
Palo Alto Networks Ansible Galaxy Role

Release 2.1.0

Palo Alto Networks

Feb 11, 2020

CONTENTS:

- 1 Role / Collection Compatibility** **3**
- 2 Installation - Collection (Recommended)** **5**
- 3 Installation - Role** **7**
 - 3.1 Examples 7
 - 3.2 Module Reference 12
 - 3.3 Release History 138
 - 3.4 Contributing to PANW Ansible modules 151
 - 3.5 Developing Palo Alto Networks Ansible Modules 151
 - 3.6 Authors 152
 - 3.7 License 152
- 4 Indices and tables** **157**

The Palo Alto Networks Ansible Galaxy role is a collection of modules that automate configuration and operational tasks on Palo Alto Networks Next Generation Firewalls (both physical and virtualized) and Panorama. The underlying protocol uses API calls that are wrapped within the Ansible framework.

This is a **community supported project**. You can find the community supported live page at <https://live.paloaltonetworks.com/ansible>.

ROLE / COLLECTION COMPATIBILITY

The Palo Alto Networks PAN-OS Ansible modules were previously distributed as an Ansible Galaxy role (<https://galaxy.ansible.com/paloaltonetworks/paloaltonetworks>). Since Ansible 2.9, RedHat has urged developers to migrate to *collections* to organize and distribute their integrations. The new collection can be found here: <https://galaxy.ansible.com/paloaltonetworks/panos>

The 1.0 version of this collection is a straight port of the Ansible Galaxy role v2.4.0. If you are using Ansible 2.9 or later and you are using the role, then you can safely use this instead with no change in functionality. Just specify the *collections* spec (as mentioned above in the Usage section), remove *PaloAltoNetworks.paloaltonetworks* from the *roles* spec, and you're done!

Now that the collection is live, no new features will be added to the role. All active development will take place on the collection moving forward. Users are encouraged to upgrade to Ansible 2.9 and start using the new collection to stay up-to-date with features and bug fixes.

INSTALLATION - COLLECTION (RECOMMENDED)

(For Ansible >= v2.9)

Install the collection using *ansible-galaxy*:

```
ansible-galaxy collection install paloaltonetworks.panos
```

Then in your playbooks you can specify that you want to use the *panos* collection like so:

```
collections:  
  - paloaltonetworks.panos
```

- Ansible Galaxy: <https://galaxy.ansible.com/PaloAltoNetworks/panos>
- GitHub repo: <https://github.com/PaloAltoNetworks/pan-os-ansible>

INSTALLATION - ROLE

(For Ansible < v2.9)

Install the collection using *ansible-galaxy*:

```
ansible-galaxy install PaloAltoNetworks.paloaltonetworks
```

To upgrade your existing role, add in the additional *-f* parameter to the above command.

Then in your playbooks you can specify that you want to use the *paloaltonetworks* role like so:

```
roles:  
  - role: PaloAltoNetworks.paloaltonetworks
```

- Ansible Galaxy: <https://galaxy.ansible.com/PaloAltoNetworks/paloaltonetworks>
- GitHub repo: <https://github.com/PaloAltoNetworks/ansible-pan>

3.1 Examples

Note: You can see complete examples [here](#)

3.1.1 Add security policy to Firewall or Panorama

Security policies allow you to enforce rules and take action, and can be as general or specific as needed. The policy rules are compared against the incoming traffic in sequence, and because the first rule that matches the traffic is applied, the more specific rules must precede the more general ones.

Firewall

```
- name: Add test rule 1 to the firewall  
  panos_security_rule:  
    provider: '{{ provider }}'  
    rule_name: 'Ansible test 1'  
    description: 'An Ansible test rule'  
    source_zone: ['internal']  
    destination_zone: ['external']  
    source_ip: ['1.2.3.4']  
    source_user: ['any']  
    destination_ip: ['any']  
    category: ['any']
```

(continues on next page)

(continued from previous page)

```
application: ['any']
service: ['service-http']
hip_profiles: ['any']
action: 'allow'
commit: 'False'
```

Panorama

```
- name: Add test pre-rule to Panorama
  panos_security_rule:
    provider: '{{ provider }}'
    rule_name: 'Ansible test 1'
    description: 'An Ansible test pre-rule'
    source_zone: ['internal']
    destination_zone: ['external']
    source_ip: ['1.2.3.4']
    source_user: ['any']
    destination_ip: ['any']
    category: ['any']
    application: ['any']
    service: ['service-http']
    hip_profiles: ['any']
    action: 'allow'
    device_group: 'DeviceGroupA'
    commit: False
```

3.1.2 Add NAT policy to Firewall or Panorama

If you define Layer 3 interfaces on the firewall, you can configure a Network Address Translation (NAT) policy to specify whether source or destination IP addresses and ports are converted between public and private addresses and ports. For example, private source addresses can be translated to public addresses on traffic sent from an internal (trusted) zone to a public (untrusted) zone. NAT is also supported on virtual wire interfaces.

Firewall

```
- name: Add the service object to the firewall first
  panos_service_object:
    provider: '{{ provider }}'
    name: 'service-tcp-221'
    protocol: 'tcp'
    destination_port: '221'
    description: 'SSH on port 221'
    commit: false

- name: Create dynamic NAT rule on the firewall
  panos_nat_rule:
    provider: '{{ provider }}'
    rule_name: 'Web SSH inbound'
    source_zone: ['external']
    destination_zone: 'external'
```

(continues on next page)

(continued from previous page)

```

source_ip: ['any']
destination_ip: ['10.0.0.100']
service: 'service-tcp-221'
snat_type: 'dynamic-ip-and-port'
snat_interface: ['ethernet1/2']
dnat_address: '10.0.1.101'
dnat_port: '22'

```

Panorama

```

- name: Add the necessary service object to Panorama first
  panos_object:
    provider: '{{ provider }}'
    name: 'service-tcp-221'
    protocol: 'tcp'
    destination_port: '221'
    description: 'SSH on port 221'
    commit: false
    device_group: 'shared_services_11022'

- name: Create dynamic NAT rule on Panorama
  panos_nat_rule:
    provider: '{{ provider }}'
    rule_name: 'Web SSH inbound'
    source_zone: ['external']
    destination_zone: 'external'
    source_ip: ['any']
    destination_ip: ['10.0.0.100']
    service: 'service-tcp-221'
    snat_type: 'dynamic-ip-and-port'
    snat_interface: ['ethernet1/2']
    dnat_address: '10.0.1.101'
    dnat_port: '22'
    device_group: 'shared_services_11022'

```

3.1.3 Change firewall admin password using SSH

Change admin password of PAN-OS device using SSH with SSH key. This is used in particular when NGFW is deployed in the cloud (such as AWS).

```

- name: Change user password using ssh protocol
  panos_admpwd:
    ip_address: '{{ ip_address }}'
    password: '{{ password }}'
    newpassword: '{{ new_password }}'
    key_filename: '{{ key_filename }}'

```

3.1.4 Generates self-signed certificate

This module generates a self-signed certificate that can be used by GlobalProtect client, SSL connector, or otherwise. Root certificate must be preset on the system first. This module depends on paramiko for ssh.

```
- name: generate self signed certificate
  panos_cert_gen_ssh:
    ip_address: "{{ ip_address }}"
    username: "{{ username }}"
    password: "{{ password }}"
    cert_cn: "{{ cn }}"
    cert_friendly_name: "{{ friendly_name }}"
    signed_by: "{{ signed_by }}"
```

3.1.5 Check if FW is ready

Check if PAN-OS device is ready for being configured (no pending jobs). The check could be done once or multiple times until the device is ready.

```
- name: Wait for FW reboot
  panos_check:
    provider: '{{ provider }}'
    register: result
    until: not result|failed
    retries: 50
    delay: 5
```

3.1.6 Import configuration

Import file into PAN-OS device.

```
- name: import configuration file into PAN-OS
  panos_import:
    ip_address: "{{ ip_address }}"
    username: "{{ username }}"
    password: "{{ password }}"
    file: "{{ config_file }}"
    category: "configuration"
```

3.1.7 DHCP on DataPort

Configure data-port (DP) network interface for DHCP. By default DP interfaces are static.

```
- name: enable DHCP client on ethernet1/1 in zone external
  panos_interface:
    provider: '{{ provider }}'
    if_name: "ethernet1/1"
    zone_name: "external"
    create_default_route: "yes"
    commit: False
```

3.1.8 Load configuration

This is example playbook that imports and loads firewall configuration from a configuration file

```

- name: import config
  hosts: my-firewall
  connection: local
  gather_facts: False

  vars:
    cfg_file: candidate-template-empty.xml

  roles:
    - role: PaloAltoNetworks.paloaltonetworks

  tasks:
    - name: Grab the credentials from ansible-vault
      include_vars: 'firewall-secrets.yml'
      no_log: 'yes'

    - name: wait for SSH (timeout 10min)
      wait_for: port=22 host='{{ provider.ip_address }}' search_regex=SSH_
      ↪timeout=600

    - name: checking if device ready
      panos_check:
        provider: '{{ provider }}'
        register: result
        until: not result|failed
        retries: 10
        delay: 10

    - name: import configuration
      panos_import:
        ip_address: '{{ provider.ip_address }}'
        username: '{{ provider.username }}'
        password: '{{ provider.password }}'
        file: '{{cfg_file}}'
        category: 'configuration'
        register: result

    - name: load configuration
      panos_loadcfg:
        ip_address: '{{ provider.ip_address }}'
        username: '{{ provider.username }}'
        password: '{{ provider.password }}'
        file: '{{result.filename}}'
        commit: False

    - name: set admin password
      panos_administrator:
        provider: '{{ provider }}'
        admin_username: 'admin'
        admin_password: '{{ provider.password }}'
        superuser: True
        commit: False

    - name: commit (blocks until finished)

```

(continues on next page)

(continued from previous page)

```
panos_commit:
  provider: '{{ provider }}'
```

3.2 Module Reference

3.2.1 panos_address_group – Create address group objects on PAN-OS devices

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create address group objects on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```

- name: Create object group 'Prod'
  panos_address_group:
    provider: '{{ provider }}'
    name: 'Prod'
    static_value: ['Test-One', 'Test-Three']
    tag: ['Prod']

- name: Create object group 'SI'
  panos_address_group:
    provider: '{{ provider }}'
    name: 'SI'
    dynamic_value: "'SI_Instances'"
    tag: ['SI']

- name: Delete object group 'SI'
  panos_address_group:
    provider: '{{ provider }}'
    name: 'SI'
    state: 'absent'

```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)
- Garfield Lee Freeman (@shinmog)

3.2.2 panos_address_object – Create address objects on PAN-OS devices

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create address objects on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Create object 'Test-One'
  panos_address_object:
    provider: '{{ provider }}'
    name: 'Test-One'
    value: '1.1.1.1'
    description: 'Description One'
    tag: ['Prod']

- name: Create object 'Test-Two'
  panos_address_object:
    provider: '{{ provider }}'
    name: 'Test-Two'
    address_type: 'ip-range'
    value: '1.1.1.1-2.2.2.2'
    description: 'Description Two'
    tag: ['SI']

- name: Create object 'Test-Three'
  panos_address_object:
    provider: '{{ provider }}'
    name: 'Test-Three'
    address_type: 'fqdn'
    value: 'foo.bar.baz'
    description: 'Description Three'

- name: Delete object 'Test-Two'
```

(continues on next page)

(continued from previous page)

```
panos_address_object:
  provider: '{{ provider }}'
  name: 'Test-Two'
  state: 'absent'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)
- Garfield Lee Freeman (@shinmog)

3.2.3 panos_admin – Add or modify PAN-OS user accounts password

New in version 2.3.

- *DEPRECATED*
- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

DEPRECATED

Removed in Ansible version: 2.12

Why This module is a subset of *panos_administrator*'s functionality.

Alternative Use *panos_administrator* instead.

Synopsis

- PanOS module that allows changes to the user account passwords by doing API calls to the Firewall using pan-api as the protocol.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>

Parameters

Notes

Note:

- Checkmode is not supported.
-

Examples

```
# Set the password of user admin to "badpassword"
# Doesn't commit the candidate config
- name: set admin password
  panos_admin:
    ip_address: "192.168.1.1"
    password: "admin"
    admin_username: admin
    admin_password: "badpassword"
    commit: False
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module will be removed in version 2.12. *[deprecated]*
- For more information see *DEPRECATED*.

