Junos Ansible Modules Documentation

Release 1.4.3

Jeremy Schulman - Juniper Networks, Inc.

Oct 09, 2017

Contents

Contents:

CHAPTER 1

junos_pmtud - Perform path MTU discovery on junos devices

Author Martin Komon

Synopsis

New in version 2.4.

perform path MTU discovery on junos devices

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
# Simple example
  tasks:
    - name: "Check MTU on backup circuit"
      junos_pmtud:
        host={{ junos_host }}
```

```
        port={{ netconf_port }}
        user={{ ansible_ssh_user }}
        passwd={{ ansible_ssh_pass }}
        dest_ip=8.8.8.8

# Using more parameters
  tasks:
    - name: "Check MTU on backup circuit"
      junos_pmtud:
        host={{ junos_host }}
        port={{ netconf_port }}
        user={{ ansible_ssh_user }}
        passwd={{ ansible_ssh_pass }}
        dest_ip=8.8.8.8
        routing_instance=internet
        max_range=128
```

CHAPTER 2

junos_rollback - Rollback configuration of device

Author Rick Sherman, Juniper Networks

Synopsis

New in version 1.2.0.

Rollback the configuration of a device running Junos

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
- junos_rollback:
   host: "{{ inventory_hostname }}"
   logfile=rollback.log
   diffs_file=rollback.diff
   rollback=1
```

```
    comment="Rolled back by Ansible"
    confirm=5

# over console server connection using PyEZ >= 2.0
- junos_rollback:
    host: "{{ inventory_hostname }}"
    logfile=rollback.log
    diffs_file=rollback.diff
    rollback=1
    comment="Rolled back by Ansible"
    confirm=5
    mode='telnet'
    port=7015
```

CHAPTER 3

junos_install_config - Load a configuration file or snippet onto a device running Junos OS.

Author Jeremy Schulman, Juniper Networks

Synopsis

New in version 1.0.0.

Load a complete Junos OS configuration (overwrite) or merge a configuration snippet onto a device running Junos OS and commit it. The default behavior is to perform a **load merge** operation (overwrite='no'). This module performs an atomic lock/edit/unlock. If the process fails at any step, then all configuration changes are discarded. You can load the configuration using either NETCONF or the CONSOLE port. Specify the *console* option to use the CONSOLE port. You provide the configuration data in a file. Supported formats when using NETCONF include ASCII text, Junos XML elements, and Junos OS **set** commands. Configurations performed through the console must only use ASCII text formatting.

Options

Note: Requires junos-eznc >= 2.1.1

Note: Requires junos-netconify >= 1.0.1, when using the *console* option

Examples

```
# load merge a change to the Junos OS configuration using NETCONF

- junos_install_config:
    host={{ inventory_hostname }}
    file=banner.conf

# load overwrite a new Junos OS configuration using the CONSOLE port

- junos_install_config:
    host={{ inventory_hostname }}
    console="--telnet={{TERMSERV}},{{TERMSERV_PORT}}"
    file=default_new_switch.conf
    overwrite=yes

# load merge a change to the Junos OS configuration using NETCONF and supplying a␣
↪commit log message
- junos_install_config:
    host={{ inventory_hostname }}
    file=banner.conf
    comment="configured by ansible"

# load replace a change to the Junos OS configuration using NETCONF
- junos_install_config:
    host={{ inventory_hostname }}
    file=snmp.conf
    replace=yes

# Install/load config using console server connection using PyEZ >= 2.0

- junos_install_config:
    host={{ inventory_hostname }}
    port=7016
    mode='telnet'
    file=default_new_switch.conf
```

CHAPTER 4

junos_shutdown - Shut down or reboot a device running Junos OS.

Author Jeremy Schulman, Juniper Networks

Synopsis

New in version 1.0.0.

Shut down (power off) or reboot a device running Junos OS. This includes all Routing Engines in a Virtual Chassis or a dual Routing Engine system. This is equivalent to executing either the Junos OS **request system power-off** or **request system reboot** operational command.

