JSNAPy Overview

Khelil Sator
ksator@juniper.net

JSNAPy overview

- Tool to automate network verifications
- Work with all devices running Junos
- This is the Python version of JSNAP (JSNAP is written in slax)
- JSNAPy is supported in three modes
  - a command line tool
  - a Python module
  - An Ansible module hosted on the Ansible Galaxy website (https://galaxy.ansible.com/Juniper/junos/)

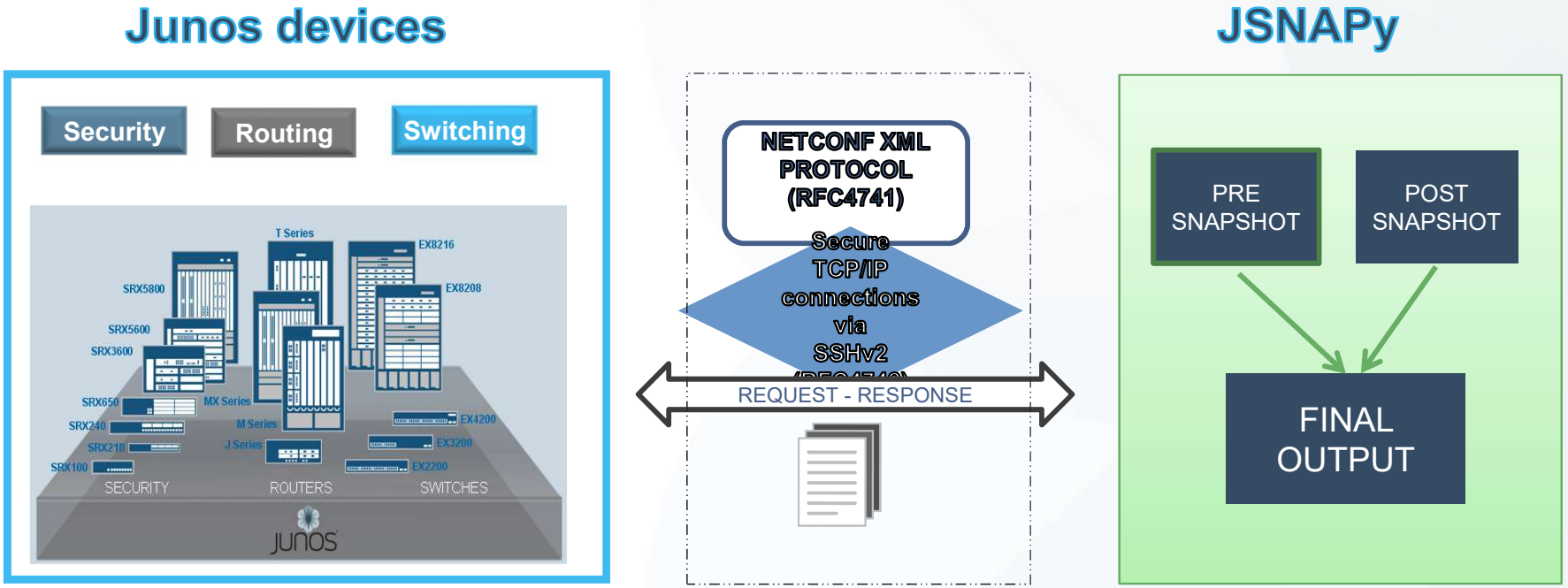
© 2016 Juniper Networks, Inc. All rights reserved.

JSNAPy overview

- JSNAPy communicates with Junos devices using PyEZ (junos_eznc python library, so: NETCONF, SSH, XML, RPC)
- Tool to take snapshots, store snapshots, compare snapshots
- 2 workflows:
  - take pre snapshots before any modification and then take post snapshots after modification and then compare them based on test cases
  - take snapshots and compare them against pre defined criteria

© 2016 Juniper Networks, Inc. All rights reserved.

JSNAPy architecture



Take, store and compare snapshots

© 2016 Juniper Networks, Inc. All rights reserved.

JSNAPy features

- snap: Take snapshot of devices for the given show commands/RPCs
- check: Compare two snapshots based on tests
- diff: Compare two snapshots word by word without any predefined test cases and criteria
- snapcheck: Take a snapshot of devices for the given show commands/RPCs and compare it against pre-defined criteria

© 2016 Juniper Networks, Inc. All rights reserved.