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer)

3.2.4 panos_administrator – Manage PAN-OS administrator user accounts

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Manages PAN-OS administrator user accounts.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - Because “request password-hash” does not always generate the same hash with the same password every time, it isn’t possible to tell if the admin’s password is correct or not. Specifying check mode or *state=present* with *admin_password* specified will always report *changed=True* in the return value.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
# Configure user "foo"
# Doesn't commit the candidate config
- name: configure foo administrator
  panos_administrator:
    provider: '{{ provider }}'
    admin_username: 'foo'
    admin_password: 'secret'
    superuser: true
    commit: false
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.5 panos_admpwd – change admin password of PAN-OS device using SSH with SSH key

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Change the admin password of PAN-OS via SSH using a SSH key for authentication.
- Useful for AWS instances where the first login should be done via SSH.

Requirements

The below requirements are needed on the host that executes this module.

- paramiko

Parameters

Examples

```
# Tries for 10 times to set the admin password of 192.168.1.1 to "badpassword"
# via SSH, authenticating using key /tmp/ssh.key
- name: set admin password
  panos_admpwd:
    ip_address: "192.168.1.1"
    username: "admin"
    key_filename: "/tmp/ssh.key"
    newpassword: "badpassword"
  register: result
  until: not result|failed
  retries: 10
  delay: 30
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer)

3.2.6 panos_aggregate_interface – configure aggregate network interfaces

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure aggregate network interfaces on PanOS

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPi <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPi <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create ael interface.
- name: create ael interface with IP in untrust zone
  panos_aggregate_interface:
    provider: '{{ provider }}'
    if_name: "ael"
    ip: '[ "192.168.0.1" ]'
    zone_name: 'untrust'
```


Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Heiko Burghardt (@odysseus107)

3.2.7 panos_api_key – retrieve api_key for username/password combination

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- This module will allow retrieval of the api_key for a given username/password

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Checkmode is NOT supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: retrieve api_key
  panos_op:
    provider: '{{ provider }}'
    register: auth

- name: show system info
  panos_op:
    ip_address: '{{ ip_address }}'
    api_key: '{{ auth.api_key }}'
    cmd: show system info
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.8 panos_bgp_aggregate – Configures a BGP Aggregation Prefix Policy

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create BGP Aggregation Rule
  panos_bgp_aggregate:
    provider: '{{ provider }}'
    vr_name: 'default'
    name: 'aggr-rule-01'
    prefix: '10.0.0.0/24'
    enable: true
    summary: true

- name: Remove BGP Aggregation Rule
  panos_bgp_aggregate:
    provider: '{{ provider }}'
    vr_name: 'default'
    name: 'aggr-rule-01'
    state: 'absent'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.9 panos_bgp_auth – Configures a BGP Authentication Profile

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is not supported.
- Panorama is supported.
- Since the *secret* value is encrypted in PAN-OS, there is no way to verify if the secret is properly set or not. Invoking this module with *state=present* will always apply the config to PAN-OS.

- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
- If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.

Examples

```
- name: Create BGP Authentication Profile
panos_bgp_auth:
  provider: '{{ provider }}'
  vr_name: 'my virtual router'
  name: auth-profile-1
  secret: SuperSecretCode
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.10 panos_bgp_conditional_advertisement – Configures a BGP conditional advertisement

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.
- In the PAN-OS GUI, this resource cannot be created without also creating at least one non-exist filter and one advertise filter. The API behaves a little differently; you can create the conditional advertisement itself, but the API will start throwing errors if you try to update it and there is not at least one non-exist filter and one advertise filter.
- In order for a conditional advertisement to be valid, you must specify at least one non-exist and one advertise filter.
- When modifying a BGP conditional advertisement, any filters attached are left as-is, unless *advertise_filter* or *non_exist_filter* are specified.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create BGP Conditional Advertisement Rule
panos_bgp_conditional_advertisement:
  provider: '{{ provider }}'
  name: 'cond-rule-01'
  enable: true
  non_exist_filter: '{{ non_exist.panos_obj }}'
  advertise_filter: '{{ advertise.panos_obj }}'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.11 panos_bgp_dampening – Configures a BGP Dampening Profile

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
- Panorama is supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
- If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.

Examples

```
- name: Create BGP Dampening Profile
  panos_bgp_dampening:
    name: damp-profile-1
    enable: true
    commit: true
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.12 panos_bgp – Configures Border Gateway Protocol (BGP)

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Configure and enable BGP
  panos_bgp:
    provider: '{{ provider }}'
    router_id: '1.1.1.1'
    local_as: '64512'
    commit: true
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.13 panos_bgp_peer_group – Configures a BGP Peer Group

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create BGP Peer Group
panos_bgp_peer_group:
  provider: '{{ provider }}'
  name: 'peer-group-1'
  enable: true
  aggregated_confed_as_path: true
  soft_reset_with_stored_info: false
  commit: true
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.14 panos_bgp_peer – Configures a BGP Peer

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
- Panorama is supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
- If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.

Examples

```
- name: Create BGP Peer
panos_bgp_peer:
  provider: '{{ provider }}'
  peer_group: 'peer-group-1'
  name: 'peer-1'
  enable: true
  local_interface: 'ethernet1/1'
  local_interface_ip: '192.168.1.1'
  peer_address_ip: '10.1.1.1'
  peer_as: '64512'
  commit: true
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.15 panos_bgp_policy_filter – Configures a BGP Policy Import/Export Rule

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.16 panos_bgp_policy_rule – Configures a BGP Policy Import/Export Rule

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
# Add a BGP Policy
- name: Create Policy Import Rule
  panos_bgp_policy_rule:
    provider: '{{ provider }}'
    vr_name: 'default'
    name: 'import-rule-001'
    type: 'import'
    enable: true
    action: 'allow'
    address_prefix:
      - '10.1.1.0/24'
      - name: '10.1.2.0/24'
        exact: false
      - name: '10.1.3.0/24'
        exact: true
    action_dampening: 'dampening-profile'

- name: Create Policy Export Rule
  panos_bgp_policy_rule:
    provider: '{{ provider }}'
    vr_name: 'default'
    name: 'export-rule-001'
    type: 'export'
    enable: true
    action: 'allow'

- name: Remove Export Rule
  panos_bgp_policy_rule:
    provider: '{{ provider }}'
    state: 'absent'
    vr_name: 'default'
    name: 'export-rule-001'
    type: 'export'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.17 panos_bgp_redistribute – Configures a BGP Redistribution Rule

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use BGP to publish and consume routes from disparate networks.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: BGP use Redistribution Policy 1
  panos_bgp_redistribute:
    provider: '{{ provider }}'
    name: '10.2.3.0/24'
    enable: true
    commit: true
    address_family_identifier: ipv4
```

(continues on next page)

(continued from previous page)

```
set_origin: incomplete
vr_name: default
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.18 panos_cert_gen_ssh – generates a self-signed certificate using SSH protocol with SSH key

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- This module generates a self-signed certificate that can be used by GlobalProtect client, SSL connector, or otherwise. Root certificate must be preset on the system first. This module depends on paramiko for ssh.

Requirements

The below requirements are needed on the host that executes this module.

- paramiko

Parameters

Notes

Note:

- Checkmode is not supported.
-

Examples

```
# Generates a new self-signed certificate using ssh
- name: generate self signed certificate
  panos_cert_gen_ssh:
    ip_address: "192.168.1.1"
    username: "admin"
    password: "paloalto"
    cert_cn: "1.1.1.1"
    cert_friendly_name: "test123"
    signed_by: "root-ca"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer)

3.2.19 panos_check – check if PAN-OS device is ready for configuration

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Check if PAN-OS device is ready for being configured (no pending jobs).
- The check could be done once or multiple times until the device is ready.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Panorama is supported.
 - Checkmode is not supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Single check.
- name: check if ready
  panos_check:
    provider: '{{ provider }}'
    timeout: 0

# Wait 2 minutes, then check every 5 seconds for 10 minutes.
- name: wait for reboot
  panos_check:
    provider: '{{ provider }}'
    initial_delay: 120
    interval: 5
    timeout: 600
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold)
- Ivan Bojer (@ivanbojer)
- Garfield Lee Freeman (@shinmog)

3.2.20 panos_commit – Commit a PAN-OS device’s candidate configuration

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Module that will commit the candidate configuration of a PAN-OS device.
- The new configuration will become active immediately.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: commit candidate config on firewall
  panos_commit:
    provider: '{{ provider }}'

- name: commit candidate config on Panorama
  panos_commit:
    provider: '{{ provider }}'
    device_group: 'Cloud-Edge'

- name: commit changes by specified admins to firewall
  panos_commit:
    provider: '{{ provider }}'
    admins: ['admin1', 'admin2']
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)
- Garfield Lee Freeman (@shinmog)

3.2.21 panos_dag – create a dynamic address group

New in version 2.3.

- *DEPRECATED*
- *Synopsis*
- *Requirements*
- *Parameters*
- *Examples*

- *Status*

DEPRECATED

Removed in Ansible version: 2.12

Why This module's functionality is a subset of *panos_address_group*.

Alternative Use *panos_address_group* instead.

Synopsis

- Create a dynamic address group object in the firewall used for policy rules

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Examples

```
- name: dag
  panos_dag:
    ip_address: "192.168.1.1"
    password: "admin"
    dag_name: "dag-1"
    dag_match_filter: "'aws-tag.aws:cloudformation:logical-id.ServerInstance' and
↪ 'instanceState.running'"
    description: 'Add / create dynamic address group to allow access to SaaS_
↪ Applications'
    operation: 'add'
```

Status

- This module will be removed in version 2.12. *[deprecated]*
- For more information see *DEPRECATED*.

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer), Vinay Venkataraghavan (@vinayvenkat)

3.2.22 panos_dag_tags – Create tags for DAG’s on PAN-OS devices

New in version 2.5.

- *DEPRECATED*
- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

DEPRECATED

Removed in Ansible version: 2.9

Why Using new modern API calls in the panos_registered_ip

Alternative Use *panos_registered_ip* instead.

Synopsis

- Create the ip address to tag associations. Tags will in turn be used to create DAG’s

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is not supported.
 - Panorama is not supported.
 - use panos_registered_ip from now on
-

Examples

```
- name: Create the tags to map IP addresses
panos_dag_tags:
  ip_address: "{{ ip_address }}"
  password: "{{ password }}"
  ip_to_register: "{{ ip_to_register }}"
  tag_names: "{{ tag_names }}"
  description: "Tags to allow certain IP's to access various SaaS Applications"
  operation: 'add'
tags: "adddagip"

- name: List the IP address to tag mapping
panos_dag_tags:
  ip_address: "{{ ip_address }}"
  password: "{{ password }}"
  tag_names: "{{ tag_names }}"
  description: "List the IP address to tag mapping"
  operation: 'list'
tags: "listdagip"

- name: Unregister an IP address from a tag mapping
panos_dag_tags:
  ip_address: "{{ ip_address }}"
  password: "{{ password }}"
  ip_to_register: "{{ ip_to_register }}"
  tag_names: "{{ tag_names }}"
  description: "Unregister IP address from tag mappings"
  operation: 'delete'
tags: "deletedagip"
```

Status

- This module will be removed in version 2.9. *[deprecated]*
- For more information see *DEPRECATED*.

