ArubaOS-Switch_REST Documentation

Release 0.1

Tomas Kubica

January 31, 2017

Contents

CHAPTER 1

Using cURL

1.1 Login

We are using Ubuntu system with cURL installed. First we need to log into switch to get token (cookie):

```
$ curl --insecure -X POST https://16.21.188.197/rest/v1/login-sessions -H "Content-Type: application/
{"uri":"/rest/v1/login-sessions","cookie":"sessionId=kC9GF2IkJjez9WixspEcPyoucwQDamodCI4TkVkAOPzkcmUe
```

As you can see JSON output is not formmated so it is hard to read. Let's install tool to make this prettier:

```
$ sudo apt-get install yajl-tools
```

Add silent mode switch to cURL and use parser for pretty output:

```
$ curl -s --insecure -X POST https://16.21.188.197/rest/v1/login-sessions -H "Content-Type: applicati
{
    "uri": "/rest/v1/login-sessions",
    "cookie": "sessionId=WSWAiijQtlbXBBAakCSBAiBZR4v9rFkb2pqEr2PI9SiiUakVNAT7aLpSg1jP7y8"
}
```

Store cookie in variable:

```
$ export cookie=sessionId=WSWAiijQtlbXBBAakCSBAiBZR4v9rFkb2pqEr2PI9SiiUakVNAT7aLpSg1jP7y8
```

Get system information:

```
$ curl -s --insecure --cookie $cookie -X GET https://16.21.188.197/rest/v1/system | json_reformat
{
    "uri": "/system",
    "name": "HP-2920-24G-PoEP",
    "location": "",
    "contact": ""
}
```

1.2 Use

Get VLAN information:

```
$ curl -s --insecure --cookie $cookie -X GET https://16.21.188.197/rest/v1/vlans | json_reformat
{
    "collection_result": {
```

```
        "total_elements_count": 1,
        "filtered_elements_count": 1
    },
    "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
        }
    ]
}
```

Get interfaces information:

```
$ curl -s --insecure --cookie $cookie -X GET https://16.21.188.197/rest/v1/ports | json_reformat
{
    "collection_result": {
        "total_elements_count": 24,
        "filtered_elements_count": 24
    },
    "port_element": [
        {
            "uri": "/rest/v1/ports/1",
            "id": "1",
            "name": "",
            "is_port_enabled": true,
            "config_mode": "PCM_AUTO",
            "trunk_mode": "PTT_NONE",
            "lacp_status": "LAS_DISABLED",
            "trunk_group": "",
            "is_flow_control_enabled": false,
            "is_dsnoop_port_trusted": false,
            "internal_mode": null
        },
        {
            "uri": "/rest/v1/ports/2",
            "id": "2",
            "name": "",
            "is_port_enabled": true,
            "config_mode": "PCM_AUTO",
            "trunk_mode": "PTT_NONE",
            "lacp_status": "LAS_DISABLED",
            "trunk_group": "",
            "is_flow_control_enabled": false,
            "is_dsnoop_port_trusted": false,
            "internal_mode": null
        },
        ...
        {
            "uri": "/rest/v1/ports/23",
            "id": "23",
            "name": "",
            "is_port_enabled": true,
```

```
            "config_mode": "PCM_AUTO",
            "trunk_mode": "PTT_NONE",
            "lacp_status": "LAS_DISABLED",
            "trunk_group": "",
            "is_flow_control_enabled": false,
            "is_dsnoop_port_trusted": false,
            "internal_mode": null
        },
        {
            "uri": "/rest/v1/ports/24",
            "id": "24",
            "name": "",
            "is_port_enabled": true,
            "config_mode": "PCM_AUTO",
            "trunk_mode": "PTT_NONE",
            "lacp_status": "LAS_DISABLED",
            "trunk_group": "",
            "is_flow_control_enabled": false,
            "is_dsnoop_port_trusted": false,
            "internal_mode": null
        }
    ]
}
```

Get one port only:

```
$ curl -s --insecure --cookie $cookie -X GET https://16.21.188.197/rest/v1/ports/1 | json_reformat
{
    "uri": "/rest/v1/ports/1",
    "id": "1",
    "name": "",
    "is_port_enabled": true,
    "config_mode": "PCM_AUTO",
    "trunk_mode": "PTT_NONE",
    "lacp_status": "LAS_DISABLED",
    "trunk_group": "",
    "is_flow_control_enabled": false,
    "internal_mode": null,
    "is_dsnoop_port_trusted": false
}
```

