docs/creatingapppolicies.md

Creating Application Policies

Now it is time to start using some of the more advanced firewall capabilities. The first set of policies we will create are application firewall or AppFW policies. This feature allows us to look into the data being sent over the connection.

Creating Application Policies with Ansible

Before we created basic firewall policies using Ansible. Now we will create application firewall policies. While the concept os the same there is an additional challenge. In a basic firewall policy you can add what amounts top just ports now you have to manage the applications that go over those ports. Typically you will want to apply many more specific applications that you want to block. You can also add other elements such as application groups. Because of this managing AppFW policies can be quite tedious. But as we will see

Reviewing the playbook

First let's take a look at the playbook that is used to accomplish this task.

Playbook Review

1. Define the name of the playbook - Configure AppFirewall 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
    - appfw_to_policy_info will be used to apply the AppFW policy to our stateful policy
    - appfw_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 AppFirewall policies
  hosts: mysrx
  connection: local
  gather_facts: no
  vars:
    junos_user: "root"
    junos_password: "Juniper"
    build_dir: "/tmp/"
    appfw_to_policy_info: [{"src_zone":"trust","dst_zone":"untrust","policy_name":"Allow_Policy
    appfw_policy_info: [{"rule_set":"ruleset1","rule_set_default_action":"permit","rules":[{"na

  tasks:
    - name: Build app firewall policies
      template: src=templates/appfw_policy.set.j2 dest={{build_dir}}/appfw_policy.set
      with_items: appfw_policy_info

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

    - name: Apply app firewall rules to policy
      template: src=templates/appfw_to_policy.set.j2 dest={{build_dir}}/appfw_to_policy.set
      with_items: appfw_to_policy_info

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

AppFW Policy Template

- It generates one "set" configuration line per loop
- Here we are generating three lines

```
{% for item in appfw_policy_info %}
    {% for i in item.rules %}
        {% for app in i.dynapps %}
set security application-firewall rule-sets {{ item.rule_set }} rule {{ i.name }} match dynamic
        {% endfor %}
set security application-firewall rule-sets {{ item.rule_set }} rule {{ i.name }} then {{ i.act
    {% endfor %}
set security application-firewall rule-sets {{ item.rule_set }} default-rule {{ item.rule_set_d
{% 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 application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 match dynamic-application junos
set security application-firewall rule-sets ruleset1 rule rule1 then deny
set security application-firewall rule-sets ruleset1 default-rule permit
```

Template to apply AppFW policy to a firewall policy

In this template we simply apply the AppFW policy to the stateful rule. We still use it as a loop in the event that we want to apply multiple policies at the same time.

```
{% for item in appfw_to_policy_info %}
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 then permit applicati
```

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 AppFirewall policies] >
 ---------------------------------------
       \   ^__^
        \  (oo)\_______
           (__)\       )\/\
               ||----w |
               ||     ||


 ___________________________________
< TASK: Build app firewall policies >
 -----------------------------------
       \   ^__^
        \  (oo)\_______
           (__)\       )\/\
               ||----w |
               ||     ||


changed: [172.16.0.1] => (item={'rules': [{'action': 'deny', 'dynapps': ['junos:GOOGLE', 'junos
 ___________________________________
< TASK: Apply app firewall policies >
 -----------------------------------
       \   ^__^
        \  (oo)\_______
           (__)\       )\/\
               ||----w |
               ||     ||


ok: [172.16.0.1]
 __________________________________________
< TASK: Apply app firewall rules to policy >
 ------------------------------------------
       \   ^__^
        \  (oo)\_______
           (__)\       )\/\
               ||----w |
               ||     ||


ok: [172.16.0.1] => (item={'appfw_rule_set': 'ruleset1', 'src_zone': 'trust', 'dst_zone': 'untr
 _______________________________
< TASK: Apply firewall policies >
 -------------------------------
       \   ^__^
        \  (oo)\_______
           (__)\       )\/\
               ||----w |
               ||     ||


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


172.16.0.1                 : ok=4    changed=1    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
root@NetDevOps-SRX01% cli

root@NetDevOps-SRX01> show security application-firewall rule-set all
Rule-set: ruleset1
    Rule: rule1
        Dynamic Applications: junos:GOOGLE, junos:GOOGLE-ACCOUNTS, junos:GOOGLE-ACCOUNTS-SSL, j
        junos:GOOGLE-ANALYTICS-TRACKING, junos:GOOGLE-APPENGINE, junos:GOOGLE-CACHE, junos:GOOG
        junos:GOOGLE-DOCS-DRAWING, junos:GOOGLE-DOCS-FORM, junos:GOOGLE-DOCS-PRESENTATION, juno
        junos:GOOGLE-DOCS-WORD-DOCUMENT, junos:GOOGLE-DRIVE, junos:GOOGLE-EARTH, junos:GOOGLE-G
        junos:GOOGLE-MOBILE-MAPS-APP, junos:GOOGLE-PICASA, junos:GOOGLE-PLUS, junos:GOOGLE-PLUS
        junos:GOOGLE-SAFEBROWSE-UPDATE, junos:GOOGLE-SKYMAP, junos:GOOGLE-STATIC, junos:GOOGLE-
        junos:GOOGLE-TRANSLATE, junos:GOOGLE-UPDATE, junos:GOOGLE-VIDEOS, junos:GOOGLE-WEBCHAT,
        SSL-Encryption: any
        Action:deny
        Number of sessions matched: 0
        Number of sessions redirected: 0
Default rule:permit
        Number of sessions matched: 0
        Number of sessions redirected: 0
Number of sessions with appid pending: 0

root@NetDevOps-SRX01> show configuration security application-firewall
rule-sets ruleset1 {
    rule rule1 {
        match {
            dynamic-application [ junos:GOOGLE junos:GOOGLE-ACCOUNTS junos:GOOGLE-ACCOUNTS-SSL
        }
        then {
            deny;
        }
    }
    default-rule {
        permit;
    }
}

root@NetDevOps-SRX01> show security policies from-zone trust to-zone untrust
From zone: trust, To zone: untrust
  Policy: default-permit, State: enabled, Index: 5, Scope Policy: 0, Sequence number: 1
    Source addresses: any
    Destination addresses: any
    Applications: any
    Action: permit
  Policy: Allow_Policy, State: enabled, Index: 7, Scope Policy: 0, Sequence number: 2
    Source addresses: LocalNet
    Destination addresses: PrivateNet
    Applications: any
    Action: permit, application services
  Application firewall:ruleset1

root@NetDevOps-SRX01> show configuration security policies 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 PrivateNet;
        application any;
    }
    then {
        permit {
            application-services {
                application-firewall {
                    rule-set ruleset1;
                }
            }
        }
    }
}

root@NetDevOps-SRX01>

root@NetDevOps-SRX01> exit

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

Testing

```
curl http://10.10.0.10:8080 -X GET -H "Host: google.com"
```