# Creating IPS Policies

Much like AppFW policies IDP policies can also be tasking to manager. In this section we will use Ansible to create our IDP policies as well.

# Creating IPS Policies with Ansible

In our third example of policy management policies we will now apply the same ideas to IPS. Much like AppFW policies IDP policies can use thousands of different objects to create the the policies. While the majority of customers use attack groups there are many that still manage individual attacks due to their complex security requirements.

## 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 IPS 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
- idp_policy_name will be used to apply the IDP policy to our stateful policy
- idp_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 IPS policies
  hosts: mysrx
  connection: local
  gather_facts: no
  vars:
    junos_user: "root"
    junos_password: "Juniper"
    build_dir: "/tmp/"
    idp_policy_name: "ips-policy1"
    idp_to_policy_info: [{"src_zone":"trust","dst_zone":"untrust","policy_n
    idp_policy_info: [{"policy_name":"ips-policy1","rules":[{"name":"rule1"

  tasks:
    - name: Build ips policies config template
      template: src=templates/idp_policy.set.j2 dest={{build_dir}}/idp_poli
      with_items: ips_policy_info

    - name: Apply ips policies
      junos_install_config: host={{ inventory_hostname }} user={{ junos_use

    - name: Build ips policy apply template
      template: src=templates/idp_policy_activate.set.j2 dest={{build_dir}}

    - name: Activate ips policy
      junos_install_config: host={{ inventory_hostname }} user={{ junos_use
```

**IDP Policy Template**

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

- Due to the potential complexities of an IPS policy this is our most complex template yet
- Each section will only generate the appropriate configuration if the elements exist
    - For example if rule.predef_attacks is defined then it will be looped through
    - If not then that part of the template will be ignored
    - If it does exist then it will be executed and the configuration will be generated

```
{% for item in idp_policy_info %}
set security idp idp-policy {{ item.policy_name }}
    {% for rule in item.rules %}
        {% for i in rule.src_ips %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
        {% endfor %}
        {% for i in rule.dst_ips %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
        {% endfor %}
        {% for i in rule.apps %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
        {% endfor %}
        {% if rule.predef_attacks is defined %}
            {% for x in rule.predef_attacks %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
            {% endfor %}
        {% endif %}
        {% if rule.predef_attack_groups is defined %}
            {% for x in rule.predef_attack_groups %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
            {% endfor %}
        {% endif %}
        {% if rule.custom_attacks is defined %}
            {% for x in item.custom_attacks %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
            {% endfor %}
        {% endif %}
        {% if rule.custom_attack_groups is defined %}
            {% for x in item.custom_attack_groups %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
            {% endfor %}
        {% endif %}
        {% if rule.dyn_attack_groups is defined %}
            {% for x in item.custom_attacks %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
            {% endfor %}
        {% endif %}
        {% if rule.notification.log is defined and rule.notification.alert
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
        {% endif %}
```

```
        {% if rule.notification.log is defined %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
        {% endif %}
        {% if rule.severity is defined %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
        {% endif %}
set security idp idp-policy {{ item.policy_name }} rulebase-ips rule {{ rul
    {% endfor %}
{% 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 idp idp-policy ips-policy1
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 match from-
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 match from-
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 match from-
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 match attac
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 match attac
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 then notifi
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 then notifi
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 then severi
set security idp idp-policy ips-policy1 rulebase-ips rule rule1 then action
```

**Template to apply IPS policy to a firewall policy**

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

```
set security idp active-policy {{ idp_policy_name }}
{% for item in idp_to_policy_info %}
set security policies from-zone {{ item.src_zone }} to-zone {{ item.dst_zon
{% 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 idp active-policy ips-policy1
set security policies from-zone trust to-zone untrust policy Allow_Policy t
```

# 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-Class/ansible$ ansible-playboo
```

### Playbook Run Example

Once run the output should look like the following

```
vagrant@NetDevOps-Student:~/JNPRAutomateDemo-Class/ansible$ ansible-playboo
 _______________________________
< PLAY [Configure IPS policies] >
 -------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


 ________________________________________
< TASK: Build ips policies config template >
 ----------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||     ||
```

```
                ||----w |
                ||     ||


ok: [172.16.0.1] => (item=ips_policy_info)

 ___________________________
< TASK: Apply ips policies >
 ---------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


changed: [172.16.0.1]

 _____________________________________
< TASK: Build ips policy apply template >
 -------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


changed: [172.16.0.1] => (item={'src_zone': 'trust', 'dst_zone': 'untrust',

 ___________________________
< TASK: Activate ips policy >
 ---------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


changed: [172.16.0.1]

 ____________
< PLAY RECAP >
 ------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||


172.16.0.1                 : ok=4    changed=3    unreachable=0    failed=0
```

```
vagrant@NetDevOps-Student:~/JNPRAutomateDemo-Class/ansible$
```

### Validating the playbook run

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

```
--- JUNOS 12.1X47-D20.7 built 2015-03-03 21:53:50 UTC
root@NetDevOps-SRX01% cli
root@NetDevOps-SRX01> show security idp status
State of IDP: Default,  Up since: 2015-03-25 22:39:58 UTC (05:30:12 ago)

Packets/second: 0               Peak: 0 @ 2015-03-26 04:09:11 UTC
KBits/second  : 0               Peak: 0 @ 2015-03-26 04:09:11 UTC
Latency (microseconds): [min: 0] [max: 0] [avg: 0]

Packet Statistics:
 [ICMP: 0] [TCP: 0] [UDP: 0] [Other: 0]

Flow Statistics:
  ICMP: [Current: 0] [Max: 0 @ 2015-03-26 04:09:11 UTC]
  TCP: [Current: 0] [Max: 0 @ 2015-03-26 04:09:11 UTC]
  UDP: [Current: 0] [Max: 0 @ 2015-03-26 04:09:11 UTC]
  Other: [Current: 0] [Max: 0 @ 2015-03-26 04:09:11 UTC]

Session Statistics:
 [ICMP: 0] [TCP: 0] [UDP: 0] [Other: 0]
  Policy Name : ips-policy1
  Running Detector Version : 12.6.130140822

root@NetDevOps-SRX01> show security idp policies
ID    Name                    Sessions    Memory      Detector
 3     ips-policy1             0           2720166     12.6.130140822

root@NetDevOps-SRX01> show configuration security idp
idp-policy ips-policy1 {
    rulebase-ips {
        rule rule1 {
            match {
                from-zone trust;
                source-address any;
                to-zone untrust;
                destination-address any;
                application default;
                attacks {
                    predefined-attack-groups [ "APP - Critical" "APP - Majo
```

```
                }
            }
            then {
                action {
                    drop-connection;
                }
                notification {
                    log-attacks {
                        alert;
                    }
                }
                severity critical;
            }
        }
    }
}
active-policy ips-policy1;

root@NetDevOps-SRX01> show configuration security policies from-zone trust
match {
    source-address LocalNet;
    destination-address any;
    application any;
}
then {
    permit {
        application-services {
            idp;
            application-firewall {
                rule-set ruleset1;
            }
        }
    }
}

root@NetDevOps-SRX01>
```