Authors

- Vinay Venkataraghavan (@vinayvenkat)

3.2.23 panos_email_profile – Manage email server profiles

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*

- *Status*

Synopsis

- Manages email server profiles.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
- Check mode is supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
# Create a profile
- name: Create email profile
  panos_email_profile:
    provider: '{{ provider }}'
    name: 'my-profile'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.24 panos_email_server – Manage email servers in an email profile

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages email servers in an email server profile.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a profile
- name: Create email server in an email profile
  panos_email_server:
    provider: '{{ provider }}'
    email_profile: 'my-profile'
    name: 'my-email-server'
    from_email: 'alerts@example.com'
    to_email: 'notify@example.com'
    email_gateway: 'smtp.example.com'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.25 panos_facts – Collects facts from Palo Alto Networks device

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Collects fact information from Palo Alto Networks firewall running PanOS.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python

Parameters

Notes

Note:

- Tested on PanOS 8.0.5
 - Checkmode is not supported.
 - Panorama is not supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Gather facts
- name: Get facts
  panos_facts:
    provider: '{{ provider }}'
    gather_subset: ['config']
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Tomi Raittinen (@traittinen)
- Garfield Lee Freeman (@shinmog)

3.2.26 panos_gre_tunnel – Create GRE tunnels on PAN-OS devices

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create GRE tunnel objects on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Minimum PAN-OS version: 9.0
 - Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create GRE tunnel
  panos_gre_tunnel:
    provider: '{{ provider }}'
    name: 'myGreTunnel'
    interface: 'ethernet1/5'
    local_address_value: '10.1.1.1/24'
    peer_address: '192.168.1.1'
    tunnel_interface: 'tunnel.7'
    ttl: 42
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.27 panos_ha – Configures High Availability on PAN-OS

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configures High Availability on PAN-OS in A/S and A/A modes including all HA interface configuration. Assumes physical interfaces are of type HA already using panos_interface.

This module has the following limitations due to no support in pandevice - * No peer_backup_ip, this prevents full configuration of ha1_backup links * Speed and Duplex of ports was intentionally skipped

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>
- currently requires specific pandevice release 0.13

Parameters

Notes

Note:

- Checkmode is supported.
- Panorama is supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
- If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.

Examples

```
- name: set ports to HA mode
  panos_interface:
    provider: '{{ provider }}'
    if_name: "{{ item }}"
    mode: "ha"
    enable_dhcp: false
    commit: false
  with_items:
    - ethernet1/1
    - ethernet1/2
    - ethernet1/3
    - ethernet1/4
    - ethernet1/5

- name: Configure Active/Standby HA
  panos_ha:
    provider: '{{ provider }}'
    state: present
    ha_peer_ip: "192.168.50.1"
    ha1_ip_address: "192.168.50.2"
    ha1_netmask: "255.255.255.252"
    ha1_port: "ethernet1/1"
    ha2_port: "ethernet1/3"
    commit: "true"

- name: Configure Active/Active HA
  panos_ha:
    provider: "{{ provider }}"
```

(continues on next page)

(continued from previous page)

```
state: present
ha_mode: "active-active"
ha_device_id: 0
ha_session_owner_selection: "first-packet"
ha_session_setup: "first-packet"
ha_peer_ip: "192.168.50.1"
ha_peer_ip_backup: "192.168.50.5"
ha1_port: "ethernet1/1"
ha1_ip_address: "192.168.50.2"
ha1_netmask: "255.255.255.252"
ha1b_port: "ethernet1/2"
ha1b_ip_address: "192.168.50.6"
ha1b_netmask: "255.255.255.252"
ha2_port: "ethernet1/3"
ha2b_port: "ethernet1/4"
ha3_port: "ethernet1/5"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Patrick Avery

3.2.28 panos_http_profile_header – Manage HTTP headers for a HTTP profile

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages HTTP headers for a HTTP profile.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1
- PAN-OS >= 8.0

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Add a header to the config log type
  panos_http_profile_header:
    provider: '{{ provider }}'
    http_profile: 'my-profile'
    log_type: 'user id'
    header: 'Content-Type'
    value: 'application/json'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.29 panos_http_profile – Manage http server profiles

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages http server profiles.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1
- PAN-OS >= 8.0

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a profile
- name: Create http profile
  panos_http_profile:
    provider: '{{ provider }}'
    name: 'my-profile'
    tag_registration: true
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.30 panos_http_profile_param – Manage HTTP params for a HTTP profile

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages HTTP params for a HTTP profile.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1
- PAN-OS >= 8.0

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Add a param to the config log type
  panos_http_profile_param:
    provider: '{{ provider }}'
    http_profile: 'my-profile'
    log_type: 'user id'
    param: 'serial'
    value: '$serial'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.31 panos_http_server – Manage HTTP servers in a HTTP server profile

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages HTTP servers in a HTTP server profile.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1
- PAN-OS >= 8.0

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Create http server
  panos_http_server:
    provider: '{{ provider }}'
    http_profile: 'my-profile'
    name: 'my-http-server'
    address: '192.168.1.5'
    http_method: 'GET'
    http_username: 'jack'
    http_password: 'burton'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.32 panos_ike_crypto_profile – Configures IKE Crypto profile on the firewall with subset of settings

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use the IKE Crypto Profiles page to specify protocols and algorithms for identification, authentication, and encryption (IKEv1 or IKEv2, Phase 1).

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Add IKE crypto config to the firewall
  panos_ike_crypto_profile:
    provider: '{{ provider }}'
    state: 'present'
    name: 'vpn-0cc61dd8c06f95cfd-0'
    dh_group: ['group2']
    authentication: ['sha1']
    encryption: ['aes-128-cbc']
    lifetime_seconds: '28800'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Ivan Bojer (@ivanbojer)

3.2.33 panos_ike_gateway – Configures IKE gateway on the firewall with subset of settings

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use this to manage or define a gateway, including the configuration information necessary to perform Internet Key Exchange (IKE) protocol negotiation with a peer gateway. This is the Phase 1 portion of the IKE/IPSec VPN setup.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Add IKE gateway config to the firewall
  panos_ike_gateway:
    provider: '{{ provider }}'
    state: 'present'
    name: 'IKEGW-Ansible'
    version: 'ikev2'
    interface: 'ethernet1/1'
    enable_passive_mode: True
    enable_liveness_check: True
    liveness_check_interval: '5'
    peer_ip_value: '1.2.3.4'
    pre_shared_key: 'CHANGEME'
    ikev2_crypto_profile: 'IKE-Ansible'
    commit: False
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Ivan Bojer (@ivanbojer)

3.2.34 panos_import – import file on PAN-OS devices

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Examples*
- *Status*

Synopsis

- Import file on PAN-OS device

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- requests
- requests_toolbelt

Parameters

Examples

```
# import software image PanOS_vm-6.1.1 on 192.168.1.1
- name: import software image into PAN-OS
  panos_import:
    ip_address: 192.168.1.1
    username: admin
    password: admin
    file: /tmp/PanOS_vm-6.1.1
    category: software
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer)

3.2.35 panos_interface – configure data-port network interfaces

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure data-port (DP) network interface. By default DP interfaces are static.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>
- pandevice >= 0.8.0

Parameters

Notes

Note:

- Checkmode is supported.
 - If the PAN-OS device is a firewall and *vsys* is not specified, then the *vsys* will default to *vsys=vsys1*.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create ethernet1/1 as DHCP.
- name: enable DHCP client on ethernet1/1 in zone public
  panos_interface:
    provider: '{{ provider }}'
    if_name: "ethernet1/1"
    zone_name: "public"
    create_default_route: "yes"

# Update ethernet1/2 with a static IP address in zone dmz.
- name: ethernet1/2 as static in zone dmz
  panos_interface:
    provider: '{{ provider }}'
    if_name: "ethernet1/2"
    mode: "layer3"
    ip: ["10.1.1.1/24"]
    enable_dhcp: false
    zone_name: "dmz"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold)
- Ivan Bojer (@ivanbojer)
- Garfield Lee Freeman (@shinmog)

3.2.36 panos_ipsec_ipv4_proxyid – Configures IPv4 Proxy Id on an IPSec Tunnel

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Add IPSec IPv4 Proxy ID
  panos_ipsec_ipv4_proxyid:
    provider: '{{ provider }}'
    name: 'IPSec-ProxyId'
    tunnel_name: 'Default_Tunnel'
    local: '192.168.2.0/24'
    remote: '192.168.1.0/24'
    commit: False
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Heiko Burghardt (@odysseus107)

3.2.37 panos_ipsec_profile – Configures IPSec Crypto profile on the firewall with subset of settings

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- IPSec Crypto profiles specify protocols and algorithms for authentication and encryption in VPN tunnels based on IPSec SA negotiation (Phase 2).

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Add IPSec crypto config to the firewall
  panos_ipsec_profile:
    provider: '{{ provider }}'
    state: 'present'
    name: 'ipsec-vpn-0cc61dd8c06f95cfd-0'
    esp_authentication: ['sha1']
    esp_encryption: ['aes-128-cbc']
    lifetime_seconds: '3600'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Ivan Bojer (@ivanbojer)

3.2.38 panos_ipsec_tunnel – Configures IPSec Tunnels on the firewall with subset of settings

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Use IPSec Tunnels to establish and manage IPSec VPN tunnels between firewalls. This is the Phase 2 portion of the
- IKE/IPSec VPN setup.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Add IPSec tunnel to IKE gateway profile
panos_ipsec_tunnel:
  provider: '{{ provider }}'
  name: 'IPSecTunnel-Ansible'
  tunnel_interface: 'tunnel.2'
  ak_ike_gateway: 'IKEGW-Ansible'
  ak_ipsec_crypto_profile: 'IPSec-Ansible'
  state: 'present'
  commit: False
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Ivan Bojer (@ivanbojer)

3.2.39 panos_l2_subinterface – configure layer2 subinterface

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure a layer2 subinterface.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.8.0

Parameters

Notes

Note:

- Panorama is supported.
 - Checkmode is supported.
 - If the PAN-OS device is a firewall and *vsys* is not specified, then the vsys will default to *vsys=vsys1*.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create ethernet1/1.5
- name: ethernet1/1.5 in zone sales
  panos_l2_subinterface:
    provider: '{{ provider }}'
    name: "ethernet1/1.5"
    tag: 5
    zone_name: "sales"
    vlan_name: "myVlan"
```


Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.40 panos_l3_subinterface – configure layer3 subinterface

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure a layer3 subinterface.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.8.0

Parameters

Notes

Note:

- Panorama is supported.
 - Checkmode is supported.
 - If the PAN-OS device is a firewall and *vsys* is not specified, then the *vsys* will default to *vsys=vsys1*.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create ethernet1/1.5 as DHCP.
- name: enable DHCP client on ethernet1/1.5 in zone public
  panos_l3_subinterface:
    provider: '{{ provider }}'
    name: "ethernet1/1.5"
    tag: 1
    create_default_route: True
    zone_name: "public"
    create_default_route: "yes"

# Update ethernet1/2.7 with a static IP address in zone dmz.
- name: ethernet1/2.7 as static in zone dmz
  panos_l3_subinterface:
    provider: '{{ provider }}'
    name: "ethernet1/2.7"
    tag: 7
    enable_dhcp: false
    ip: ["10.1.1.1/24"]
    zone_name: "dmz"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.41 panos_lic – apply authcode to a device/instance

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Apply an authcode to a device.
- The authcode should have been previously registered on the Palo Alto Networks support portal.
- The device should have Internet access.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Panorama is supported
 - Checkmode is not supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Activate my authcode
  panos_lic:
    provider: '{{ provider }}'
    auth_code: "IBADCODE"
  register: result

- debug:
  msg: 'Serial number is {{ result.serialnumber }}'
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer)

3.2.42 panos_loadcfg – load configuration on PAN-OS device

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Examples*
- *Status*

Synopsis

- Load configuration on PAN-OS device

Requirements

The below requirements are needed on the host that executes this module.

- pan-python

Parameters

Examples

