data/Cisco-IOS-XE-native:native/interface/GigabitEthernet=0%2F0%2F2?fields=name
- depth=1 - https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native/interface/GigabitEthernet?depth=1
- Name and IP - https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native/interface?fields=GigabitEthernet/ip/address/primary;name
- MTU (fields) - https://10.104.50.97/restconf/data/Cisco-IOS-XE-native:native/interface?fields=GigabitEthernet(mtu)
- MTU - https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native/interface/GigabitEthernet=3/mtu

- Port-Channel - https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native/interface/Port-channel
- “Char” to “Hex” conversion chart: http://www.columbia.edu/kermit/ascii.html

RESTCONF API Resource

The API resource is the top-level resource located at +restconf. It supports the following media types:

- Application/YANG-Data+XML OR Application/YANG-Data+JSON
- The API resource contains the RESTCONF root resource for the RESTCONF DATASTORE and OPERATION resources. For example:

```
The client may then retrieve the top-level API resource, using the
   root resource "/restconf".

      GET /restconf HTTP/1.1
      Host: example.com
      Accept: application/yang-data+json

   The server might respond as follows:

      HTTP/1.1 200 OK
      Date: Thu, 26 Jan 2017 20:56:30 GMT
      Server: example-server
      Content-Type: application/yang-data+json

      {
        "ietf-restconf:restconf" : {
          "data" : {},
          "operations" : {},
          "yang-library-version" : "2016-06-21"
        }
      }
```

For more information, refer to RFC 3986

Reserved or Unreserved Characters

Conbody

Methods

The content query parameter controls how descendant nodes of the requested data nodes are processed in the reply:

- Must be supported by the server.
- If not present in URI, the default value is: all. Allowed only for GET/HEAD method.

  A "400 Bad Request" status-line is returned if used for other methods or resource types.

Examples for allowed values are:

1. https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native?content=config

2. https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native?content=nonconfig’

Query Parameters (Fields)

- The depth-query parameter is used to limit the depth of subtrees returned by the server.
- The value of the "depth" parameter is either an integer between 1 and 65535 or the string "unbounded".
- Supported if present in the capability URI.
- If not present in URI, the default value is: “unbounded”.
- Only allowed for GET/HEAD method.

  A 400 Bad Request status-line is returned if used for other methods or resource types.

Examples:

```
1) ‘https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native?content=config&depth=65535’

2) ‘https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native?content=nonconfig&depth=0’

>>> resp

<Response [400]>

>>> resp.text

'{"errors": {"error": [{"error-message": "invalid value for depth query parameter",
"error-tag": "malformed-message", "error-type": "application"}]}}\n'

>>>
```

Examples:

- • The "fields" query parameter is used to optionally identify data nodes within the target resource to be retrieved in a GET method.
- Supported if present in the capability URI.

  Allowed only for GET/HEAD method.
- A "400 Bad Request" status-line is returned if used for other methods or resource types.
- A value of the "fields" query parameter matches the following rule:

  ```
  fields-expr = path "(" fields-expr ")" / path ";" fields-expr / path path = api-identifier
   [ "/" path ]
  ```

- 1. ";" is used to select multiple nodes.

  2. Parentheses are used to specify sub-selectors of a node. Note that there is no path separator character "/" between a "path" field and a left parenthesis character "(".

  3. "/" is used in a path to retrieve a child node of a node.

- A value of the "fields" query parameter matches the following rule:

  ```
  fields-expr = path "(" fields-expr ")" / path ";" fields-expr / path path = api-identifier
   [ "/" path ]
  ```

- 1. ";" is used to select multiple nodes.

2. Parentheses are used to specify sub-selectors of a node. Note that there is no path separator character "/" between a "path" field and a left parenthesis character "(".
3. "/" is used in a path to retrieve a child node of a node.

Examples:

1. Server module information:
   'https://10.85.116.59:443/restconf/data?fields=ietf-yang-library:modules-state/module(name;revision;schema;namespace)'
2. Name and IP:
   'https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:native/interface?fields=GigabitEthernet/ip/address/primary;name'

Query Parameters (Point)

- The "point" query parameter uses to specify the insertion point for a data resource that is being created or moved within an ordered-by user list or leaf-list.
- Must be supported by the server:
  - Only allowed for POST and PUT methods.

    The value of the "point" parameter is a string that identifies the path to the insertion point object. The format is the same as a target resource URI string.

