Hewlett Packard Enterprise

REST API and JSON Schema

Abstract

This switch software guide is intended for network administrators and support personnel, and applies to the switch models listed on this page unless otherwise noted. This guide does not provide information about upgrading or replacing switch hardware. The information in this guide is subject to change without notice.

Applicable Products

Aruba 2530 Switch Series (J9772A-J9778A, JL070A, J9853-J9855A, J9779-J9783A)
Aruba 2920 Switch Series (J9836A, J9726A-J9729A)
Aruba 3810 Switch Series (JL071A, JL072A, JL073A, JL074A, JL075A, JL076A)
Aruba 5400Rzl2 Switch Series (J8698A, J8700A, J9823A-J9824A, J9825A, J9826A, J9868A, J9447A, J9448A)
Aruba 5406R Switch Series (JL002A, JL003A, JL095A,J9850A)
Aruba 5406zl Switch Series (J9821A, J9822A)
Aruba 5412R Switch Series (J9851A, JL001A)
HPE 2620 Switch Series (J9623A–J9627A)
HPE 3800 Switch Series (J9573A–J9576A, J9584A–J9588A)

Part Number: 5200-1316
Published: April 2016
Edition: 1

Contents

1 REST API

1.1 Overview

Representational State Transfer (REST) is a software architecture style consisting of guidelines and best practices for creating scalable web services. RESTful systems typically, but not always, communicate over HTTP using the same verbs (i.e. GET, POST, UT, DELETE) used by web browsers to retrieve web pages and send data to remote servers.

Both REST and Web UI requests are received on port 80. While reading the request, REST identifies the request and hands it over to a REST control task. The existing HTTP module is then modified to identify the REST request. You can then send REST requests on an HTTP port and receive responses on the same port. Rest requests can also be send on HTTPS port (443).

A new task called `REST_ctrl` is created to handle the messages being sent by an HTTP task for REST request processing. This task is created with Consumer priority and acts as a postmaster between the HTTP and REST server tasks. Both HTTP daemon tasks and `REST_ctrl` tasks have the same priorities (i.e. consumer). A new task creation will avoid overloading the HTTP module for processing REST requests.

NOTE:

- The REST interface is enabled by default on the device.
- It is mandatory that either web management or web management SSL be enabled. If management is disabled, requests will not be processed, irrespective of the REST interface status.

More information

JSON Schema and REST API documentation for X.16.01.xxxx software: **JSON Schema**

1.2 Commands

1.2.1 rest-interface

From within the configure context:

Syntax

```
rest-interface
```

Description

Enables the REST interface.

NOTE: WEB management or web management SSL should always be enabled to process the REST requests. If WEB management is disabled, the REST requests cannot be processed even though the REST interface is enabled.

Example 1 show rest-interface

```
HP-2530-48G-PoEP# show rest-interface
REST Interface - Server Configuration
  REST Interface             : Enabled
  REST Operational Status    : Up
  REST Session Idle Timeout : 600 seconds
  HTTP Access                : Enabled
  HTTPS Access               : Disabled
```

Example 2 [no] web-management

```
HP-2530-48G-PoEP(config)# no web-management
HP-2530-48G-PoEP(config)# show rest-interface
REST Interface - Server Configuration
REST Interface             : Enabled
  REST Operational Status    : Down
  HTTP Access                : Disabled
  HTTPS Access               : Disabled
```

1.2.1.1 [no] rest-interface

Syntax

`[no] rest-interface`

Description

Disables the REST-interface.

Example 3 [no] rest-interface

```
HP-2530-48G-PoEP(config)# no rest-interface
HP-2530-48G-PoEP(config)# show rest-interface

REST Interface - Server Configuration
  REST Interface             : Disabled
  REST Operational Status    : Down
  HTTP Access                : Enabled
  HTTPS Access               : Disabled
```

1.2.1.2 Validation rules

Validation	Error/Warning/Prompt
REST interface is disabled and a REST request is sent on HTTP port for processing.	The REST interface is disabled and requests cannot be processed.
Both REST and WEB management are enabled. An attempt is made to disable web-management.	The REST requests cannot be processed when WEB management is disabled.
Both REST and WEB management are disabled. An attempt is made to enabled the REST interface.	The REST requests cannot be processed because WEB management is disabled. Enable WEB management for the REST interface to be operational.
If redirection of the REST request fails from HTTP	Redirection of the REST request to the REST server failed
If there is no response from REST server for about 2 mins	No response from the REST server

1.2.2 rest-interface session-idle-timeout

Syntax

`rest-interface session-idle-timeout <SECONDS>`

Description

Configure session-idle-timeout for the the REST interface sessions. The configurable value range is from 120 to 7200 seconds with the default value at 600 seconds.

Usage

`[no] rest-interface session-idle-timeout`

Sets the session-idle-timeout to the default value.

1.3 Show commands

1.3.1 show rest Interface

Syntax

`show rest interface`

Description

The REST operational status will be shown as Up only when the REST interface is enabled and either WEB management or WEB management SSL is enabled. HTTP access is enabled when management is enabled. HTTPS access is enabled when WEB management SSL is enabled.