```
# Import and load config file from URL
- name: import configuration
  panos_import:
    ip_address: "192.168.1.1"
    password: "admin"
    url: "{{ConfigURL}}"
    category: "configuration"
  register: result
- name: load configuration
  panos_loadcfg:
    ip_address: "192.168.1.1"
    password: "admin"
    file: "{{result.filename}}"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer)

3.2.43 panos_log_forwarding_profile_match_list_action – Manage log forwarding profile match list actions

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages log forwarding profile match list actions.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a log forwarding server match list action
- name: Create the action
  panos_log_forwarding_profile_match_list_action:
    provider: '{{ provider }}'
    log_forwarding_profile: 'my-profile'
    log_forwarding_profile_match_list: 'ml-1'
    name: 'my-action'
    action: 'add-tag'
    target: 'source-address'
    registration: 'localhost'
    tags: ['foo', 'bar']
    timeout: 2
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.44 panos_log_forwarding_profile_match_list – Manage log forwarding profile match lists

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages log forwarding profile match lists.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a server match list
- name: Create log forwarding profile match list
  panos_log_forwarding_profile_match_list:
    provider: '{{ provider }}'
    log_forwarding_profile: 'my-profile'
    name: 'ml-1'
    description: 'created by Ansible'
    log_type: 'threat'
    filter: '(action eq allow) and (zone eq DMZ)'
    syslog_profiles: ['syslog-profl']
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.45 panos_log_forwarding_profile – Manage log forwarding profiles

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages log forwarding profiles.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a profile
- name: Create log forwarding profile
  panos_log_forwarding_profile:
    provider: '{{ provider }}'
    name: 'my-profile'
    enhanced_logging: true
```


Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.46 panos_loopback_interface – configure network loopback interfaces

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure loopback interfaces on PanOS

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPi <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPi <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Delete loopback.1
- name: delete loopback.1
  panos_loopback_interface:
    provider: '{{ provider }}'
    if_name: "loopback.1"
    state: 'absent'

# Update/create loopback comment.
- name: update loopback.1 comment
  panos_loopback_interface:
    provider: '{{ provider }}'
    if_name: "loopback.1"
    ip: ["10.1.1.1/32"]
    comment: "Loopback interface"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Geraint Jones (@nexus_moneky_nz)
- Garfield Lee Freeman (@shinmog)

3.2.47 panos_management_profile – Manage interface management profiles

New in version 2.6.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- This module will allow you to manage interface management profiles on PAN-OS.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: ensure mngt profile foo exists and allows ping and ssh and commit
  panos_management_profile:
    provider: '{{ provider }}'
    name: 'foo'
    ping: true
    ssh: true

- name: make sure mngt profile bar does not exist without doing a commit
  panos_management_profile:
    provider: '{{ provider }}'
    name: 'bar'
    state: 'absent'
    commit: false
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.48 panos_match_rule – Test for match against a security rule on PAN-OS devices or Panorama management console

New in version 2.5.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Security policies allow you to enforce rules and take action, and can be as general or specific as needed.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>
- xmltodict

Parameters

Notes

Note:

- Checkmode is not supported.
- Panorama NOT is supported. However, specifying Panorama *provider* info with a target serial number is.

- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: check security rules for Google DNS
  panos_match_rule:
    provider: '{{ provider }}'
    source_ip: '10.0.0.0'
    destination_ip: '8.8.8.8'
    application: 'dns'
    destination_port: '53'
    protocol: '17'
  register: result
- debug: msg='{{ result.rule }}'

- name: check security rules inbound SSH with user match
  panos_match_rule:
    provider: '{{ provider }}'
    source_ip: '0.0.0.0'
    source_user: 'mydomain\jsmith'
    destination_ip: '192.168.100.115'
    destination_port: '22'
    protocol: '6'
  register: result
- debug: msg='{{ result.rule }}'

- name: check NAT rules for source NAT
  panos_match_rule:
    provider: '{{ provider }}'
    rule_type: 'nat'
    source_zone: 'Prod-DMZ'
    source_ip: '10.10.118.50'
    to_interface: 'ethernet1/2'
    destination_zone: 'Internet'
    destination_ip: '0.0.0.0'
    protocol: '6'
  register: result
- debug: msg='{{ result.rule }}'

- name: check NAT rules for inbound web
  panos_match_rule:
    provider: '{{ provider }}'
    rule_type: 'nat'
    source_zone: 'Internet'
    source_ip: '0.0.0.0'
    to_interface: 'ethernet1/1'
    destination_zone: 'Prod DMZ'
    destination_ip: '192.168.118.50'
    destination_port: '80'
    protocol: '6'
  register: result
- debug: msg='{{ result.rule }}'

- name: check security rules for outbound POP3 in vsys4
```

(continues on next page)

(continued from previous page)

```
panos_match_rule:
  provider: '{{ provider }}'
  vsys_id: 'vsys4'
  source_ip: '10.0.0.0'
  destination_ip: '4.3.2.1'
  application: 'pop3'
  destination_port: '110'
  protocol: '6'
  register: result
- debug: msg='{{ result.rule }}'
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Robert Hagen (@rh556)

3.2.49 panos_mgtconfig – Module used to configure some of the device management

New in version 2.4.

```
• Synopsis
• Requirements
• Parameters
• Notes
• Examples
• Status
```

Synopsis

- Configure management settings of device. Not all configuration options are configurable at this time.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: set dns and panorama
  panos_mgtconfig:
    provider: '{{ provider }}'
    dns_server_primary: "1.1.1.1"
    dns_server_secondary: "1.1.1.2"
    panorama_primary: "1.1.1.3"
    panorama_secondary: "1.1.1.4"
    ntp_server_primary: "1.1.1.5"
    ntp_server_secondary: "1.1.1.6"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer), Patrik Malinen (@pmalinen), Francesco Vigo (@fvigo)

3.2.50 panos_nat_rule_facts – Get information about a NAT rule

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Get information about one or more NAT rules.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Checkmode is not supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```

- name: Get a list of all NAT rules
  panos_nat_rule_facts:
    provider: '{{ provider }}'
    listing: true
    register: res1

- debug:
  msg: '{{ res1.listing }}'

- name: Get the NAT rule foo
  panos_nat_rule_facts:
    provider: '{{ provider }}'
    rule_name: 'foo'
    register: res2

- debug:
  msg: '{{ res2.object }}'

```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.51 panos_nat_rule – create a policy NAT rule

New in version 2.4.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create a policy nat rule. Keep in mind that we can either end up configuring source NAT, destination NAT, or both.
- Instead of splitting it into two we will make a fair attempt to determine which one the user wants.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a source and destination nat rule
- name: Create NAT SSH rule for 10.0.1.101
  panos_nat_rule:
    provider: '{{ provider }}'
    rule_name: "Web SSH"
    source_zone: ["external"]
    destination_zone: "external"
    source_ip: ["any"]
    destination_ip: ["10.0.0.100"]
    service: "service-tcp-221"
    snat_type: "dynamic-ip-and-port"
    snat_interface: "ethernet1/2"
    dnat_address: "10.0.1.101"
    dnat_port: "22"

- name: disable a specific security rule
  panos_nat_rule:
    provider: '{{ provider }}'
    rule_name: 'Prod-Legacy 1'
    state: 'disable'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold)
- Ivan Bojer (@ivanbojer)
- Robert Hagen (@rnh556)
- Michael Richardson (@mrichardson03)
- Garfield Lee Freeman (@shinmog)

3.2.52 panos_object_facts – Retrieve facts about objects on PAN-OS devices

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Retrieves tag information objects on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is not supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Retrieve address group object 'Prod'
  panos_object_facts:
    provider: '{{ provider }}'
    name: 'Prod'
    object_type: 'address-group'
  register: result

- name: Retrieve service group object 'Prod-Services'
  panos_object_facts:
    provider: '{{ provider }}'
    name: 'Prod-Services'
    object_type: 'service-group'
  register: result

- name: Find all address objects with "Prod" in the name
  panos_object_facts:
    provider: '{{ provider }}'
    name_regex: '.*Prod.*'
    object_type: 'address'
  register: result

- name: Find all static address objects that use addy1
  panos_object_facts:
    provider: '{{ provider }}'
    object_type: 'address-group'
    field: 'static_value'
    field_search_type: 'exact'
    field_search_value: 'addy1'
  register: result
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Michael Richardson (@mrichardson03)
- Garfield Lee Freeman (@shinmog)

3.2.53 panos_object – create/read/update/delete object in PAN-OS or Panorama

New in version 2.4.

- *DEPRECATED*
- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

DEPRECATED

Removed in Ansible version: 2.9

Why Updated to idempotent modules

Alternative Use `panos_address_object`, `panos_address_group`, `panos_service_object`, `panos_service_group`, or `panos_tag_object` as appropriate.

Synopsis

- Policy objects form the match criteria for policy rules and many other functions in PAN-OS. These may include
- address object, address groups, service objects, service groups, and tag.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is not supported.
 - Panorama is supported.
-

Examples

```
- name: search for shared address object
panos_object:
  ip_address: '{{ ip_address }}'
  username: '{{ username }}'
  password: '{{ password }}'
  operation: 'find'
  address: 'DevNet'

- name: create an address group in devicegroup using API key
panos_object:
  ip_address: '{{ ip_address }}'
  api_key: '{{ api_key }}'
  operation: 'add'
  addressgroup: 'Prod_DB_Svrs'
  static_value: ['prod-db1', 'prod-db2', 'prod-db3']
  description: 'Production DMZ database servers'
  tag_name: 'DMZ'
  devicegroup: 'DMZ Firewalls'

- name: create a global service for TCP 3306
panos_object:
  ip_address: '{{ ip_address }}'
  api_key: '{{ api_key }}'
  operation: 'add'
  serviceobject: 'mysql-3306'
  destination_port: '3306'
  protocol: 'tcp'
  description: 'MySQL on tcp/3306'

- name: create a global tag
panos_object:
  ip_address: '{{ ip_address }}'
  username: '{{ username }}'
  password: '{{ password }}'
```

(continues on next page)

(continued from previous page)

```

operation: 'add'
tag_name: 'ProjectX'
color: 'yellow'
description: 'Associated with Project X'
- name: delete an address object from a devicegroup using API key
  panos_object:
    ip_address: '{{ ip_address }}'
    api_key: '{{ api_key }}'
    operation: 'delete'
    addressobject: 'Win2K test'

```

Status

- This module will be removed in version 2.9. *[deprecated]*
- For more information see *DEPRECATED*.

Authors

- Bob Hagen (@rnh556)

3.2.54 panos_op – execute arbitrary OP commands on PANW devices (e.g. show interface all)

New in version 2.5.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- This module will allow user to pass and execute any supported OP command on the PANW device.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is NOT supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: show list of all interfaces
  panos_op:
    provider: '{{ provider }}'
    cmd: 'show interfaces all'

- name: show system info
  panos_op:
    provider: '{{ provider }}'
    cmd: 'show system info'

- name: show system info as XML command
  panos_op:
    provider: '{{ provider }}'
    cmd: '<show><system><info/></system></show>'
    cmd_is_xml: true
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Ivan Bojer (@ivanbojer)
- Garfield Lee Freeman (@shinmog)

3.2.55 panos_pbf_rule – Manage Policy Based Forwarding rules on PAN-OS

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manage Policy Based Forwarding rules on PAN-OS.

Requirements

The below requirements are needed on the host that executes this module.

- pandevice >= 0.13.0
- pan-python

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: add a pbf rule
panos_pbf_rule:
  provider: '{{ provider }}'
  name: 'my-pbf'
  description: 'Made by Ansible'
  from_value: ['myZone']
  action: 'discard'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.56 panos_pg – create a security profiles group

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create a security profile group

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Panorama is supported.
- Checkmode is supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: setup security profile group
  panos_pg:
    provider: '{{ provider }}'
    pg_name: "pg-default"
    virus: "default"
    spyware: "default"
    vulnerability: "default"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold), Ivan Bojer (@ivanbojer)

3.2.57 panos_query_rules – PANOS module that allows search for security rules in PANW NGFW devices

New in version 2.5.