JSNAPy features

- Operational state verifications and configuration verifications as well
- Supports RPCs with arguments for taking snapshots
- Connects to multiple devices
- Uses YAML files

© 2016 Juniper Networks, Inc. All rights reserved.

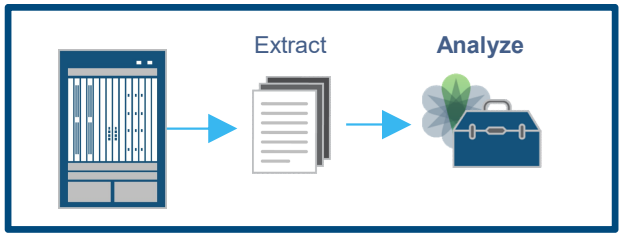
JSNAPy comparison operators

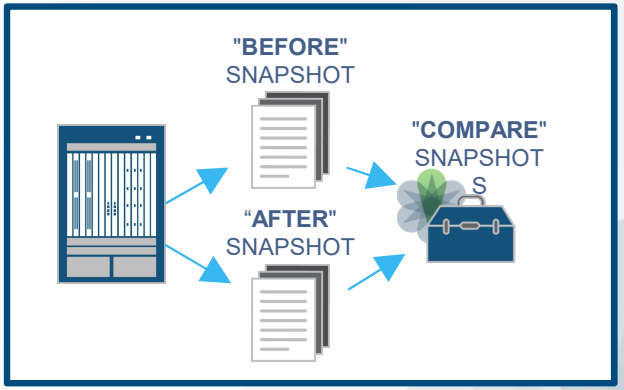
DIRECT

- Is equal
- Is not equal
- Is less than
- Is greather than
- Is in this list
- Is not in this list
- Is in this range
- Is not in this range
- Exists
- Does not exists
- Contains

COMPARISON

- Same-list
- List-not-less
- List-not-more
- no-diff
- all-same
- delta
- Regex





© 2016 Juniper Networks, Inc. All rights reserved.

test file for the snap-snap-check workflow

```
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ more testfiles/test_file_check_bgp_states.yml
#tests_include:
#  - bgp_neighbor

bgp_neighbor:
  - command: show bgp neighbor
  - ignore-null: True
# - rpc: get-bgp-neighbor-information
  - iterate:
#     xpath: '/bgp-information/bgp-peer'
      xpath: '//bgp-peer'
      id: './peer-address'
      tests:
        - no-diff: peer-state
          err: "Test Failed!! state of the peer <{{pre['peer-address']}}> is not the same. it was <{{pre['peer-state']}}>. it is <{{post['peer-state']}}> "
          info: "Test succeeded!! state of the peer <{{pre['peer-address']}}> is the same"

        - no-diff: flap-count
          err: "Test Failed!! flap-count of the peer <{{pre['peer-address']}}> is not the same. it was <{{pre['flap-count']}}>. it is <{{post['flap-count']}}> "
          info: "Test succeeded!! flap-count of the peer <{{pre['peer-address']}}> is the same. it is  <{{post['flap-count']}}>"
```

test file for the snapcheck workflow:

```
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ more testfiles/test_file_snapcheck_bgp_states.yml
tests_include:
  - snapcheck_bgp_summary
  - snapcheck_bgp_neighbor

snapcheck_bgp_summary:
# - command: show bgp summary
  - rpc: get-bgp-summary-information
  - item:
      xpath: '/bgp-information'
      tests:
        - is-equal: down-peer-count, 0
          err: "Test Failed!! down-peer-count is not equal to 0. it is equal to <{{post['down-peer-count']}}>"
          info: "Test succeeded!! down-peer-count is equal to <{{post['down-peer-count']}}>"
  - item:
      xpath: '//bgp-rib'
      tests:
        - in-range: total-prefix-count, 1, 100
          err: "Test Failed!! value of total-prefix-count is not in range of 1-100, it is equal to <{{post['total-prefix-count']}}>"
          info: "Test succeeded!! value of total-prefix-count is in range of 1-100, it is equal to <{{post['total-prefix-count']}}>"
  - iterate:
      xpath: '//bgp-peer/bgp-rib'
      tests:
        - in-range: accepted-prefix-count, 1, 50
          err: "Test Failed!! value of accepted-prefix-count is not in range of 1-50, it is equal to <{{post['accepted-prefix-count']}}>"
          info: "Test succeeded!! value of total-prefix-count is in range of 1-50, it is equal to <{{post['accepted-prefix-count']}}>"
  - iterate:
      xpath: //bgp-information/bgp-peer[normalize-space(peer-state)='Established']
      tests:
        - is-gt: elapsed-time/@seconds, 36
          err:  "BGP peer <{{ post['peer-address']}}> has a very small uptime of <{{ post['elapsed-time/@seconds']}}>"
          info: "BGP peer <{{ post['peer-address']}}> state is <{{ post['peer-state']}}> and has an uptime of <{{ post['elapsed-time/@seconds'] }}>"
```