Examples:

```
PUT:
https://10.85.116.59:443/restconf/data/Cisco-IOS-XE-native:privilege/exec/level=2/command-list=show%20terminal?insert=after&point=%2Fcisco-IOS-XE-native%3Aprivilege%2Fexec%2Flevel%3D2%2Fcommand-list=show%20t

{

  "Cisco-IOS-XE-native:command-list": [

    {

      "command": "show terminal"

    }

  ]
```

Query Parameters (with defaults)

The 'with-defaults' query parameter is used to specify how information about default data nodes is returned in response to GET requests on data resources. Default basic-mode in capability is explicit.

Value	Description
Report-All	All data nodes are reported
Trim	Data nodes set to the YANG default are not reported
Explicit	Data nodes set to the YANG default by the client are reported

- The "point" query parameter uses to specify the insertion point for a data resource that is being created or moved within an ordered-by user list or leaf-list.

Examples:

```
Sync default settings (error):
'https://10.85.116.59:443/restconf/data/cisco-self-mgmt:netconf-yang/cisco-ia:cisco-ia/cisco-ia:logging/cisco-ia:sync-log-level?with-defaults=report-all'
Intelligent sync (true):
'https://10.85.116.59:443/restconf/data/cisco-self-mgmt:netconf-yang/cisco-ia:cisco-ia/cisco-ia:intelligent-sync?with-defaults=report-all'
```

How to Configure RESTCONF Programmable Interface

Authentication of NETCONF/RESTCONF Using AAA

Before you begin

NETCONF and RESTCONF connections must be authenticated using authentication, authorization, and accounting (AAA). As a result, RADIUS or TACACS+ users defined with privilege level 15 access are allowed access into the system.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Device(config)# aaa new-model`	Enables AAA.
Step 4	**aaa group server radius** *server-name* **Example:** `Device(config)# aaa group server radius ISE`	Adds the RADIUS server and enters server group RADIUS configuration mode. • The *server-name* argument specifies the RADIUS server group name.
Step 5	**server-private** *ip-address* **key** *key-name* **Example:** `Device(config-sg-radius)# server-private 172.25.73.76 key Cisco123`	Configures a IP address and encryption key for a private RADIUS server.

	Command or Action	Purpose
Step 6	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-sg-radius)# ip vrf forwarding Mgmt-intf`	Configures the virtual routing and forwarding (VRF) reference of a AAA RADIUS or TACACS+ server group.
Step 7	**exit** **Example:** `Device(config-sg-radius)# exit`	Exits server group RADIUS configuration mode and returns to global configuration mode.
Step 8	**aaa authentication login default group** *group-name* **local** **Example:** `Device(config)# aaa authentication login default group ISE local`	Sets the specified group name as the default local AAA authentication during login.
Step 9	**aaa authentication login** *list-name* **none** **Example:** `Device(config)# aaa authentication login NOAUTH none`	Specifies that no authentication is required while logging into a system.
Step 10	**aaa authorization exec default group** *group-name* **local** **Example:** `Device(config)# aaa authorization exec default group ISE local`	Runs authorization to determine if an user is allowed to run an EXEC shell.
Step 11	**aaa session-id common** **Example:** `Device(config)# aaa session-id common`	Ensures that session identification (ID) information that is sent out for a given call will be made identical.
Step 12	**line console** *number* **Example:** `Device(config)# line console 0`	Identifies a specific line for configuration and enter line configuration mode.
Step 13	**login authentication** *authentication-list* **Example:** `Device(config-line)# login authentication NOAUTH`	Enables AAA authentication for logins.
Step 14	**end** **Example:** `Device(config-line)# end`	Exits line configuration mode and returns to privileged EXEC mode.

Enabling Cisco IOS HTTP Services for RESTCONF

Perform this task to use the RESTCONF interface.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**restconf** **Example:** `Device(config)# restconf`	Enables the RESTCONF interface on your network device.
Step 4	**ip http secure-server** **Example:** `Device(config)# ip http secure-server`	Enables a secure HTTP (HTTPS) server.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and enters privileged EXEC mode

Verifying RESTCONF Configuration

When a device boots up with the startup configuration, the *nginx* process will be running. However; DMI proceses are not enabled.

The following sample output from the **show platform software yang-management process monitor** command shows that the *nginx* process is running:

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

COMMAND           PID S    VSZ   RSS %CPU %MEM     ELAPSED
nginx           27026 S 332356 18428  0.0  0.4       01:34
nginx           27032 S 337852 13600  0.0  0.3       01:34
```

NGINX is an internal webserver that acts as a proxy webserver. It provides Transport Layer Security (TLS)-based HTTPS. RESTCONF request sent via HTTPS is first received by the NGINX proxy web serve,r and the request is transferred to the confd web server for further syntax/semantics check.

The following sample output from the **show platform software yang-management process** command shows the status of the all processes when a device is booted with the startup-configuration:

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

confd            : Not Running
nesd             : Not Running
syncfd           : Not Running
ncsshd           : Not Running
dmiauthd         : Not Running
nginx            : Running
ndbmand          : Not Running
pubd             : Not Running
```

The *nginx* process gets restrated and DMI process are started, when the **restconf** command is configured.