1.4 Restrictions

1. REST interface is not supported in stacking device, including VSF.
2. REST interface is not supported in FIPS.

1.5 JSON Schema

JSON/REST API on device configuration and state

- Used by application APIs to read/update configuration.
- Enabled by default. Use the command `show rest-interface` to verify status.
- Administrator can disable feature.
- This feature is disabled in stacking devices by default and can not be enabled if stacking is enabled.
- REST supports GET/POST/PUT/DELETE methods only.

More information

JSON Schema and REST API documentation for X.16.01.xxxx software: **JSON Schema**

1.5.1 Use case — login-sessions

If user is configured on the switch, login request must be posted to the switch for authentication. If user validation is successful, the switch returns a session id as a cookie that should be used for further requests.

```
WorkStation# curl --noproxy10.100.167.104 -X POST
http://10.100.167.104:80/rest/v1/login-sessions -d '{"userName":"test", "password":"test"}'
```

Response from the switch in case of successful user validation:

```
{
"uri": "/rest/v1/login-sessions","cookie":
"sessionId=09CG1bRuT5hkCPzI97mmDjpn4uLtsmgkBsAaWUr9h7GxlkbsiASak1PEyj7Ov3n"
}
```

1.5.2 Use case — authentication failure

In case of authentication failure, switch returns error:

```
WorkStation# curl --noproxy 10.100.167.104 -X POST
http://10.100.167.104:80/rest/v1/login-sessions -d '{"userName":"test", "password":"test"}'
```

Response from the switch in case of failure:

```
{
 "message": "Authentication failed."
}
```

NOTE:

- Operator user is allowed only for GET method where as manager user is allowed for all the supported methods.
- If no user name is configured, posting of login request is not required and requests can be sent without cookie.

1.5.3 Use case — creating a VLAN

Creating VLAN using REST:

```
WorkStation# curl --noproxy10.100.167.104 --cookie
"sessionId=09CG1bRuT5hkCPzI97mmDjpn4uLtsmgkBsAaWUr9h7GxlkbsiASak1PEyj7Ov3n" -X POST
http://10.100.167.104:80/rest/v1/vlans -d '{"vlan_id":5, "name":"VLAN5"}'
```

Response from the Switch:

```
{
    "uri": "/rest/v1/vlans/5",
    "vlan_id": 5,
    "name": "VLAN5",
    "status": "VS_PORT_BASED",
    "type": "VT_STATIC",
    "is_voice_enabled": false,
    "is_jumbo_enabled": false,
    "is_dsnoop_enabled": false
}
```

1.5.4 Use case — fetching the VLAN

Fetching the VLAN details using REST:

```
WorkStation# curl --noproxy10.100.167.104 --cookie
"sessionId=09CG1bRuT5hkCPzI97mmDjpn4uLtsmgkBsAaWUr9h7GxlkbsiASak1PEyj7Ov3n" -X GET
http://10.100.167.104:80/rest/v1/vlans
```

Response from the Switch:

```
{
   "collection_result": {
      "total_elements_count": 2,
      "filtered_elements_count": 2
   },
   "vlan_element": [
      {
          "uri": "/rest/v1/vlans/1",
          "vlan_id": 1,
          "name": "DEFAULT_VLAN",
          "status": "VS_PORT_BASED",
          "type": "VT_STATIC",
          "is_voice_enabled": false,
          "is_jumbo_enabled": false,
          "is_dsnoop_enabled": false
        },
        {
          "uri": "/rest/v1/vlans/5",
          "vlan_id": 5,
          "name": "VLAN5",
          "status": "VS_PORT_BASED",
          "type": "VT_STATIC",
          "is_voice_enabled": false,
          "is_jumbo_enabled": false,
          "is_dsnoop_enabled": false
        }
    ]
}
```

1.5.5 Use case — logout of session

Logout of REST session.

```
WorkStation# curl --noproxy10.100.167.104 --cookie
 "sessionId=09CG1bRuT5hkCPzI97mmDjpn4uLtsmgkBsAaWUr9h7GxlkbsiASak1PEyj7Ov3n" -X
DELETE http://10.100.167.104:80/rest/v1/login-sessions
```

Will not get any response from the switch in case of successful logout

1.6 debug rest-interface

Syntax

`debug rest-interface`

Enables debug logs for the rest-interface.

Example 4 debug rest-interface

```
HP-2920-48G(config)# debug rest-interface
0000:00:23:14.01 rest tHttpd:Received REST POST request
0000:00:23:14.08 rest tHttpd:(http_state:)http_process_post END
0000:00:23:14.15 rest tHttpd:REST request redirected to REST server
0000:00:23:14.22 rest tHttpd:Send REST request message to REST control task
0000:00:23:14.30 rest tHttpd:tHttpd is unblocked with necessary cleanup
0000:00:23:14.38 rest mrest_ctrl:Request for parse header
0000:00:23:14.44 rest mrest_ctrl:Request for parse header after method POST
```

Index