- *DEPRECATED*
- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

DEPRECATED

Removed in Ansible version: 2.12

Why Querying rules is handled better by *panos_match_rule*.

Alternative Use *panos_match_rule*

Synopsis

- Security policies allow you to enforce rules and take action, and can be as general or specific as needed.
- The policy rules are compared against the incoming traffic in sequence, and because the first rule that matches
- the traffic is applied, the more specific rules must precede the more general ones.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>
- xmldict can be obtains from PyPI <https://pypi.python.org/pypi/xmldict>

Parameters

Notes

Note:

- Checkmode is not supported.
 - Panorama is supported.
-

Examples

```
- name: search for rules with tcp/3306
  panos_query_rules:
    ip_address: '{{ ip_address }}'
    username: '{{ username }}'
    password: '{{ password }}'
    source_zone: 'DevNet'
    destination_zone: 'DevVPC'
    destination_port: '3306'
    protocol: 'tcp'

- name: search devicegroup for inbound rules to dmz host
  panos_query_rules:
    ip_address: '{{ ip_address }}'
    api_key: '{{ api_key }}'
    destination_zone: 'DMZ'
    destination_ip: '10.100.42.18'
```

(continues on next page)

(continued from previous page)

```

    address: 'DeviceGroupA'
- name: search for rules containing a specified rule tag
  panos_query_rules:
    ip_address: '{{ ip_address }}'
    username: '{{ username }}'
    password: '{{ password }}'
    tag_name: 'ProjectX'

```

Status

- This module will be removed in version 2.12. *[deprecated]*
- For more information see *DEPRECATED*.

Authors

- Bob Hagen (@rnh556)

3.2.58 panos_redistribution – Configures a Redistribution Profile on a virtual router

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configures a Redistribution Profile on a virtual router

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create Redistribution Profile
  panos_redistribution:
    provider: '{{ provider }}'
    name: 'my-profile'
    priority: 42
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.59 panos_registered_ip_facts – Retrieve facts about registered IPs on PAN-OS devices

New in version 2.7.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*

- *Status*

Synopsis

- Retrieves tag information about registered IPs on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is not supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: Get facts for all registered IPs
  panos_registered_ip_facts:
    provider: '{{ provider }}'
    register: registered_ip_facts

- name: Get facts for specific tag
  panos_registered_ip_facts:
    provider: '{{ provider }}'
    tags: ['First_Tag']
    register: first_tag_registered_ip_facts

- name: Get facts for a specific IP address
  panos_registered_ip_facts:
    provider: '{{ provider }}'
    ips: ['192.168.1.1']
    register: ipaddress_registered_ip_facts
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Michael Richardson (@mrichardson03)

3.2.60 panos_registered_ip – Register IP addresses for use with dynamic address groups on PAN-OS devices

New in version 2.7.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Registers tags for IP addresses that can be used to build dynamic address groups.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Check mode is supported.
- Panorama is not supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: Add 'First_Tag' tag to 1.1.1.1
  panos_registered_ip:
    provider: '{{ provider }}'
    ips: ['1.1.1.1']
    tags: ['First_Tag']
    state: 'present'

- name: Add 'First_Tag' tag to 1.1.1.2
  panos_registered_ip:
    provider: '{{ provider }}'
    ips: ['1.1.1.2']
    tags: ['First_Tag']
    state: 'present'

- name: Add 'Second_Tag' tag to 1.1.1.1
  panos_registered_ip:
    provider: '{{ provider }}'
    ips: ['1.1.1.1']
    tags: ['Second_Tag']
    state: 'present'

- name: Remove 'Second_Tag' from 1.1.1.1
  panos_registered_ip:
    provider: '{{ provider }}'
    ips: ['1.1.1.1']
    tags: ['Second_Tag']
    state: 'absent'

- name: Remove 'First_Tag' from 1.1.1.2 (will unregister entirely)
  panos_registered_ip:
    provider: '{{ provider }}'
    ips: ['1.1.1.2']
    tags: ['First_Tag']
    state: 'absent'
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Michael Richardson (@mrichardson03)

3.2.61 panos_restart – Restart a device

New in version 2.3.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Restart a PAN-OS device.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is not supported.
- Panorama is supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: Restart PAN-OS
  panos_restart:
    provider: '{{ provider }}'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Luigi Mori (@jtschichold)
- Ivan Bojer (@ivanbojer)
- Garfield Lee Freeman (@shinmog)

3.2.62 panos_sag – Create a static address group

New in version 2.4.

- *DEPRECATED*
- *Synopsis*
- *Requirements*
- *Parameters*
- *Examples*
- *Status*

DEPRECATED

Removed in Ansible version: 2.12

Why This module's functionality is a subset of *panos_address_group*.

Alternative Use *panos_address_group* instead.

Synopsis

- Create a static address group object in the firewall used for policy rules.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>
- xmltodict can be obtained from PyPI <https://pypi.python.org/pypi/xmltodict>

Parameters

Examples

```
- name: sag
  panos_sag:
    ip_address: "192.168.1.1"
    password: "admin"
    sag_name: "sag-1"
    static_value: ['test-addresses', ]
    description: "A description for the static address group"
    tags: ["tags to be associated with the group", ]
```

Status

- This module will be removed in version 2.12. *[deprecated]*
- For more information see *DEPRECATED*.

Authors

- Vinay Venkataraghavan @vinayvenkat

3.2.63 panos_security_rule_facts – Get information about a security rule

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Get information about a single security rule or the names of all security rules.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Checkmode is not supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Get a list of all security rules
  panos_security_rule_facts:
    provider: '{{ provider }}'
    register: sec_rules

- debug:
  msg: '{{ sec_rules.rules }}'
```

(continues on next page)

(continued from previous page)

```
- name: Get the definition for rule 'HTTP Multimedia'
  panos_security_rule_facts:
    provider: '{{ provider }}'
    rule_name: 'HTTP Multimedia'
    register: rule1

- debug:
  msg: '{{ rule1.spec }}'
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.64 panos_security_rule – Create security rule policy on PAN-OS devices or Panorama management console

New in version 2.4.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Security policies allow you to enforce rules and take action, and can be as general or specific as needed.
- The policy rules are compared against the incoming traffic in sequence, and because the first rule that matches the traffic is applied, the more specific rules must precede the more general ones.

Requirements

The below requirements are needed on the host that executes this module.

- `pandevice` can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
- Panorama is supported.
- PAN-OS connectivity should be specified using `provider` or the classic PAN-OS connectivity params (`ip_address`, `username`, `password`, `api_key`, and `port`). If both are present, then the classic params are ignored.

Examples

```
- name: add SSH inbound rule to Panorama device group
  panos_security_rule:
    provider: '{{ provider }}'
    device_group: 'Cloud Edge'
    rule_name: 'SSH permit'
    description: 'SSH rule test'
    tag_name: ['production']
    source_zone: ['public']
    source_ip: ['any']
    destination_zone: ['private']
    destination_ip: ['1.1.1.1']
    application: ['ssh']
    action: 'allow'

- name: add a rule to allow HTTP multimedia only to CDNs
  panos_security_rule:
    provider: '{{ provider }}'
    rule_name: 'HTTP Multimedia'
    description: 'Allow HTTP multimedia only to host at 1.1.1.1'
    source_zone: ['private']
    destination_zone: ['public']
    category: ['content-delivery-networks']
    application: ['http-video', 'http-audio']
    service: ['service-http', 'service-https']
    action: 'allow'
```

(continues on next page)

```
- name: add a more complex rule that uses security profiles
panos_security_rule:
  provider: '{{ provider }}'
  rule_name: 'Allow HTTP'
  source_zone: ['public']
  destination_zone: ['private']
  log_start: false
  log_end: true
  action: 'allow'
  antivirus: 'strict'
  vulnerability: 'strict'
  spyware: 'strict'
  url_filtering: 'strict'
  wildfire_analysis: 'default'

- name: disable a Panorama pre-rule
panos_security_rule:
  provider: '{{ provider }}'
  device_group: 'Production edge'
  rule_name: 'Allow telnet'
  source_zone: ['public']
  destination_zone: ['private']
  source_ip: ['any']
  destination_ip: ['1.1.1.1']
  log_start: false
  log_end: true
  action: 'allow'
  disabled: true

- name: delete a device group security rule
panos_security_rule:
  provider: '{{ provider }}'
  state: 'absent'
  device_group: 'DC Firewalls'
  rule_name: 'Allow telnet'

- name: add a rule at a specific location in the rulebase
panos_security_rule:
  provider: '{{ provider }}'
  rule_name: 'SSH permit'
  description: 'SSH rule test'
  source_zone: ['untrust']
  destination_zone: ['trust']
  source_ip: ['any']
  source_user: ['any']
  destination_ip: ['1.1.1.1']
  category: ['any']
  application: ['ssh']
  service: ['application-default']
  action: 'allow'
  location: 'before'
  existing_rule: 'Allow MySQL'
```


Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Ivan Bojer (@ivanbojer)
- Robert Hagen (@stealthllama)
- Michael Richardson (@mrichardson03)
- Garfield Lee Freeman (@shinmog)

3.2.65 panos_service_group – Create service group objects on PAN-OS devices

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create service group objects on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Create service group 'Prod-Services'
  panos_service_group:
    provider: '{{ provider }}'
    name: 'Prod-Services'
    value: ['ssh-tcp-22', 'mysql-tcp-3306']

- name: Delete service group 'Prod-Services'
  panos_service_group:
    provider: '{{ provider }}'
    name: 'Prod-Services'
    state: 'absent'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)

3.2.66 panos_service_object – Create service objects on PAN-OS devices

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create service objects on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
- Check mode is supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: Create service object 'ssh-tcp-22'
  panos_service_object:
    provider: '{{ provider }}'
    name: 'ssh-tcp-22'
    destination_port: '22'
    description: 'SSH on tcp/22'
    tag: ['Prod']

- name: Create service object 'mysql-tcp-3306'
  panos_service_object:
    provider: '{{ provider }}'
    name: 'mysql-tcp-3306'
    destination_port: '3306'
    description: 'MySQL on tcp/3306'

- name: Delete service object 'mysql-tcp-3306'
  panos_service_object:
    provider: '{{ provider }}'
    name: 'mysql-tcp-3306'
    state: 'absent'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)

3.2.67 panos_snmp_profile – Manage SNMP server profiles

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages SNMP server profiles.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create snmp profile
- name: Create snmp profile
  panos_snmp_profile:
    provider: '{{ provider }}'
    name: 'my-profile'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.68 panos_snmp_v2c_server – Manage SNMP v2c servers

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages SNMP v2c servers.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a snmp v2 server
- name: Create snmp v2 server
  panos_snmp_v2c_server:
    provider: '{{ provider }}'
    snmp_profile: 'my-profile'
    name: 'my-v2c-server'
    manager: '192.168.55.10'
    community: 'foobar'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.69 panos_snmp_v3_server – Manage SNMP v3 servers

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages SNMP v3 servers.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create snmp v3 server
- name: Create snmp v3 server
  panos_snmp_v3_server:
    provider: '{{ provider }}'
    snmp_profile: 'my-profile'
    name: 'my-v3-server'
    manager: '192.168.55.10'
    user: 'jdoe'
    auth_password: 'password'
    priv_password: 'drowssap'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.70 panos_software – Manage PAN-OS software versions

New in version 2.6.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Install specific release of PAN-OS.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```

- name: Install PAN-OS 8.1.6 and restart
  panos_software:
    provider: '{{ provider }}'
    version: '8.1.6'
    restart: true

- name: Download PAN-OS 9.0.0 base image only
  panos_software:
    provider: '{{ provider }}'
    version: '9.0.0'
    install: false
    restart: false