Map devices to test files

```
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ more cfg_file_check_bgp_states.yml
    hosts:
      - include: devices.yml
        group: EX4300
    tests:
      - test_file_check_bgp_states.yml
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ more cfg_file_snapcheck_bgp_states.yml
    hosts:
      - include: devices.yml
        group: EX4300
    tests:
      - test_file_snapcheck_bgp_states.yml
```

```
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ more testfiles/devices.yml
MX:
  - 172.30.177.170:
      username: pytraining
      passwd: Poclab123
QFX:
  - 172.30.179.239:
      username: pytraining
      passwd: Poclab123
EX4200:
  - 172.30.179.107:
```

Execute the snapcheck workflow

```
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ jsnapy --snapcheck -f cfg_file_snapcheck_bgp_states.yml
Connecting to device 172.30.179.95 ................
Taking snapshot of RPC: get-bgp-summary-information
Taking snapshot of COMMAND: show bgp neighbor
*************************** Device: 172.30.179.95 ****************************
Tests Included: snapcheck_bgp_summary
*************************RPC is get-bgp-summary-information*********************
PASS | All "down-peer-count" is equal to "0" [ 1 matched ]
PASS | All "total-prefix-count" is in range "1.000000 - 100.000000" [ 1 matched ]
PASS | All "accepted-prefix-count" is in range "1.000000 - 50.000000" [ 2 matched ]
PASS | All "elapsed-time/@seconds" is greater than 36" [ 2 matched ]
Tests Included: snapcheck_bgp_neighbor
************************** Command: show bgp neighbor ************************
PASS | All "peer-state" is equal to "Established" [ 2 matched ]
PASS | All "peer-as" is in range "100.000000 - 900.000000" [ 2 matched ]
------------------------------ Final Result!! ------------------------------
snapcheck_bgp_neighbor : Passed
snapcheck_bgp_summary : Passed
Total No of tests passed: 6
Total No of tests failed: 0
Overall Tests passed!!!
Connecting to device 172.30.179.74 ................
Taking snapshot of RPC: get-bgp-summary-information
Taking snapshot of COMMAND: show bgp neighbor
*************************** Device: 172.30.179.74 ****************************
Tests Included: snapcheck_bgp_summary
*************************RPC is get-bgp-summary-information*********************
PASS | All "down-peer-count" is equal to "0" [ 1 matched ]
PASS | All "total-prefix-count" is in range "1.000000 - 100.000000" [ 1 matched ]
```

Execute the snap-snap-check workflow

```
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ jsnapy --snap pre -f cfg_file_check_bgp_states.yml
Connecting to device 172.30.179.95 ................
Taking snapshot of COMMAND: show bgp neighbor
Connecting to device 172.30.179.74 ................
Taking snapshot of COMMAND: show bgp neighbor
Connecting to device 172.30.179.73 ................
Taking snapshot of COMMAND: show bgp neighbor
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ jsnapy --snap post -f cfg_file_check_bgp_states.yml
Connecting to device 172.30.179.95 ................
Taking snapshot of COMMAND: show bgp neighbor
Connecting to device 172.30.179.74 ................
Taking snapshot of COMMAND: show bgp neighbor
Connecting to device 172.30.179.73 ................
Taking snapshot of COMMAND: show bgp neighbor
ksator@ubuntu:~/junos-verifications-automation-with-jsnapy$ jsnapy --check pre post -f cfg_file_check_bgp_states.yml
**************************** Device: 172.30.179.95 ****************************
Tests Included: bgp_neighbor
************************* Command: show bgp neighbor *************************
PASS | All "peer-state" is same in pre and post snapshot [ 2 matched ]
PASS | All "flap-count" is same in pre and post snapshot [ 2 matched ]
------------------------------ Final Result!! ------------------------------
bgp_neighbor : Passed
Total No of tests passed: 2
Total No of tests failed: 0
Overall Tests passed!!!
**************************** Device: 172.30.179.74 ****************************
Tests Included: bgp_neighbor
************************* Command: show bgp neighbor *************************
PASS | All "peer-state" is same in pre and post snapshot [ 2 matched ]
PASS | All "flap-count" is same in pre and post snapshot [ 2 matched ]
------------------------------ Final Result!! ------------------------------
bgp_neighbor : Passed
Total No of tests passed: 2
Total No of tests failed: 0
```

