# Adding Firewall Policies

At this point in the lab your vSRX will have access to the rest of the lab. However your NetDevOps appliance will not. To allow this we must first enable firewall policies on your vSRX device.

# Using Ansible to enable firewall policies

For this step we will use Ansible to create both our firewall policies and the required address book objects. This is one of the most requested automation elements when dealing with a firewall. Typically once a firewall is deployed and integrated into the network there are few changes that are required on the networking side of things. However the majority of heavy lifting is centered around managing security policies and the associated address book objects. Address book objects are typically quite painful due to the need to create, manage, and have the correct association to zones. Luckily using Ansible makes these tasks a snap.

## Reviewing the playbook

First let's take a look at the playbook that is used to accomplish this task. We briefly looked at this playbook during the Ansible overview section, but now we will dive deeper into the applied steps.

**Playbook Review**

1. Define the name of the playbook - Configure basic firewall policies
    - This will be displayed and logged as you start to run the playbook
2. Define the hosts the playbook should be applied to
    - In this case we use the group "**mysrx**" to apply to
        - The host list is picked up from either the default Ansible host list in "/etc/ansible/hosts"
        - Alternatively when the playbook is run you can specify your own custom inventory
3. Connection is defined to as local - Typically when Ansible runs it transports an execution environment over to the host and runs it
    - Because this will not work on Junos hosts we use connection defined to local to run the execution environment
4. Gather facts
    - Ansible will gather local facts about the host such as interfaces and hostnames
    - Because this isn't possible on Junos we disable this feature
5. Vars - These are the variables that we will use to apply to our tasks
    - They can be applied at many different locations for our run
    - But to keep everything together we have included the variables into the playbook
    - address_entires will be used to generate the address book entries
    - fw_policy_info will be used to define our policies
6. Tasks - These are the tasks that we will use
    - The build phase for the playbook generates the Junos config from the templates
    - The apply phase will apply the configuration to the device
    - This will be run as two separate commits, but in doing so we can simplify the tasks and see

which step fails

**Playbook**

```
---
- name: Configure basic firewall policies
  hosts: mysrx
  connection: local
  gather_facts: no
  vars:
    junos_user: "root"
    junos_password: "Juniper"
    build_dir: "/tmp/"
    address_entries: [ {'name':'LocalNet','prefix':'172.16.0.0/24'},{'name':'PrivateNet','prefi
    fw_policy_info: [ {'policy_name':'Allow_Policy','src_zone':'trust','dst_zone':'untrust','sr

  tasks:
    - name: Build address book entries
      template: src=templates/fw_address_book_global.set.j2 dest={{build_dir}}/fw_address_book_
      with_items: address_entries

    - name: Apply address book entries
      junos_install_config: host={{ inventory_hostname }} user={{ junos_user }} passwd={{ junos_


    - name: Build firewall policies config template
      template: src=templates/fw_policy.set.j2 dest={{build_dir}}/fw_policy.set
      with_items: fw_policy_info

    - name: Apply firewall policies
      junos_install_config: host={{ inventory_hostname }} user={{ junos_user }} passwd={{ junos_
```

**Address book template**

- The address book template loops through the address entries in the variable
- It generates one "set" configuration line per loop
- Here we are generating three lines

```
{% for i in address_entries %}
set security address-book global address {{ i.name }} {{ i.prefix }}
{% endfor %}
```

**Output after generation**

- This is the generated output from the template being applied with variables
- These commands are then committed to Junos
- If one or more of the entries are already created it will recognize this as "OK"

```
set security address-book global address LocalNet 172.16.0.0/24
set security address-book global address PrivateNet 192.168.10.0/24
set security address-book global address PublicNet 10.10.0.0/22
```

**Policy Template**

This template is a bit more complex. We need to loop through source IPs, Destination IPs, and applications. Each loop through these variables will generate a single line of "set" commands. Creating the template this way allows us to reuse it in the future when we have a larger list of addresses and applications to apply.

```
{% for item in fw_policy_info %}
    {% for i in item.src_ips %}
set security policies from-zone {{ item.src_zone }} to-zone {{ item.dst_zone }} policy {{ item.
    {% endfor %}
    {% for i in item.dst_ips %}
set security policies from-zone {{ item.src_zone }} to-zone {{ item.dst_zone }} policy {{ item.
    {% endfor %}
    {% for i in item.apps %}
set security policies from-zone {{ item.src_zone }} to-zone {{ item.dst_zone }} policy {{ item.
    {% endfor %}
set security policies from-zone {{ item.src_zone }} to-zone {{ item.dst_zone }} policy {{ item.
{% endfor %}
```