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
- junos_shutdown:
    host={{ inventory_hostname }}
    shutdown="shutdown"
```

```
    reboot=yes

# over console server connection using PyEZ >= 2.0
- junos_shutdown:
    host={{ inventory_hostname }}
    shutdown="shutdown"
    reboot=yes
    mode='telnet'
    port=7016
```

CHAPTER 5

junos_zeroize - Erase all data, including configuration and log files, on a device running Junos OS.

Author Jeremy Schulman, Juniper Networks

Synopsis

New in version 1.0.0.

Execute the Junos OS **request system zeroize** command to remove all configuration information on the Routing Engines and reset all key values on a device running Junos OS. The command removes all data files, including customized configuration and log files, by unlinking the files from their directories. The command also removes all user-created files from the system including all plain-text passwords, secrets, and private keys for SSH, local encryption, local authentication, IPsec, RADIUS, TACACS+, and SNMP. This command reboots the device and sets it to the factory default configuration. After the reboot, you must log in through the console as root in order to access the device.

Options

Note: Requires junos-eznc >= 1.2.2

Note: Requires junos-netconify >= 1.0.1, when using the *console* option

Examples

```
- junos_zeroize:
    host={{ inventory_hostname }}
    zeroize="zeroize"

# over console server connection using PyEZ >= 2.0
- junos_zeroize:
    host={{ inventory_hostname }}
    zeroize="zeroize"
    port=7011
    mode="telnet"
```

Note: You **MUST** either use the *host* option or the *console* option to designate how the device is accessed.

CHAPTER 6

junos_jsnapy

Author Roslan Zaki & Damien Garros, Juniper Networks

Synopsis

New in version 1.4.0.

Execute JSNAPy test from Ansible. Attention, to not break Ansible behavior, this module only report "failed" if the module itself fails, not if a test fails. To check the test results you need to subscribe to the result and assert the returned value. An experimental Callback_Plugin for junos_jsnapy is available to provide additional information about tests that failed. To enable it, you need to add "callback_whitelist = jsnapy" in your ansible configuration file.

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
   - name: JUNOS Post Checklist
     junos_jsnapy:
       host: "{{ inventory_hostname}}"
       passwd: "{{ tm1_password }}"
       action: "snap_post"
       config_file: "first_test.yml"
       logfile: "migration_post.log"
     register: test1

   - name: Check JSNAPy tests results
     assert:
     that:
       - "test1.passPercentage == 100"

   - name: Debug jsnapy
     debug: msg=test1

---------
   - name: Test based on a test_file directly
     junos_jsnapy:
       host: "{{ junos_host }}"
       port: "{{ netconf_port }}"
       user: "{{ ansible_ssh_user }}"
       passwd: "{{ ansible_ssh_pass }}"
       test_files: tests/test_junos_interface.yaml
       action: snapcheck
     register: test1

   - name: Check JSNAPy tests results
     assert:
     that:
       - "test1.passPercentage == 100"
---------
   - name: "Collect Pre Snapshot"
    junos_jsnapy:
      host: "{{ junos_host }}"
      port: "{{ netconf_port }}"
      user: "{{ ansible_ssh_user }}"
      passwd: "{{ ansible_ssh_pass }}"
      test_files: tests/test_loopback.yml
      action: snap_pre
    register: test_pre

---------
  - name: "Collect Post Snapshot"
    junos_jsnapy:
      host: "{{ junos_host }}"
      port: "{{ netconf_port }}"
      user: "{{ ansible_ssh_user }}"
      passwd: "{{ ansible_ssh_pass }}"
      test_files: tests/test_loopback.yml
      action: snap_post
    register: test_post

---------
  - name: "Check after PRE - POST check"
```

```
    junos_jsnapy:
      host: "{{ junos_host }}"
      port: "{{ netconf_port }}"
      user: "{{ ansible_ssh_user }}"
      passwd: "{{ ansible_ssh_pass }}"
      test_files: tests/test_loopback.yml
      action: check
    register: test_check

    - name: Check Results
      assert:
        that:
          - test_check|succeeded
          - test_check.passPercentage == 100
```

CHAPTER 7

junos_commit - Execute commit on device

Author Rick Sherman, Juniper Networks

Synopsis

New in version 1.2.0.