- name: Download PAN-OS 9.0.1 and sync to HA peer
  panos_software:
    provider: '{{ provider }}'
    version: '9.0.1'
    sync_to_peer: true
    install: false
    restart: false

```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)

3.2.71 panos_static_route – Create static routes on PAN-OS devices

New in version 2.6.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create static routes on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - IPv6 is not supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create route 'Test-One'
  panos_static_route:
    provider: '{{ provider }}'
    name: 'Test-One'
    destination: '1.1.1.0/24'
    nexthop: '10.0.0.1'

- name: Create route 'Test-Two'
  panos_static_route:
    provider: '{{ provider }}'
    name: 'Test-Two'
    destination: '2.2.2.0/24'
    nexthop: '10.0.0.1'

- name: Create route 'Test-Three'
  panos_static_route:
    provider: '{{ provider }}'
    name: 'Test-Three'
    destination: '3.3.3.0/24'
    nexthop: '10.0.0.1'

- name: Delete route 'Test-Two'
  panos_static_route:
```

(continues on next page)

(continued from previous page)

```
provider: '{{ provider }}'  
name: 'Test-Two'  
state: 'absent'  
  
- name: Create route 'Test-Four'  
  panos_static_route:  
    provider: '{{ provider }}'  
    name: 'Test-Four'  
    destination: '4.4.4.0/24'  
    nexthop: '10.0.0.1'  
    virtual_router: 'VR-Two'  
  
- name: Create route 'Test-Five'  
  panos_static_route:  
    provider: '{{ provider }}'  
    name: 'Test-Five'  
    destination: '5.5.5.0/24'  
    nexthop_type: 'none'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)
- Garfield Lee Freeman (@shinmog)

3.2.72 panos_syslog_profile – Manage syslog server profiles

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages syslog server profiles.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create a profile
- name: Create syslog profile
  panos_syslog_profile:
    provider: '{{ provider }}'
    name: 'my-profile'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.73 panos_syslog_server – Manage syslog server profile syslog servers

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manages syslog servers in an syslog server profile.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice >= 0.11.1

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Create syslog server
  panos_syslog_server:
    provider: '{{ provider }}'
    syslog_profile: 'my-profile'
    name: 'my-syslog-server'
    server: '10.1.1.1'
    syslog_port: 514
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.74 panos_tag_object – Create tag objects on PAN-OS devices

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Create tag objects on PAN-OS devices.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Create tag object 'Prod'
  panos_tag_object:
    provider: '{{ provider }}'
    name: 'Prod'
    color: 'red'
    comments: 'Prod Environment'

- name: Remove tag object 'Prod'
  panos_tag_object:
    provider: '{{ provider }}'
    name: 'Prod'
    state: 'absent'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Michael Richardson (@mrichardson03)

3.2.75 panos_tunnel – configure tunnel interfaces

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure tunnel interfaces on PanOS

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPi <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPi <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
# Create tunnel.1
- name: create tunnel.1
  panos_tunnel:
    provider: '{{ provider }}'
    if_name: "tunnel.1"
    ip: ["10.1.1.1/32"]

# Update tunnel comment.
- name: update tunnel.1 comment
  panos_tunnel:
    provider: '{{ provider }}'
    if_name: "tunnel.1"
    ip: ["10.1.1.1/32"]
    comment: "tunnel interface"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Joshua Colson (@freakinhippie)

3.2.76 panos_type_cmd – Execute arbitrary TYPE commands on PAN-OS

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- This module allows you to execute arbitrary TYPE commands on PAN-OS.
- This module does not provide guards of any sort, so USE AT YOUR OWN RISK.
- Refer to the PAN-OS and Panorama API guide for more info.
- <https://docs.paloaltonetworks.com/pan-os.html>

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is not supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
-

Examples

```
- name: Create an address object using set.
panos_type_cmd:
  provider: '{{ provider }}'
  xpath: |
    /config/devices/entry[@name='localhost.localdomain']
    /vsys/entry[@name='vsys1']
    /address
  element: |
    <entry name="sales-block">
      <ip-netmask>192.168.55.0/24</ip-netmask>
      <description>Address CIDR for sales org</description>
    </entry>

- name: Then rename it.
panos_type_cmd:
  provider: '{{ provider }}'
  cmd: 'rename'
  xpath: |
    /config/devices/entry[@name='localhost.localdomain']
    /vsys/entry[@name='vsys1']
    /address/entry[@name='sales-block']
  new_name: 'dmz-block'

- name: Show the address object.
panos_type_cmd:
  provider: '{{ provider }}'
  cmd: 'show'
  xpath: |
    /config/devices/entry[@name='localhost.localdomain']
    /vsys/entry[@name='vsys1']
    /address/entry[@name='dmz-block']
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.77 panos_userid – Allow for registration and de-registration of userid

New in version 2.6.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Userid allows for user to IP mapping that can be used in the policy rules.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is not supported.
- Panorama is not supported.
- This operation is runtime and does not require explicit commit of the firewall configuration.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
- name: Register user ivanb to 10.0.1.101
  panos_userid:
    provider: '{{ provider }}'
    userid: 'ACMECORP\ivanb'
    register_ip: '10.0.1.101'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Ivan Bojer (@ivanbojer)

3.2.78 panos_virtual_router_facts – Retrieves virtual router information

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Retrieves information on virtual routers from a firewall or Panorama.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Panorama is supported.
- Check mode is not supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
- If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.

Examples

```
# Get information on a specific virtual router
- name: Get vr3 info
  panos_virtual_router_facts:
    provider: '{{ provider }}'
    name: 'vr3'
    register: ans

# Get the config of all virtual routers
- name: Get all virtual routers
  panos_virtual_router_facts:
    provider: '{{ provider }}'
    register: vrlist
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.79 panos_virtual_router – Configures a Virtual Router

New in version 2.9.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manage PANOS Virtual Router

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create Virtual Router
  panos_virtual_router:
    provider: '{{ provider }}'
    name: vr-1
    commit: true
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Joshua Colson (@freakinhippie)
- Garfield Lee Freeman (@shinmog)

3.2.80 panos_virtual_wire – Configures Virtual Wires (vwire)

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manage PAN-OS Virtual Wires (vwire).

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create Vwire
  panos_virtual_wire:
    provider: '{{ provider }}'
    name: 'vwire1'
    interface1: 'ethernet1/1'
    interface2: 'ethernet1/2'
    tag: 100
```

(continues on next page)

(continued from previous page)

```
multicast: 'true'  
pass_through: 'true'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Patrick Avery

3.2.81 panos_vlan_interface – configure VLAN interfaces

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure VLAN interfaces.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Checkmode is supported.
- If the PAN-OS device is a firewall and *vsys* is not specified, then the *vsys* will default to *vsys=vsys1*.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.

Examples

```
# Create vlan.2 as DHCP
- name: enable DHCP client on ethernet1/1 in zone public
  panos_vlan_interface:
    provider: '{{ provider }}'
    name: "vlan.2"
    zone_name: "public"
    enable_dhcp: true
    create_default_route: true

# Set vlan.7 with a static IP
- name: Configure vlan.7
  panos_vlan_interface:
    provider: '{{ provider }}'
    name: "vlan.7"
    ip: ["10.1.1.1/24"]
    management_profile: "allow ping"
    vlan_name: "dmz"
    zone_name: "L3-untrust"
    vr_name: "default"
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.82 panos_vlan – Configures VLANs

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Manage PAN-OS VLANs.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Checkmode is supported.
 - Panorama is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
- name: Create VLAN
  panos_vlan:
    provider: '{{ provider }}'
    name: 'Internal'
    virtual_interface: 'vlan.2'
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Garfield Lee Freeman (@shinmog)

3.2.83 panos_zone_facts – Retrieves zone information

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Return Values*
- *Status*

Synopsis

- Retrieves information on zones from a firewall or Panorama.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python
- pandevice

Parameters

Notes

Note:

- Panorama is supported.
- Check mode is not supported.
- PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
- If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.

Examples

```
# Get information on a specific zone
- name: Get zone3 info
  panos_zone_facts:
    provider: '{{ provider }}'
    name: 'zone3'
    register: ans

# Get the config of all zones
- name: Get all zones
  panos_zone_facts:
    provider: '{{ provider }}'
    register: zones
```

Return Values

Common return values are [documented here](#), the following are the fields unique to this module:

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is [maintained by the Ansible Community](#).

Authors

- Garfield Lee Freeman (@shinmog)

3.2.84 panos_zone – configure security zone

New in version 2.8.

- *Synopsis*
- *Requirements*
- *Parameters*
- *Notes*
- *Examples*
- *Status*

Synopsis

- Configure security zones on PAN-OS firewall or in Panorama template.

Requirements

The below requirements are needed on the host that executes this module.

- pan-python can be obtained from PyPI <https://pypi.python.org/pypi/pan-python>
- pandevice can be obtained from PyPI <https://pypi.python.org/pypi/pandevice>
- pandevice >= 0.8.0

Parameters

Notes

Note:

- Panorama is supported.
 - Check mode is supported.
 - PAN-OS connectivity should be specified using *provider* or the classic PAN-OS connectivity params (*ip_address*, *username*, *password*, *api_key*, and *port*). If both are present, then the classic params are ignored.
 - If the PAN-OS to be configured is Panorama, either *template* or *template_stack* must be specified.
-

Examples

```
# Create an L3 zone.
- name: create DMZ zone on a firewall
  panos_zone:
    provider: '{{ provider }}'
    zone: 'dmz'
    mode: 'layer3'
    zone_profile: 'strict'

# Add an interface to the zone.
- name: add ethernet1/2 to zone dmz
  panos_interface:
    provider: '{{ provider }}'
    zone: 'dmz'
    mode: 'layer3'
    interface: ['ethernet1/2']
    zone_profile: 'strict'

# Delete the zone.
- name: delete the DMZ zone
  panos_interface:
    provider: '{{ provider }}'
    zone: 'dmz'
    state: 'absent'
```

(continues on next page)

(continued from previous page)

```
# Add a zone to a multi-VSYS Panorama template
- name: add Cloud zone to template
  panos_interface:
    provider: '{{ provider }}'
    template: 'Datacenter Template'
    vsys: 'vsys4'
    zone: 'datacenter'
    mode: 'layer3'
    enable_userid: true
    exclude_acl: ['10.0.200.0/24']
```

Status

- This module is not guaranteed to have a backwards compatible interface. *[preview]*
- This module is maintained by the Ansible Community.

Authors

- Robert Hagen (@stealthllama)
- Garfield Lee Freeman (@shinmog)

3.3 Release History

3.3.1 V2.4.0

- *Released:* 2020-01-17

New modules:

- panos_pbf_rule
- panos_gre_tunnel
- panos_nat_rule_facts

Enhancements:

- panos_aggregate_interface has had its params fixed
- panos_ike_gateway does not force peer_id_check anymore
- panos_software has new flags allowing just download or just install
- panos_object_facts has new search options
- Various bug and doc fixes / updates

3.3.2 V2.3.0

- *Released:* 2019-12-09

New modules:

- `panos_virtual_wire`
- `panos_aggregate_interface`
- `panos_ha`

Enhancements:

- `panos_op` now reports `changed=False` for safe commands
- `panos_l3_subinterface` can now create aggregate subinterfaces
- `panos_commit` can now perform partial commits
- `panos_type_cmd` now reports `changed=False` for safe commands
- `panos_bgp` now supports `enable` and `disable` as states
- `panos_match_rule` should now work with rules that came from Panorama

Bug Fixes:

- `panos_ike_gateway`
- `panos_mgtconfig`
- `panos_bgp_auth`
- `panos_vlan`

New modules:

- `panos_virtual_wire`

Enhancements:

- add list of op commands that will not produce changes
- Add list of type commands that will not produce changes
- Return output from commands in `panos_type_cmd`
- Fixed `panos_commit` returning `changed` as `True`, even if a commit was not performed.