The following sample output from the **show platform software yang-management process** command shows that the *nginx* process and DMI processes are up and running:

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

confd            : Running
nesd             : Running
syncfd           : Running
ncsshd           : Not Running ! NETCONF-YANG is not configured, hence ncsshd process is
in not running.
dmiauthd         : Running
vtyserverutild   : Running
opdatamgrd       : Running
nginx            : Running ! nginx process is up due to the HTTP configuration, and it is
restarted when RESTCONF is enabled.
ndbmand          : Running
```

The following sample output from the show platform software yang-management process monitor command displays detailed information about all processes:

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

COMMAND           PID S    VSZ   RSS %CPU %MEM     ELAPSED
confd           28728 S 860396 168496 42.2  4.2      00:12
confd-startup.s 28448 S  19664  4496  0.2  0.1       00:12
dmiauthd        29499 S 275356 23340  0.2  0.5       00:10
ndbmand         29321 S 567232 65564  2.1  1.6       00:11
nesd            29029 S 189952 14224  0.1  0.3       00:11
nginx           29711 S 332288 18420  0.6  0.4       00:09
nginx           29717 S 337636 12216  0.0  0.3       00:09
pubd            28237 S 631848 68624  2.1  1.7       00:13
syncfd          28776 S 189656 16744  0.2  0.4       00:12
```

After AAA and the RESTCONF interface is configured, and nginx process and relevant DMI processes are running; the device is ready to receive RESTCONF requests.

Use the **show netconf-yang sessions** command to view the status of NETCONF/RESTCONF sessions:

```
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 : 1

session-id  transport    username             source-host            global-lock
```

```
-------------------------------------------------------------------------------
19          netconf-ssh  admin                2001:db8::1             None
```

Use the **show netconf-yang sessions detail** command to view detailed information about NETCONF/RESTCONF sessions:

```
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
```

Configuration Examples for RESTCONF Programmable Interface

Example: Configuring the RESTCONF Protocol

RESTCONF Requests (HTTPS Verbs):

The following is a sample RESTCONF request that shows the HTTPS verbs allowed on a targeted resource. In this example, the **logging monitor** command is used..

```
root:~# curl -i -k -X "OPTIONS"
"https://10.85.116.30:443/restconf/data/Cisco-IOS-XE-native:native/logging/monitor/severity"
 \
>      -H 'Accept: application/yang-data+json' \
>      -u 'admin:admin'
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 23 Apr 2018 15:27:57 GMT
Content-Type: text/html
Content-Length: 0
Connection: keep-alive
Allow: DELETE, GET, HEAD, PATCH, POST, PUT, OPTIONS    >>>>>>>>>>>    Allowed methods
Cache-Control: private, no-cache, must-revalidate, proxy-revalidate
Accept-Patch: application/yang-data+xml, application/yang-data+json
Pragma: no-cache

root:~#
```

POST (Create) Request

The POST operation creates a configuration which is not present in the targeted device.

Note Ensure that the **logging monitor** command is not availabel in the running configuration.

The following sample POST request uses the **logging monitor alerts** command.

```
Device:~# curl -i -k -X "POST"
"https://10.85.116.30:443/restconf/data/Cisco-IOS-XE-native:native/logging/monitor" \
>      -H 'Content-Type: application/yang-data+json' \
>      -H 'Accept: application/yang-data+json' \
>      -u 'admin:admin' \
>      -d $'{
>   "severity": "alerts"
> }'
HTTP/1.1 201 Created
Server: nginx
Date: Mon, 23 Apr 2018 14:53:51 GMT
Content-Type: text/html
Content-Length: 0
Location:
https://10.85.116.30/restconf/data/Cisco-IOS-XE-native:native/logging/monitor/severity
Connection: keep-alive
Last-Modified: Mon, 23 Apr 2018 14:53:51 GMT
Cache-Control: private, no-cache, must-revalidate, proxy-revalidate
Etag: 1524-495231-97239
Pragma: no-cache

Device:~#
```

PUT: (Create or Replace) Request:

If the specified command is not present on the device, the POST request creates it ; however, if it is already present in the running configuration, the command will be replaced by this request.