Execute a Commit on a device running Junos independently of loading a configuration

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
- junos_commit:
   host: "{{ inventory_hostname }}"
   logfile=changes.log
   comment="Non load commit"
```

```
# over console server connection using PyEZ >= 2.0
- name: junos commit
  junos_commit:
    host={{ inventory_hostname }}
    port=7016
    mode='telnet'
    comment="commit with console connection via PyEZ"
```

CHAPTER 8

junos_get_table - Retrieve data from a Junos device using Tables/Views

Author Jason Edelman (@jedelman8)

Synopsis

New in version 1.9.0.

Retrieve data from a Junos device using Tables/Views

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
# GET NEIGHBOR INFO USING STD LLDP TABLE
- junos_get_table: table=LLDPNeighborTable file=lldp.yml host={{ inventory_hostname }}
↪ user={{ un }} passwd={{ pwd }}

# GET NEIGHBOR INFO USING CUSTOM LLDP TABLE IN CUSTOM PATH
```

```
- junos_get_table: table=NTCNeighborTable path=tables/ file=ntclldp.yaml host={{␣
↪inventory_hostname }} user={{ un }} passwd={{ pwd }}

#
- name: Table/View example via console server connection using PyEZ >= 2.0
  junos_get_table:
    table=RouteTable
    file=routes.yml
    host={{ inventory_hostname }}
    port=7016
    mode='telnet'
```

CHAPTER 9

junos_cli - Execute CLI on device and save the output locally

Author Damien Garros, Juniper Networks

Synopsis

New in version 1.2.0.

Execute CLI on device and save the output locally on a file

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
- junos_cli:
   host: "{{ inventory_hostname }}"
   cli: "show chassis hardware"
   logfile: cli.log
   dest: "{{ inventory_hostname }}.xml"
```

```
   format: xml

# Run cli over console server connection using PyEZ >= 2.0
- junos_cli:
    cli="show chassis hardware"
    host={{ inventory_hostname }}
    port=7001
    mode='telnet'
    dest="{{ inventory_hostname }}.xml"
    format='xml'
```

CHAPTER 10

junos_install_os - Install a Junos OS image.

Author Jeremy Schulman, Juniper Networks

Synopsis

New in version 1.0.0.

Install a Junos OS image on one or more Routing Engines. This module supports installations on single Routing Engine devices, MX Series routers with dual Routing Engines, and EX Series switches in a non-mixed Virtual Chassis. This action is equivalent to performing the Junos OS **request system software add** operational command. If the existing Junos OS version matches the desired version, no action is performed, and the "changed" attribute reports False. If the existing version does not match, then the module performs the following actions (1) Computes the MD5 checksum of the package located on the server. (2) Copies the Junos OS software package to the device running Junos OS. (3) Computes the MD5 checksum on the device running Junos OS and compares the two. (4) Installs the Junos OS software package. (5) Reboots the device (default). Running the module in check mode reports whether the current Junos OS version matches the desired version.

Options

Note: Requires py-junos-eznc >= 1.2.2

Examples

```
- junos_install_os:
    host={{ inventory_hostname }}
    version=12.1X46-D10.2
    package=/usr/local/junos/images/junos-vsrx-12.1X46-D10.2-domestic.tgz
    logfile=/usr/local/junos/log/software.log
```

CHAPTER 11

junos_get_config - Retrieve configuration of device

Author Rick Sherman, Juniper Networks

Synopsis

New in version 1.2.0.

Retrieve the configuration of a device running Junos and save it to a file. **Note** unicode chars will be converted to '??' as also done in PyEZ

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
- junos_get_config:
   host: "{{ inventory_hostname }}"
   logfile: get_config.log
   dest: "{{ inventory_hostname }}.xml"
```

```
    format: xml
    filter: "interfaces"
    options: {inherit: inherit, groups: groups}

# over console server connection using PyEZ >= 2.0
- junos_get_config:
    host: "{{ inventory_hostname }}"
    logfile: get_config.log
    dest: "{{ inventory_hostname }}.xml"
    port: 7016
    mode: 'telnet'
    format: xml
```

CHAPTER 12

junos_rpc - run given rpc

Author Nitin Kumar, Juniper Networks

Synopsis

New in version 1.9.

run given rpc

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
# retrieve rpc response using NETCONF

- junos_rpc:
  host={{ inventory_hostname }}
  rpc=get-interface-information
```

```
  dest=get_interface_information.conf
  register=junos

- junos_rpc:
  host={{ inventory_hostname }}
  rpc=get-interface-information
  kwargs="interface_name=em0"
  format=xml/text/json
  dest=get_interface_information.conf
  register=junos

# Example to fetch device configuration
- name: Get Device Configuration
  junos_rpc:
    host={{ inventory_hostname }}
    rpc=get-config
    dest=get_config.conf

# Fetch configuration over console server connection using PyEZ >= 2.0
- name: Get Device Configuration
  junos_rpc:
    host={{ inventory_hostname }}
    port=7005
    mode='telnet'
    rpc=get-config
    dest=get_config.conf

