Network automation with Google Sheets?

Ievgen Kostiukevych
Senior IP Media Technology Architect
European Broadcasting Union
Where did I plug my box?..

Seriously, dude, no idea...
*Taken from the JT-NM Tested March 2019
*Taken from the JT-NM Tested March 2019
Idea

- A centralized, but dynamic and collaborative repository for often changed switch parameters.
- A centralized, but dynamic and collaborative view of troubleshooting data from the switch, the “show” commands.
- A real time update or periodic polling of both.
- Has to support Arista switches

Possible solutions

- CLI?
- Ansible?
- Python?
  - CLI (Telnet/SSH)?
  - JSON?
  - Native API?
AIMS IP Showcase IBC 2019

September 2019

Curated by Video Services Forum vsf.tv

Python

- telnetlib
- paramiko
- netmiko
- native programmatic API library (pyeapi)

```
import getpass
import sys
import telnetlib

ip_address = "192.168.122.71"
user = 'admin'
password = 'admin'

tn = telnetlib.Telnet(ip_address)

# Connect
tn.read_until("Username: ")
tn.write(user + "\n")
tn.read_until("Password: ")
tn.write(password + "\n")
tn.write("enable\n")
tn.write("super_secure_enable_pass\n")
tn.write("show version\n")

print(tn.read_all())
```
```python
from netmiko import ConnectHandler

S1 = {
    'device_type': 'arista_eos',
    'ip': '192.168.122.72',
    'username': 'admin',
    'password': 'admin',
}

net_connect = ConnectHandler(**S1)
output = net_connect.send_command('show version')
print(output)
```
```python
import pyeapi
connect = pyeapi.client.connect(
    transport='https',
    host='192.168.22.72',
    username='admin',
    password='admin')
connectedSwitch = pyeapi.client.Node(connect)
connectedSwitch.enable('show version')
```
def googleAuthorize():
    # Sets access to Google spreadsheets and Google drive
    scope = ['https://spreadsheets.google.com/feeds',
             'https://www.googleapis.com/auth/drive']
    # Imports the JSON access token and extracts credentials
    credentials = ServiceAccountCredentials.from_json_keyfile_name(
                   args.api_key, scope)
    # Authorization with extracted credentials
    gc = gspread.authorize(credentials)
    logger.info('Google Cloud API authorization successful')
    return gc

# Initial authorization at program start
gc = googleAuthorize()

# Extracts data from the spreadsheet
switchConfigs = gc.open('config').sheet('Switch Configs')
 vlanList = switchConfigs.worksheet('Vlan List')
 vlanPorts = switchConfigs.worksheet(
              'Interfaces VLAN Allocation and Descriptions')
 interfacesMacTable = switchConfigs.worksheet('MAC addresses table')
 interfacesStatusTable = switchConfigs.worksheet('Interfaces status table')
<table>
<thead>
<tr>
<th>Interface</th>
<th>Vlan ID</th>
<th>autoNegotiateActive</th>
<th>autoNegotiateActive</th>
<th>bandwidth</th>
<th>description</th>
<th>duplex</th>
<th>interfaceType</th>
<th>lineProtocolStatus</th>
<th>linkStatus</th>
<th>MAC Address</th>
<th>Entry Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet1</td>
<td>1</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 1</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet10</td>
<td>30</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 10</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet11</td>
<td>20</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 11</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet12</td>
<td>240</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 12</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet13</td>
<td>20</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 2</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet14</td>
<td>20</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 3</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet5</td>
<td>100</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 4</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet6</td>
<td>20</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 5</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet7</td>
<td>10</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 6</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Ethernet8</td>
<td>220</td>
<td>FALSE</td>
<td>FALSE</td>
<td>0</td>
<td>Device 7</td>
<td>duplexFull</td>
<td>EebraTestPhyPort</td>
<td>up</td>
<td>connected</td>
<td>ba:ac:81:78:3e:16a</td>
<td>N/A</td>
</tr>
<tr>
<td>Management1</td>
<td>N/A</td>
<td>TRUE</td>
<td>TRUE</td>
<td>1000000000</td>
<td>DuplexFull</td>
<td>10/100/1000</td>
<td>up</td>
<td>connected</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

# First - connect to the switch and check connection and eAPI
connect = pyeapi.client.connect(  
    transport='https', host=cfg['ip'], username=cfg['username'], password=cfg['password'])
logger.info('Trying to connect to {} using provided username and password'.format(cfg['ip']))
connectedSwitch = pyeapi.client.Node.connect(  
    hostname=connectedSwitch.enable('show hostname'))
logger.info('Successfully connected to ' +  
    hostname)  
version = connectedSwitch.enable('show version')
logger.info('pp.pprint( version[0]["result"] )')

# Second - read initial data from the switch
v1ans = connectedSwitch.api('vlans')
portsDescribe = connectedSwitch.api('interfaces')

# Start executing selected tasks when continuous flag is not set
if not args.continous:
    logger.info('----------------------------------------')
    logger.info('All tasks finished. Exiting...')
What about real time update?
try:
    while args.continuous:
        logger.info('{} - str(datetime.now().replace(microsecond=0))'.format(19))
        logger.info('{} - Successfully connected to + waiting'.format(20))
        if args.vlan_list:
            try:
                createVlans()
                except gpisrael.exceptions.APIError:
                    wait('api')
                    gc = googleAuthenticated()
            if args.interfaces_description:
                try:
                    setInterfaceDescriptions()
                    except gpisrael.exceptions.APIError:
                        wait('api')
                        gc = googleAuthenticated()
            if args.interfaces_vlans:
                try:
                    setInterfaceVlans()
                    except gpisrael.exceptions.APIError:
                        wait('api')
                        gc = googleAuthenticated()
            if args.interfaces_status:
                try:
                    getInterfaceState()
                    except gpisrael.exceptions.APIError:
                        wait('api')
                        gc = googleAuthenticated()
        except KeyboardInterrupt:
            logger.info('Stopped')
Useful links

- [https://github.com/ktbyers/netmiko](https://github.com/ktbyers/netmiko)
- [https://github.com/paramiko/paramiko](https://github.com/paramiko/paramiko)
- [https://docs.python.org/3.7/library/telnetlib.html](https://docs.python.org/3.7/library/telnetlib.html)
- [https://www.gns3.com/](https://www.gns3.com/)
Thank you

Ievgen Kostiukevych
IP Media Technology Architect
European Broadcasting Union
kostiukevych@ebu.ch
+41 79 225 37 35

Thank you to our Media Partners