The following sample PUT request uses the **logging monitor warnings** command.

```
Device:~# curl -i -k -X "PUT"
"https://10.85.116.30:443/restconf/data/Cisco-IOS-XE-native:native/logging/monitor/severity"
 \
>      -H 'Content-Type: application/yang-data+json' \
>      -H 'Accept: application/yang-data+json' \
>      -u 'admin:admin' \
>      -d $'{
>   "severity": "warnings"
> }'
HTTP/1.1 204 No Content
Server: nginx
Date: Mon, 23 Apr 2018 14:58:36 GMT
Content-Type: text/html
Content-Length: 0
Connection: keep-alive
Last-Modified: Mon, 23 Apr 2018 14:57:46 GMT
Cache-Control: private, no-cache, must-revalidate, proxy-revalidate
Etag: 1524-495466-326956
Pragma: no-cache

Device:~#
```

PATCH: (Update) Request

The following sample PATCH request uses the **logging monitor informational** command.

```
Device:~# curl -i -k -X "PATCH"
"https://10.85.116.30:443/restconf/data/Cisco-IOS-XE-native:native" \
>      -H 'Content-Type: application/yang-data+json' \
>      -H 'Accept: application/yang-data+json' \
>      -u 'admin:admin' \
>      -d $'{
>   "native": {
>     "logging": {
>       "monitor": {
>         "severity": "informational"
>       }
>     }
>   }
> }'
HTTP/1.1 204 No Content
Server: nginx
Date: Mon, 23 Apr 2018 15:07:56 GMT
Content-Type: text/html
Content-Length: 0
Connection: keep-alive
Last-Modified: Mon, 23 Apr 2018 15:07:56 GMT
Cache-Control: private, no-cache, must-revalidate, proxy-revalidate
Etag: 1524-496076-273016
Pragma: no-cache
Device:~#
```

GET Request (To Read)

The following sample GET request uses the **logging monitor informational** command.

```
Device:~# curl -i -k -X "GET"
"https://10.85.116.30:443/restconf/data/Cisco-IOS-XE-native:native/logging/monitor/severity"
 \
>      -H 'Accept: application/yang-data+json' \
>      -u 'admin:admin'
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 23 Apr 2018 15:10:59 GMT
Content-Type: application/yang-data+json
Transfer-Encoding: chunked
Connection: keep-alive
Cache-Control: private, no-cache, must-revalidate, proxy-revalidate
Pragma: no-cache

{
  "Cisco-IOS-XE-native:severity": "informational"
}
Device:~#
```

DELETE Request (To Delete the Configuration)

```
Device:~# curl -i -k -X "DELETE"
"https://10.85.116.30:443/restconf/data/Cisco-IOS-XE-native:native/logging/monitor/severity"
 \
>      -H 'Content-Type: application/yang-data+json' \
>      -H 'Accept: application/yang-data+json' \
>      -u 'admin:admin'
HTTP/1.1 204 No Content
Server: nginx
Date: Mon, 23 Apr 2018 15:26:05 GMT
Content-Type: text/html
Content-Length: 0
Connection: keep-alive
Last-Modified: Mon, 23 Apr 2018 15:26:05 GMT
Cache-Control: private, no-cache, must-revalidate, proxy-revalidate
Etag: 1524-497165-473206
Pragma: no-cache

linux_host:~#
```

Additional References for the RESTCONF Protocol

Related Documents

Related Topic	Document Title
YANG data models for various releases of IOS XE, IOS XR, and NX-OS platforms	To access Cisco YANG models in a developer-friendly way, please clone the GitHub repository, and navigate to the vendor/ciscosubdirectory. Models for various releases of IOS-XE, IOS-XR, and NX-OS platforms are available here.

Standards and RFCs

Standard/RFC	Title
RFC 6020	YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)
RFC 8040	Representational State Transfer Configuration Protocol (RESTCONF)

Technical Assistance

Description	Link
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.	https://www.cisco.com/c/en/us/support/index.html

Feature Information for the RESTCONF 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/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for the RESTCONF Protocol

Feature Name	Releases	Feature Information
RESTCONF Protocol	Cisco IOS XE Everest 16.6.1	RESTCONF provides a programmatic interface based on standard mechanisms for accessing configuration data, state data, data-model-specific RPC operations and event notifications defined in the YANG model. This feature was introduced on the following platforms: • Cisco 4000 Series Integrated Services Router • Cisco ASR 1000 Aggregation Services Routers • Cisco Cloud Services Router 1000V Series The following commands were introduced or modified: **ip http server** and **restconf**