# Example to fetch device configuration
- name: Get Device Configuration for interface
  junos_rpc:
    host={{ inventory_hostname }}
    rpc=get-config
    filter_xml="<configuration><interfaces/></configuration>"
    dest=get_config.conf
  register: junos

# Example to fetch configuration in json for >=14.2
# and use it with rpc_reply
- name: Get Device Configuration
  hosts: all
  roles:
    - Juniper.junos
  connection: local
  gather_facts: no
  tasks:
    - name: Get interface information
      junos_rpc:
        host: "{{ inventory_hostname }}"
        rpc: get-interface-information
        kwargs:
          interface_name: em0
          media: True
        format: json
        dest: get_interface_information.conf
      register: junos

    - name: Print configuration
      debug: msg="{{ junos.rpc_reply }}"
```

CHAPTER 13

junos_srx_cluster - Create an srx chassis cluster for cluster capable srx running Junos OS.

Author Patrik Bok, Ashley Burston, Rick Sherman, Juniper Networks

Synopsis

New in version 1.2.0.

Create an srx chassis cluster and reboot the device. The device must be capable of forming an srx cluster and have the correct cables installed.

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
-junos_srx_cluster:
  host={{ inventory_hostname }}
  console="--port={{ serial }}"
  user=rick
  passwd=password123
  cluster_enable=true
  logfile=cluster.log
  cluster_id={{ cluster_id }}
  node={{ node_id }}

-junos_srx_cluster:
  host={{ inventory_hostname }}
  user=rick
  passwd=password123
  cluster_enable=false
  logfile=cluster.log

# over console server connection using PyEZ >= 2.0
-junos_srx_cluster:
  host={{ inventory_hostname }}
  user=rick
  passwd=password123
  mode="telnet"
  port=7032
  cluster_enable=true
  logfile=cluster.log
```

CHAPTER 14

junos_ping - execute ping on junos devices

Author Damien Garros, Juniper Networks

Synopsis

New in version 1.3.1.

execute ping on junos devices

Options

Note: Requires junos-eznc >= 1.2.2

Examples

```
# Simple example
  tasks:
    - name: "Execute ping peer"
      junos_ping:
        host={{ junos_host }}
```

```
        port={{ netconf_port }}
        user={{ ansible_ssh_user }}
        passwd={{ ansible_ssh_pass }}
        dest_ip=8.8.8.8

# ping over console server connection using PyEZ >= 2.0
  tasks:
    - name: "Execute ping peer"
      junos_ping:
        host={{ inventory_hostname }}
        port=2011
        mode='telnet'
        user={{ ansible_ssh_user }}
        passwd={{ ansible_ssh_pass }}
        dest_ip=8.8.8.8

# Using loop and more parameters
  tasks:
    - name: "Execute ping peer"
      junos_ping:
        host={{ junos_host }}
        port={{ netconf_port }}
        user={{ ansible_ssh_user }}
        passwd={{ ansible_ssh_pass }}
        dest_ip={{ item.peer_ip }}
        source_ip={{ item.local_ip }}
        do_not_fragment=True
        ttl=1
      with_items: "{{underlay.neighbors}}"
```

CHAPTER 15

junos_get_facts - Retrieve facts for a device running Junos OS.

Author Jeremy Schulman, Juniper Networks

Synopsis

New in version 1.0.0.

Retrieve facts for a device running Junos OS, which includes information such as the serial number, product model, and Junos OS version. The module supports using both NETCONF and CONSOLE-based retrieval and returns the information as a JSON dictionary. The information is similar to facts gathered by other IT frameworks.

Options

Note: Requires junos-eznc >= 1.2.2

Note: Requires junos-netconify >= 1.0.1, when using the *console* option

Examples

```
# retrieve facts using NETCONF, assumes ssh-keys

- junos_get_facts: host={{ inventory_hostname }}
  register: junos

# retrieve facts using CONSOLE, assumes Amnesiac system
# root login, no password

- junos_get_facts:
    host={{ inventory_hostname }}
    user=root
    console="--telnet={{TERMSERV}},{{TERMSERVPORT}}"
    savedir=/usr/local/junos/inventory
  register: junos

# access the facts

- name: version
  debug: msg="{{ junos.facts.version }}"

# retrieve facts using console server connection using PyEZ >= 2.0

- junos_get_facts:
    host={{ inventory_hostname }}
    mode="telnet"
    port=7016
  register: junos

# access the facts

- name: version
  debug: msg="{{ junos.facts }}"
```