3.3.3 V2.2.3

- *Released:* 2019-07-09

Bug fixes:

- Fixed `create_default_route` handling with both `panos_interface` and `panos_l3_subinterface`

3.3.4 V2.2.2

- *Released:* 2019-06-18

Bug fixes:

- Fixed module handling when targeting Panorama template stacks

3.3.5 V2.2.1

- *Released:* 2019-06-12

Bug fixes:

- Fixed panos_l3_subinterface DHCP handling to match panos_interface

3.3.6 V2.2.0

- *Released:* 2019-06-11

New modules:

- panos_zone_facts
- panos_ipsec_ipv4_proxyid
- panos_virtual_router_facts
- panos_l3_subinterface
- panos_l2_subinterface
- panos_log_forwarding_profile
- panos_log_forwarding_profile_match_list
- panos_log_forwarding_profile_match_list_action
- panos_email_profile
- panos_email_server
- panos_snmp_profile
- panos_snmp_v2c_server
- panos_snmp_v3_server
- panos_syslog_profile
- panos_syslog_server
- panos_http_profile
- panos_http_profile_header
- panos_http_profile_param
- panos_http_server
- panos_type_cmd

Enhancements:

- panos_security_rule_facts can now return full policy info.

Bug fixes:

- Added module encoding to all modules.
- Various documentation fixes.

3.3.7 V2.1.2

- *Released:* 2019-05-24

Enhancements:

- `panos_registered_ip_facts` can now filter on IP addresses (in addition to tags)
- All modules: Panorama commits can now still push to a device group even if a Panorama commit is unnecessary
- `panos_nat_rule`: Changed the default location to unspecified instead of bottom

Bug fixes:

- `panos_bgp`: Added the “state” param to this module
- `panos_facts`: Corrected virtual router output name to use underscores

3.3.8 V2.1.1

- *Released:* 2019-05-08
- *Status:* Released
- Minor bug fix with `panos_op`
- Documentation tweaks

3.3.9 V2.1.0

- *Released:* 2019-04-26
- *Status:* Released

New modules:

- `panos_security_rule_facts`
- `panos_vlan`
- `panos_vlan_interface`

The following shorthand applies to this version’s updates:

- `provider` - Any module below that lists a change of `provider` means that it supports a new provider dict for PAN-OS authentication credentials in addition to the old `ip_address/username/password/api_key`. Additionally these modules now support Panorama to firewall connections, performed by specifying Panorama IP address, username, and password, then specifying a firewall’s serial number using the `serial_number` param in the `provider` dict.
- `removed operation` - This module has had the old `operation` param removed in favor of `state`. Please update your playbooks to use `state` instead.
- `template support` - This module now supports Panorama templates.
- `full template support` - This module now supports both Panorama templates and template stacks.

- `vsys` support - This module now includes support for specifying the firewall `vsys`.
- `checkmode` - This module now supports Ansible's check mode.

Given the above shorthand, the following modules have been updated as follows:

- `panos_address_group`: provider; checkmode
- `panos_address_object`: provider; checkmode
- `panos_administrator`: provider; full template support; checkmode; Now supports supplying the password hash directly
- `panos_api_key`: provider
- `panos_bgp`: provider; full template support; checkmode
- `panos_bgp_aggregate`: provider; full template support; checkmode
- `panos_bgp_auth`: provider; full template support; checkmode; `replace` is deprecated as this is now the default behavior for `state=apply`
- `panos_bgp_conditional_advertisement`: provider; full template support; checkmode; `advertise_filter` and `non_exist_filter` have been deprecated, add filters using `panos_bgp_policy_filter` instead
- `panos_bgp_dampening`: provider; full template support; checkmode
- `panos_bgp_peer`: provider; full template support; checkmode
- `panos_bgp_peer_group`: provider; full template support; checkmode
- `panos_bgp_policy_filter`: provider; full template support; checkmode; “`state=return-object`” has been deprecated, just use states of `absent/present` like other modules as normal; `address_prefix` can now be a dict with “`name`”/“`exact`” keys or a string
- `panos_bgp_policy_rule`: provider; full template support; checkmode; `address_prefix` can now be a dict with “`name`”/“`exact`” keys or a string
- `panos_bgp_redistribute`: provider; full template support; checkmode
- `panos_check`: provider; fixed #183; fixed #311
- `panos_commit`: provider; added `include_template` param; `devicegroup` is deprecated, use `device_group` instead
- `panos_facts`: provider; fixed bug when running against VM NGFW; `host` has been removed, use `provider` instead
- `panos_ike_crypto_profile`: provider; full template support; checkmode
- `panos_ike_gateway`: provider; full template support; checkmode; many params have been aliased to new param names to better match the `pandevice` naming
- `panos_interface`: provider; template support; checkmode; removed operation; fixed #193; fixed #266; fixed #267; `vsys_dg` is deprecated, use `vsys` instead
- `panos_ipsec_profile`: provider; full template support; checkmode
- `panos_ipsec_tunnel`: provider; full template support; checkmode; many new params added to support missing functionality added in, please refer to the module documentation for the complete list of params now supported
- `panos_lic`: provider; added new output licenses
- `panos_loopback_interface`: provider; template support; checkmode; `vsys_dg` is deprecated; use `vsys` instead

- `panos_management_profile`: provider; full template support; checkmode; `panorama_template` is deprecated, use `template` instead
- `panos_match_rule`: provider; `vsys_id` is deprecated, use `vsys`; fixed #248; output `stdout_lines` is deprecated, use `rule` instead (note: this has a different format, so please update your playbooks)
- `panos_mgtconfig`: provider; checkmode; `devicegroup` is removed as this param was not doing anything; added `verify_update_server`
- `panos_nat_rule`: provider; removed operation; checkmode; `devicegroup` is deprecated, use `device_group`; `tag_name` (string type) is deprecated, use `tag` (list type); added `enable` and `disable` types for the `state` param
- `panos_object_facts`: provider; added support for name regexes and a new `objects` output
- `panos_op`: provider
- `panos_pg`: provider; added Panorama support; added `state`
- `panos_redistribution`: provider; full template support; checkmode
- `panos_registered_ip`: provider; `vsys` support; checkmode
- `panos_registered_ip_facts`: provider; `vsys` support
- `panos_restart`: provider
- `panos_security_rule`: provider; removed operation; checkmode; `devicegroup` is deprecated, use `device_group` instead
- `panos_service_group`: provider; checkmode
- `panos_service_object`: provider; checkmode
- `panos_software`: provider; checkmode
- `panos_static_route`: provider; full template support; added `nexthop` type of “next-vr”
- `panos_tag_object`: provider; checkmode
- `panos_tunnel`: provider; template support; checkmode; `vsys_dg` is deprecated, use `vsys` instead
- `panos_userid`: provider; removed operation; `state` added as a param
- `panos_virtual_router`: provider; full template support; checkmode
- `panos_zone`: provider; full template support; checkmode

Generic updates across all modules mentioned above:

- The minimum version of `pandevice` to run all “provider” modules is 0.9.1
- Cleaned up module documentation

The following modules have been deprecated:

- `panos_admin`
- `panos_dag`
- `panos_query_rules`
- `panos_sag`

The following modules have not been modified:

- `panos_admpwd`
- `panos_cert_gen_ssh`

- panos_dag_tags
- panos_import
- panos_loadcfg
- panos_object

3.3.10 V2.0.4

- Released: 2019-03-11
- Status: Released (minor)
- Fixes the DHCP param handling of panos_interface

3.3.11 V2.0.3

- Released: 2019-03-04
- Status: Released

New modules

- panos_api_key: retrieve api_key for username/password combination
- panos_bgp: Manages basic BGP configuration settings
- panos_bgp_aggregate: Manages BGP Aggregation Policy Rules
- panos_bgp_auth: Manages BGP Authentication Profiles
- panos_bgp_conditional_advertisement: Manages BGP Conditional Advertisement Policy Rules
- panos_bgp_dampening: Manages BGP Dampening Profiles
- panos_bgp_peer: Manages BGP Peers
- panos_bgp_peer_group: Manages BGP Peer Groups
- panos_bgp_policy_filter: Manages BGP Policy Filters, children of Aggregate and Conditional Advertisement
- panos_bgp_policy_rule: Manage BGP Import/Export Rules
- panos_bgp_redistribute: Manages BGP Redistribution Rules
- panos_loopback_interface: manage loopback interfaces
- panos_redistribution: Manages virtual router Redistribution Profiles

Refactored modules

- panos_ike_gateway: fixed misspelling of passive_mode and added additional module arguments to support more advanced configurations

3.3.12 V2.0.1

- Released: 2018-10-08
- Status: Released (minor)

This is minor release to address issue <https://github.com/PaloAltoNetworks/ansible-pan/issues/163>

3.3.13 V2.0.0

- Released: 2018-09-27
- Status: Released

New modules

- panos_administrator: Manages Panorama / NGFW administrators
- panos_registered_ip: Use this instead of panos_dag_tags
- panos_registered_ip_facts: Use this instead of panos_dag_tags
- panos_address_object: Use this instead of panos_object
- panos_address_group: Use this instead of panos_object
- panos_service_object: Use this instead of panos_object
- panos_service_group: Use this instead of panos_object
- panos_tag_object: Use this instead of panos_object
- panos_object_facts: Get facts about objects

Removed modules

Refactored modules

Now supporting state / idempotency

- panos_interface
- panos_nat_rule
- panos_security_rule

Miscellanies / Fixes

- merged Ansible role repo together with this one
- <https://github.com/PaloAltoNetworks/ansible-pan/issues/44>
- adding beta support for connections lib
- <https://github.com/PaloAltoNetworks/ansible-pan/issues/150>

3.3.14 V1.0.8

- Released: 2018-09-13
- Status: Released

New modules

- `panos_management_profile`: Manages interface management profiles
- `panos_ike_crypto_profile`: Use the IKE Crypto Profiles page to specify protocols and algorithms for identification, authentication, and encryption (IKEv1 or IKEv2, Phase 1).
- `panos_ipsec_profile`: Configures IPsec Crypto profile on the firewall with subset of settings.
- `panos_ike_gateway`: Configures IKE gateway on the firewall with subset of settings.
- `panos_ipsec_tunnel`: Configure data-port (DP) network interface for DHCP. By default DP interfaces are static.