Change port name:

```
$ curl -s --insecure --cookie $cookie -X PUT https://16.21.188.197/rest/v1/ports/1 -d '{"name":"myTes
{
    "uri": "/rest/v1/ports/1",
    "id": "1",
    "name": "myTest",
    "is_port_enabled": true,
    "config_mode": "PCM_AUTO",
    "trunk_mode": "PTT_NONE",
    "lacp_status": "LAS_DISABLED",
    "trunk_group": "",
    "is_flow_control_enabled": false,
    "internal_mode": null,
    "is_dsnoop_port_trusted": false
}
```

You can check results in CLI:

```
HP-2920-24G-PoEP# show interfaces 1

 Status and Counters - Port Counters for port 1

  Name   : myTest
  MAC Address      : 308d99-a23e3f
  Link Status      : Down
  Totals (Since boot or last clear) :
```

CHAPTER 2

Python

2.1 Login script and getting ready

Asuming Python is by default installed in you Ubuntu 14.04 let's install pip and some dependencies we will need:

```
sudo apt-get install python-pip
sudo pip install argparse
sudo pip install requests
sudo pip install simplejson
```

As we know already we need to authenticate against ArubaOS-Switch to get cookie. We will create Python script to do so and store information in file on disk.

This is login.py:

```
import argparse
import requests
import json

# Parse inputs
parser = argparse.ArgumentParser()
parser.add_argument('--ip', type=str, help="Controller IP address")
parser.add_argument('--user', type=str, help="Username")
parser.add_argument('--password', type=str, help="Password")
args = parser.parse_args()

# Use authentication call and pass credentials
url="https://" + args.ip + "/rest/v1/login-sessions"

data = {
 "userName":args.user,
 "password":args.password
}

r = requests.post(url, data=json.dumps(data), verify=False)

# Parse token
cookie = r.json()["cookie"].split("=")[1]
print("Cookie: " + cookie)

# Prepare JSON structure with ip address and token
login = {"ip":args.ip,"cookie":cookie}
```

```
# Write JSON to file to be read by other scripts
f = open('mylogin.txt', 'w')
f.write(json.dumps(login))
f.close
```

Use it to get cookie and provide IP address and your credentials:

```
$ python login.py --ip 16.21.188.197 --user manager --password hpe
Cookie: 5qiBb0lphJynk7vfCS2Ayo6MZ0jmNaP9OzucX3m0jNRoupnR7FMsAj7YaKqQvbd
```

You can check that information has been stored in mylogin.txt file:

```
$ cat mylogin.txt
{"ip": "16.21.188.197", "cookie": "5qiBb0lphJynk7vfCS2Ayo6MZ0jmNaP9OzucX3m0jNRoupnR7FMsAj7YaKqQvbd"}
```

2.2 Give your interfaces nice random names

Have you heard about pets vs. cattle discussion? We will go against trend and give our port pets nice random names.

We will use public random word service so make sure your station has Internet access.

This is niceport.py:

```
import argparse
import requests
import json

# Read login information from file
f = open('mylogin.txt', 'r')
login = json.loads(f.readlines()[0])
f.close

# Get all ports
url="https://" + login["ip"]  + "/rest/v1/ports"

r = requests.get(url, verify=False, cookies=dict(sessionId=login["cookie"]))

randomword_url="http://randomword.setgetgo.com/get.php?len=8"

# Repeat for all ports in switch
for port in r.json()["port_element"]:
    # Get random word
    rword = requests.get(randomword_url)
    name = rword.text

    # Prepare port URL and JSON with new name
    porturi="https://" + login["ip"] + port["uri"]
    portdata={
        "name":name
    }

    # Change port name and print
    rport = requests.put(porturi, verify=False, cookies=dict(sessionId=login["cookie"]), data=json.du
    print port["id"] + " is now " + rport.json()["name"]
```

Here is output of this app:

```
$ python niceports.py
1 is now alterate
2 is now ptilinal
3 is now palmated
4 is now spicknel
5 is now mangling
6 is now subframe
7 is now Sinaitic
8 is now uncandor
9 is now wreckful
10 is now provoker
11 is now ironical
12 is now ateuchus
13 is now subspace
14 is now launcher
15 is now digitize
16 is now screwman
17 is now pycnotic
18 is now rearrive
19 is now Fossoria
20 is now encanker
21 is now purparty
22 is now masonite
23 is now maphrian
24 is now griseous
```

You can check this now in CLI:

```
HP-2920-24G-PoEP# show interfaces custom all port name

 Status and Counters - Custom Port Status


  Port   Name
  ------ ----------
  1      alterate
  2      ptilinal
  3      palmated
  4      spicknel
  5      mangling
  6      subframe
  7      Sinaitic
  8      uncandor
  9      wreckful
  10     provoker
  11     ironical
  12     ateuchus
  13     subspace
  14     launcher
  15     digitize
  16     screwman
  17     pycnotic
  18     rearrive
  19     Fossoria
  20     encanker
  21     purparty
  22     masonite
  23     maphrian
```

```
  24      griseous
```