snap-snap-check workflow with PYTHON

```
from jnpr.jsnapy import SnapAdmin
from pprint import pprint

# instanciate the class SnapAdmin
js = SnapAdmin()

# the variable config_file refers to the jsnapy configuration file
config_file = "cfg_file_check_bgp_states.yml"

# taking first snapshots using jsnapy
# Performing function similar to --snap
print "taking first snapshots using jsnapy"
js.snap(config_file, "pre")
# jsnapy closed the connection after the snapshot.

# this is where you are supposed to apply your configuration changes

# taking second snapshot using jsnapy
# Performing function similar to --snap
print "taking second snapshots using jsnapy"
js.snap(config_file, "post")
# jsnapy closed the connection after the snapshot.

# Performing function similar to --check
# comparing first and second snapshots using jsnapy. and printing the result.
# sending slack notifications
print "comparing first and second snapshots using jsnapy"
chk = js.check(config_file, "pre", "post")
#for check in chk:
#     print "Tested on", check.device
#     print "Final result: ", check.result
#     print "Total passed: ", check.no_passed
#     print "Total failed:", check.no_failed
#     print check.test_details
#     pprint(dict(check.test_details))
```

snapcheck workflow with PYTHON

```
# performing function similar to --snapcheck option in command line
# usage of jsnapy python library without pyez.

from jnpr.jsnapy import SnapAdmin

# instanciate the class SnapAdmin
js = SnapAdmin()

# the variable config_file refers to the jsnapy configuration file
config_file = "cfg_file_snapcheck_bgp_states.yml"

# Performing function similar to --snapcheck
# taking a snapshot (called snap_from_python) and comparing it against predefined criteria
snapvalue = js.snapcheck(config_file, "snap_from_python")
# jsnapy closed the connection after the snapshot.

# printing the result
print "comparing snapshots against predefined criteria using jsnapy"
for snapcheck in snapvalue:
#     print "\n -----------snapcheck----------"
#     print "Tested on", snapcheck.device
#     print "Final result: ", snapcheck.result
#     print "Total passed: ", snapcheck.no_passed
#     print "Total failed:", snapcheck.no_failed
#     pprint(dict(snapcheck.test_details))
    if snapcheck.no_failed != 0:
     print "this device failed some tests: " + snapcheck.device
```

JSNAPy tutorial and demo

- https://github.com/ksator/junos-verifications-automation-with-jsnapy
- https://github.com/ksator/junos-verifications-automation-with-jsnapy/wiki

© 2016 Juniper Networks, Inc. All rights reserved.

JSNAPy documentation

- Source code: https://github.com/Juniper/jsnapy
- Documentation: https://github.com/Juniper/jsnapy/wiki
- Samples: https://github.com/Juniper/jsnapy/tree/master/samples
- Book: http://forums.juniper.net/t5/Day-One-Books/Day-One-Enabling-Automated-Network-Verifications-with-JSNAPy/ba-p/302104
- JSNAPY Guide: https://www.juniper.net/techpubs/en_US/junos-snapshot1.0/information-products/pathway-pages/junos-snapshot-python.pdf
- Videos:
  - JSNAPy Overview: https://www.youtube.com/watch?v=t7oGEbfdCt8
  - JSNAPy Tutorial (Detailed): https://www.youtube.com/watch?v=it4HxJq0jR0
  - Network Automation using Ansible & JSNAPy: https://www.youtube.com/watch?v=lv7lh3kwKns

© 2016 Juniper Networks, Inc. All rights reserved.


Thank you