Removed modules

Refactored modules

Miscellanies

- `panos_security_rule` - New `[log_setting]{.title-ref}` param added to specify the log forwarding profile to be used
- re-wrote documentation

3.3.15 V1.0.7

- Released: 2018-05-03
- Status: Released

New modules

- `panos_userid`: added ability to (un)register userid with ip address
- `panos_software`: Upgrade and downgrade PAN-OS on firewalls and Panorama.
- `panos_userid`: added ability to (un)register userid with ip address
- `panos_static_route`: ability to manipulate static routing tables

Removed modules

N/A

Refactored modules

-

```
panos\_interface: Added full support for static configuration of ethernet interfaces
: - <https://github.com/PaloAltoNetworks/ansible-pan/pull/61>
```

-

```
Add functionality to list static address groups
: - <https://github.com/PaloAltoNetworks/ansible-pan/pull/64>
```

-

```
Pass api\_key to pandevice
```

```
: - <https://github.com/PaloAltoNetworks/ansible-pan/pull/63>
```

•

```
panos\_security\_rule: Security Policy position/order
```

```
: - <https://github.com/PaloAltoNetworks/ansible-pan/issues/14>
```

•

```
panos\_security\_rule: unable to add security policies in Post rule
```

```
: - <https://github.com/PaloAltoNetworks/ansible-pan/issues/38>
```

Miscellanies - <https://github.com/PaloAltoNetworks/ansible-pan/pull/78> - <https://github.com/PaloAltoNetworks/ansible-pan/issues/22>

3.3.16 V1.0.6

- Released: 2018-2-6
- Status: Released

New modules

N/A

Removed modules

N/A

Miscellanies

•

```
Synchronized repository with RedHat Ansible official repo. Added missing modules:
```

```
: - panos\_op.py  
  - panos\_dag\_tags.py  
  - panos\_query\_rules.py  
  - panos\_match\_rule.py
```

Closed issues

- <https://github.com/PaloAltoNetworks/ansible-pan/issues/52>
- <https://github.com/PaloAltoNetworks/ansible-pan/issues/46>

3.3.17 V1.0.5

- Released: 2017-12-20
- Status: Released

New modules

- `panos_op`: OP commands module that allows execution of the arbitrary op commands on the PANOS devices

Refactored modules

N/A

Removed modules

N/A

Miscellanies

N/A

Closed issues

#36 <https://github.com/PaloAltoNetworks/ansible-pan/issues/36>

3.3.18 V1.0.4

- Released: 2017-08-31
- Status: Released

New modules

- `panos_sag`: Added the ability to add / delete static address groups.
-

```
panos\_dag\_tags: A new module to create registered IP to tag associations
: Implemented the ability to create / delete / list IP to tag
  associations
```

- `panos_security_rule`
- `panos_nat_rule`

Refactored modules

- `panos_restart` refactored to use `PanDevice` internally; supports Panorama
- `panos_mgtconfig` refactored to use `PanDevice` internally; added support for NTP servers config
-

```
panos\_dag: Converted the module to use pandevice
: Also added the ability to perform create / delete / list
```

Removed modules

- `panos_nat_policy` (Use `panos_nat_rule`)
- `panos_nat_security_policy` (use `panos_security_rule`)

- panos_service (use panos_object)

Miscellanies

- removed deprecated_libraries folder
- consolidated all samples from samples/ into examples/
- synchronized repo with core Ansible distribution

3.3.19 V1.0.3

Minor release with documentation updates and few BUG fixes.

3.3.20 V1.0.2

- Released: 2017-04-13

Another major refactor in order to streamline the code.

- Refactored modules
- panos_address --> panos_object
- panos_match_rule
- panos_nat_policy --> panos_nat_rule
- panos_query_rules
- panos_security_policy --> panos_security_rule
- panos_service --> panos_object

3.3.21 V1.0.1

- Released: 2017-02-15
- Status: Release

All modules have been touched and refactored to adhere to Ansible module development practices. Documentation has been added as well as sample playbooks for each module.

Refactored modules (now part of core Ansible)

- panos_admin
- panos_admpwd
- panos_commit
- panos_restart
- panos_cert_gen_ssh
- panos_check
- panos_dag
- panos_service
- panos_mgtconfig
- panos_import

- panos_loadcfg
- panos_pg
- panos_lic
- panos_interface

New modules

- panos_address
- panos_security_policy

Deprecated modules

- panos_srule
- panos_content
- panos_swininstall
- panos_tunnelif
- panos_cstapphost
- panos_gpp_gateway
- panos_vulnprofile
- panos_swapif
- panos_vulnprofile

3.3.22 V1.0.0

- Released: 2016-11-27
- Status: Release

First release that adheres to the Ansible development practices, now part of the Ansible core development. The modules have been completely refactored. Some retired and some new modules created.

3.3.23 V0.1.3

- Released: 2015-12-09
- Status: Alpha

Bug fixes and documentation updates

3.3.24 Alpha

- Released: 2015-07-28
- Status: Alpha

First alpha and documentation

3.4 Contributing to PANW Ansible modules

3.5 Developing Palo Alto Networks Ansible Modules

(draft)

3.5.1 Should you develop a module?

Developing PANW Ansible modules is easy, but often it isn't necessary. Before you start writing a new module, ask:

Does a similar module already exist?

An existing module may cover the functionality you want. You might just need additional functionality in the existing module. If you are not sure feel free to email PANW maintainers.

Does a Pull Request already exist?

An existing Pull Request may cover the functionality you want. If someone else has already started developing a similar module, you can review and test it.

- GitHub new module PRs <https://github.com/PaloAltoNetworks/ansible-pan/pulls>
- Already closed but not yet released modules <https://github.com/PaloAltoNetworks/ansible-pan/blob/develop/docs/history.md>

If you find an existing PR that looks like it addresses your needs, please provide feedback on the PR. Community feedback speeds up the review and merge process.

Should you write multiple modules instead of one module?

The functionality you want may be too large for a single module. You might want to split it into separate modules or enhance already existing module.

3.5.2 Contributing to codebase

If your use case isn't covered by an existing module or an open PR then you're ready to start developing a new module.

In order to do this you need to (draft):

1. fork develop branch (**NOT MASTER**)
2. do your changes
 - update / change module
 - update `history.md` with changes
 - make sure you run code through linter (TBD)
3. create pull request against **DEVELOP** branch
 - sometimes it is necessary to rebase your changes. If you need more info on how to do this there is a good write-up that can be applied in our case: https://docs.ansible.com/ansible/2.5/dev_guide/developing_rebasing.html

3.6 Authors

Development Leads

- Ivan Bojer (@ivanbojer)
- Garfield Lee Freeman (@shinmog)

Contributors

- Robert Hagen (@rnh556)
- Luigi Mori (@jtschichold)
- Vinay Venkataraghavan (@vinayvenkat)
- Michael Richardson (@mrichardson03)
- Joshua Colson (freakinhippie)

Credits

Thank you Kevin Steves, creator of the pan-python library. (<https://github.com/kevinsteves/pan-python>)

Also, big high-five to Brian Torres-Gil, creator of the pandevice library. (<https://github.com/PaloAltoNetworks/pandevice>)

3.7 License

```
Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction,
and distribution as defined by Sections 1 through 9 of this document.

"Licenser" shall mean the copyright owner or entity authorized by
the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all
other entities that control, are controlled by, or are under common
control with that entity. For the purposes of this definition,
"control" means (i) the power, direct or indirect, to cause the
direction or management of such entity, whether by contract or
otherwise, or (ii) ownership of fifty percent (50%) or more of the
outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity
exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications,
including but not limited to software source code, documentation
source, and configuration files.

"Object" form shall mean any form resulting from mechanical
```

(continues on next page)

(continued from previous page)

transformation **or** translation of a Source form, including but **not** limited to compiled **object** code, generated documentation, **and** conversions to other media types.

"Work" shall mean the work of authorship, whether **in** Source **or** Object form, made available under the License, **as** indicated by a copyright notice that **is** included **in or** attached to the work (an example **is** provided **in** the Appendix below).

"Derivative Works" shall mean **any** work, whether **in** Source **or** Object form, that **is** based on (**or** derived from) the Work **and for** which the editorial revisions, annotations, elaborations, **or** other modifications represent, **as** a whole, an original work of authorship. For the purposes of this License, Derivative Works shall **not** include works that remain separable from, **or** merely link (**or** bind by name) to the interfaces of, the Work **and** Derivative Works thereof.

"Contribution" shall mean **any** work of authorship, including the original version of the Work **and any** modifications **or** additions to that Work **or** Derivative Works thereof, that **is** intentionally submitted to Licensor **for** inclusion **in** the Work by the copyright owner **or** by an individual **or** Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means **any** form of electronic, verbal, **or** written communication sent to the Licensor **or** its representatives, including but **not** limited to communication on electronic mailing lists, source code control systems, **and** issue tracking systems that are managed by, **or** on behalf of, the Licensor **for** the purpose of discussing **and** improving the Work, but excluding communication that **is** conspicuously marked **or** otherwise designated **in** writing by the copyright owner **as** "Not a Contribution."

"Contributor" shall mean Licensor **and any** individual **or** Legal Entity on behalf of whom a Contribution has been received by Licensor **and** subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms **and** conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, **and** distribute the Work **and** such Derivative Works **in** Source **or** Object form.
3. Grant of Patent License. Subject to the terms **and** conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (**except as** stated **in** this section) patent license to make, have made, use, offer to sell, sell, import, **and** otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone **or** by combination of their Contribution(s) **with** the Work to which such Contribution(s) was submitted. If You institute patent litigation against **any** entity (including a cross-claim **or** counterclaim **in** a lawsuit) alleging that the Work **or** a Contribution incorporated within the Work constitutes direct **or** contributory patent infringement, then **any** patent licenses granted to You under this License **for** that Work shall terminate **as** of the date such litigation **is** filed.

(continues on next page)

(continued from previous page)

4. Redistribution. You may reproduce **and** distribute copies of the Work **or** Derivative Works thereof **in any** medium, **with or** without modifications, **and in** Source **or** Object form, provided that You meet the following conditions:
 - (a) You must give **any** other recipients of the Work **or** Derivative Works a copy of this License; **and**
 - (b) You must cause **any** modified files to carry prominent notices stating that You changed the files; **and**
 - (c) You must retain, **in** the Source form of **any** Derivative Works that You distribute, **all** copyright, patent, trademark, **and** attribution notices **from the** Source form of the Work, excluding those notices that do **not** pertain to **any** part of the Derivative Works; **and**
 - (d) If the Work includes a "NOTICE" text file **as** part of its distribution, then **any** Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do **not** pertain to **any** part of the Derivative Works, **in** at least one of the following places: within a NOTICE text file distributed **as** part of the Derivative Works; within the Source form **or** documentation, **if** provided along **with** the Derivative Works; **or**, within a display generated by the Derivative Works, **if and** wherever such third-party notices normally appear. The contents of the NOTICE file are **for** informational purposes only **and** do **not** modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside **or as** an addendum to the NOTICE text **from the** Work, provided that such additional attribution notices cannot be construed **as** modifying the License.

You may add Your own copyright statement to Your modifications **and** may provide additional **or** different license terms **and** conditions **for** use, reproduction, **or** distribution of Your modifications, **or for any** such Derivative Works **as** a whole, provided Your use, reproduction, **and** distribution of the Work otherwise complies **with** the conditions stated **in** this License.

5. Submission of Contributions. Unless You explicitly state otherwise, **any** Contribution intentionally submitted **for** inclusion **in** the Work by You to the Licensor shall be under the terms **and** conditions of this License, without **any** additional terms **or** conditions. Notwithstanding the above, nothing herein shall supersede **or** modify the terms of **any** separate license agreement you may have executed **with** Licensor regarding such Contributions.
6. Trademarks. This License does **not** grant permission to use the trade names, trademarks, service marks, **or** product names of the Licensor, **except as** required **for** reasonable **and** customary use **in** describing the origin of the Work **and** reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law **or** agreed to **in** writing, Licensor provides the Work (**and** each

(continues on next page)

(continued from previous page)

Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express **or** implied, including, without limitation, any warranties **or** conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, **or** FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible **for** determining the appropriateness of using **or** redistributing the Work **and** assume any risks associated **with** Your exercise of permissions under this License.

8. Limitation of Liability. In no event **and** under no legal theory, whether **in** tort (including negligence), contract, **or** otherwise, unless required by applicable law (such **as** deliberate **and** grossly negligent acts) **or** agreed to **in** writing, shall any Contributor be liable to You **for** damages, including any direct, indirect, special, incidental, **or** consequential damages of any character arising **as** a result of this License **or** out of the use **or** inability to use the Work (including but **not** limited to damages **for** loss of goodwill, work stoppage, computer failure **or** malfunction, **or** any **and** all other commercial damages **or** losses), even **if** such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty **or** Additional Liability. While redistributing the Work **or** Derivative Works thereof, You may choose to offer, **and** charge a fee **for**, acceptance of support, warranty, indemnity, **or** other liability obligations **and/or** rights consistent **with** this License. However, **in** accepting such obligations, You may act only on Your own behalf **and** on Your sole responsibility, **not** on behalf of any other Contributor, **and** only **if** You agree to indemnify, defend, **and** hold each Contributor harmless **for** any liability incurred by, **or** claims asserted against, such Contributor by reason of your accepting any such warranty **or** additional liability.

END OF TERMS AND CONDITIONS

INDICES AND TABLES

- genindex
- modindex
- search