#### Output after generation

Once run here are the set commands that will be loaded onto the device. Again if additional elements are added they will be generated into individual set commands.

```
set security policies from-zone trust to-zone untrust policy Allow_Policy match source-address
set security policies from-zone trust to-zone untrust policy Allow_Policy match destination-add
set security policies from-zone trust to-zone untrust policy Allow_Policy match application any
set security policies from-zone trust to-zone untrust policy Allow_Policy then permit
```

# Running the playbook

To run the playbook you must use the "ansible-playbook" command. We must specify the inventory file and the playbook to apply. The templates will be automatically loaded from the playbook. Since cowsay is installed it will also add the comical cow for our enjoyment. If you dislike our bovine friend then you can simply remove cowsay from your running host.

#### Playbook Command

Ensure before running the command you are in the "**ansible**" directory.

```
vagrant@NetDevOps-Student:~/JNPRAutomateDemo-Student/ansible$ ansible-playbook -i inventory.yml
```

#### Playbook Run Example

Once run the output should look like the following

```
vagrant@NetDevOps-Student:~/JNPRAutomateDemo-Student/ansible$ ansible-playbook -i inventory.yml
 ________________________________________
< PLAY [Configure basic firewall policies] >
 ----------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


 _____________________________
< TASK: Build address book entries >
 -----------------------------
```

```
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


changed: [172.16.0.1] => (item={'prefix': '172.16.0.0/24', 'name': 'LocalNet'})
changed: [172.16.0.1] => (item={'prefix': '192.168.10.0/24', 'name': 'PrivateNet'})
changed: [172.16.0.1] => (item={'prefix': '10.10.0.0/22', 'name': 'PublicNet'})
 ____________________________________
< TASK: Apply address book entries >
 ------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


changed: [172.16.0.1]
 ________________________________________________
< TASK: Build firewall policies config template >
 ------------------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


changed: [172.16.0.1] => (item={'src_zone': 'trust', 'dst_zone': 'untrust', 'src_ips': ['LocalN
 _________________________________
< TASK: Apply firewall policies >
 ---------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


changed: [172.16.0.1]
 _____________
< PLAY RECAP >
 -------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


172.16.0.1                 : ok=4    changed=4    unreachable=0    failed=0
```

### Validating the playbook run

Now connect to your vSRX instance from your NetDevOpsVM and validate the change

```
vagrant@NetDevOps-Student:~/JNPRAutomateDemo-Student/ansible$ ssh root@172.16.0.1
Password:
--- JUNOS 12.1X47-D20.7 built 2015-03-03 21:53:50 UTC
croot@NetDevOps-SRX01% cli
```

```
root@NetDevOps-SRX01> show configuration security address-book
global {
    address LocalNet 172.16.0.0/24;
    address PrivateNet 192.168.10.0/24;
    address PublicNet 10.10.0.0/22;
}

root@NetDevOps-SRX01> show configuration security policies
from-zone trust to-zone trust {
    policy default-permit {
        match {
            source-address any;
            destination-address any;
            application any;
        }
        then {
            permit;
        }
    }
}
from-zone trust to-zone untrust {
    policy default-permit {
        match {
            source-address any;
            destination-address any;
            application any;
        }
        then {
            permit;
        }
    }
    policy Allow_Policy {
        match {
            source-address LocalNet;
            destination-address any;
            application any;
        }
        then {
            permit;
        }
    }
}
from-zone untrust to-zone trust {
    policy default-deny {
        match {
            source-address any;
            destination-address any;
            application any;
        }
        then {
            deny;
        }
    }
}

root@NetDevOps-SRX01> exit

root@NetDevOps-SRX01% exit
logout
Connection to 172.16.0.1 closed.
vagrant@NetDevOps-Student:~/JNPRAutomateDemo-Student/ansible$
```