NETWORK UNIT TESTING
Nuts (Network Unit Testing System) with SaltStack

Urs Baumann
Network Engineering
Bern, 09. November 2017
Urs Baumann
Network Engineer
Python Developer
SDN, Cloud Infrastructure, OpenStack, Python, Cisco and Brocade Instructor

urs.baumann@ins.hsr.ch
@ubaumann_ch

http://ins.hsr.ch
@INSHSR
Agenda

- NUTS
- NAPALM
- SaltStack
Testing Network Change

- How do you test your network after a change?
  - Today
    - ping 8.8.8.8 ✓
    - facebook.com ✓
    - youtube.com ✓
    - e-mail to my private address ✓
    - print next train connection ✓

- Rest of my network life
  - Nuts (Network Unit Test System)
Nuts (Network Unit Testing System)

```yaml
-nuts$ [ubemann@localhost nuts]$ cat ../mytest.yml
name: bandwidth_ping
  command: bandwidth
  devices: "git"
  parameter: "['ip[0]']"
  operator: '<'
  expected: 100000000
setup:
  - command: network.ip_addr
    devices: "git"
    save: ip
  - command: cmd.run
    devices: "git"
    parameter: ['iperf3 -s -D -1']
teardown:
  - command: cmd.run
    parameter: ['killall iperf3']
  devices: "git"
  - name: versionK_01
    command: checkversion
    devices: "c"
    parameter: []
  operator: '='
  expected: 'CSR1000V Software (X86_64 LINUX_IOSD-UNIVERSALK9-M), Version 15.5(2)S, RELEASE SOFTWARE (fc3)'

-nuts$ [ubemann@localhost nuts]$ python main.py ../mytest.yml -c /home/ubemann/config_nuts.yml
Test Cases:
  Async Tests:
  Sync Tests:
    Name: bandwidth_ping, Command: bandwidth, Devices: "git", Parameter: "['ip[0]']", Operator: '<', Expected: 100000000
Start test versionK_01
  Started test 1 of 1
  -------------------------
  Collect result of test versionK_01
  Collected results from 1 of 1 tests
  -------------------------
  Start test bandwidth_ping

versionK_01: Test passed
bandwidth_ping: Test passed
-------------------------
Summary
5 out of 2 tests passed
```

https://github.com/HSRNetwork/Nuts

Urs Baumann, Network Unit Testing, 09.11.2017
Nuts

- Command for network devices (= < > not)
  - connectivity
  - traceroute
  - interfacestatus
  - interfacespeed
  - arp
  - checkversion
  - checkuser

- Command for debian systems (= < > not)
  - connectivity
  - traceroute
  - dnscheck
  - dhcpcheck
  - webresponse
  - portresponse
  - bandwidth

- Comming soon
  - lldp_neighbor
  - bgp_neighbor
  - ospf_neighbor
  - route_to
  - stp_root
Network Devices

**arp**

- name: example_arp
  command: arp
  devices: cisco.csr.1000v
  parameter: [192.168.16.128]
  operator: '='
  expected: '00:0C:29:EA:D1:68'

**interfacestatus**

- name: example_interfacestatus
  command: interfacestatus
  devices: cisco.csr.1000v
  parameter: [GigabitEthernet1]
  operator: '='
  expected: True
```yaml
# bandwidth
- name: bandwidth_ping
  command: bandwidth
  devices: srvlnx0001
  parameter: ['{{ ip[0] }}']
  operator: '<'
  expected: 100000000

setup:
- command: network.ip_addrs
  devices: srvlnx0099
  save: ip
- command: cmd.run
  devices: srvlnx0099
  parameter: ['iperf3 -s -D -1']

teardown:
- command: cmd.run
  parameter: ['pkill iperf3']
  devices: srvlnx0099

# dhcpcheck
- name: dhcp_test
  command: dhcpcheck
  devices: srvlnx0014
  parameter: ['10.10.10.11']
  operator: '='
  expected: True
```
Network Automation and Programmability Abstraction Layer with Multivendor support

https://napalm.readthedocs.io/
https://napalm-automation.net/

Support matrix


```python
from napalm import get_network_driver

driver = get_network_driver('ios')
device = driver('152.96.9.201', 'cisco', 'cisco')

device.open()
arp = device.get_arp_table()
device.close()

print(arp)
```
## Support Matrix

<table>
<thead>
<tr>
<th></th>
<th>EOS</th>
<th>IOS</th>
<th>IOSXR</th>
<th>JUNOS</th>
<th>NXOS</th>
<th>NXOS SSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_arp_table</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_bgp_config</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_bgp_neighbors</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_bgp_neighbors_detail</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_config</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_environment</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_facts</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_firewall_policies</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_interfaces</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_interfaces_counters</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_interfaces_ip</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_ipv6_neighbors_table</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_lldp_neighbors</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_lldp_neighbors_detail</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_mac_address_table</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_network_instances</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_ntp_peers</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_ntp_servers</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_ntp_stats</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_optics</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_probes_config</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_probes_results</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_route_to</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_snmp_information</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>get_users</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>is_alive</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>ping</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>traceroute</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>
- Python
- YAML
- Jinja2
- Easy to extend
- API
- Event bus
- Agent-based / agent-less
- Proxy minions

https://docs.salstack.com
SaltStack and NAPALM

```
root@sr-000801:~$ salt 'sw*' ntp.servers
sw8262-d:
    ---
    comment:  
    out:
    - time0.ins.hsr.ch
    - 152.96.120.60
    - 152.96.120.53
    result:  True
sw8262-e:
    ---
    comment:  
    out:
    - time0.ins.hsr.ch
    - 152.96.120.60
    - 152.96.120.53
    result:  True
sw8262-b:
    ---
    comment:  
    out:
```

```
root@sr-000801:~$ salt 'sw*' ntp.set_servers timel.ins.hsr.ch
sw8262-b:
    ---
    already_configured:  False
    comment:  
    diff:
    - ntp server timel.ins.hsr.ch
    loaded_config:  
    result:  True
sw8262-e:
    ---
    already_configured:  False
    comment:  
    diff:
    - ntp server timel.ins.hsr.ch
    loaded_config:  
    result:  True
sw8262-a:
    ---
```
Use Case

- Connectivity
- OS Version
- ARP Table

Interface checks
- ARP Table
- OS Version

DHCP Check
- Bandwidth to server
- Open ports on Server

DNS check
- Connectivity
- Traceroute

Connectivity
- Check local user

RESTful API

SALTSTACK
Summary

- NUTS
  - https://nuts.readthedocs.io/
  - https://github.com/HSRNetwork/Nuts/

- NAPALM
  - https://napalm.readthedocs.io/
  - https://napalm-automation.net/

- SaltStack
  - https://docs.saltstack.com/

- Slack https://networktocode.slack.com/
  - #nuts
  - #napalm
  - #saltstack