Las Vegas + Digital | June 12-16, 2022

Managing Cisco Firepower Threat Défense using REST API HOLDEV-1005

Speakers:
Madhuri Dewangan – Security Consulting Engineer

Table of Contents

Learning Objectives

Upon completion of this lab, you will be able to:

- Understand working of Rest API.
- Understand how Cisco FTD Rest API works
- Understand automation requirement for Cisco FTD

Scenario

In this lab activity, you will learn how to write the reusable templates for Infrastructure as Code based languages (Ansible) for Managing the Cisco Secure Firewall. This lab will be based on 3 parts, divided as follows:

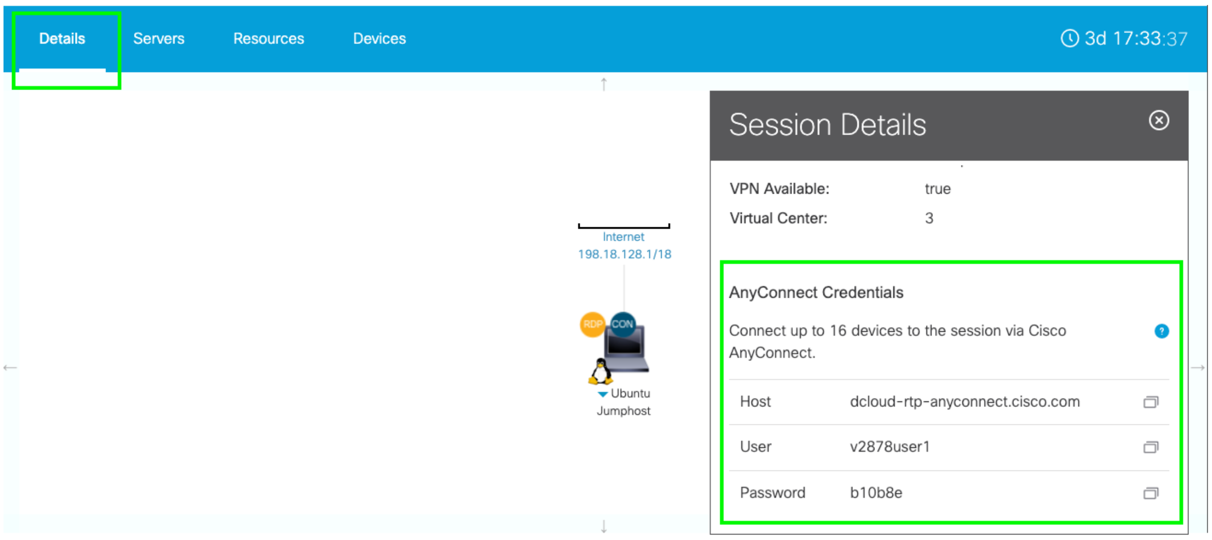
1. Hands On Experience for Postman and Cisco FTD
2. Hands On Experience for Ansible and Cisco FTD
3. Hands On Experience for Python and Cisco FTD

These tasks are designed to understand benefit of Automation whilst doing the same task manually.

Pre-Requisites:

In this lab activity, make sure you have the required pre-reqs in place:

1. Login to the VPN to access the Jump host. Make sure you have AnyConnect present in the system.
2. Please login to jump host to access the test environment. You can find your login credentials from the dcloud portal.
   To login to your jump host follow below procedure.



Session Details

VPN Available: true
Virtual Center: 3

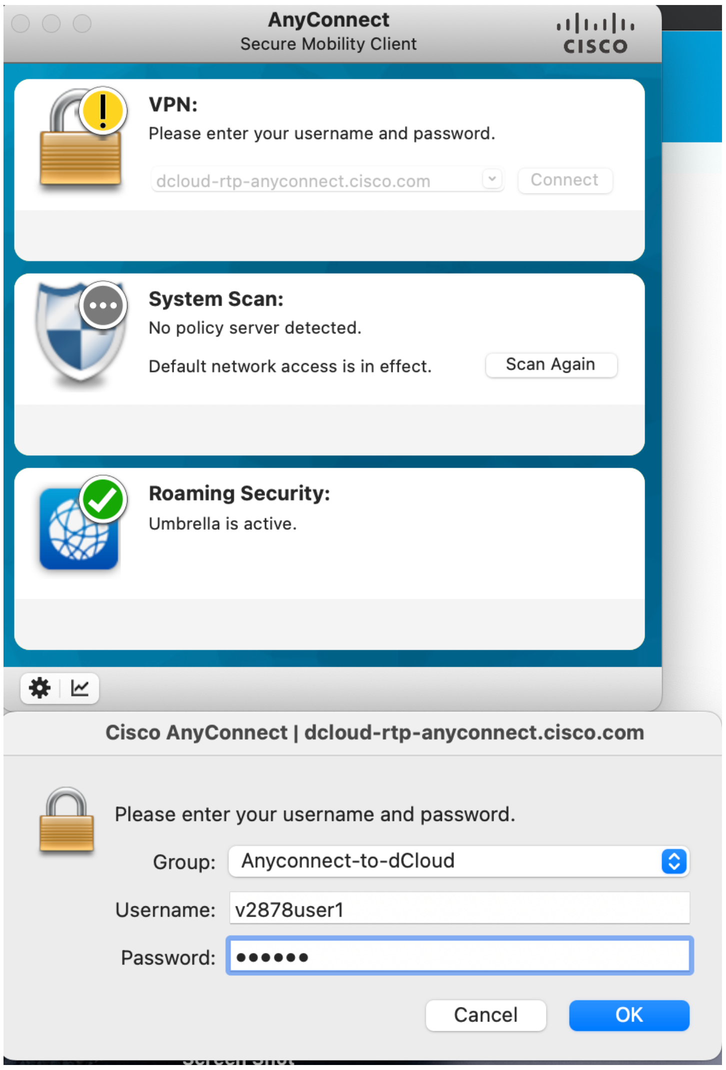
AnyConnect Credentials

Connect up to 16 devices to the session via Cisco AnyConnect.

Host	dcloud-rtp-anyconnect.cisco.com
User	v2878user1
Password	b10b8e

Figure 1 AnyConnect Details

a. User the provided details to connect to the network.



VPN:
Please enter your username and password.
dcloud-rtp-anyconnect.cisco.com

System Scan:
No policy server detected.
Default network access is in effect.

Roaming Security:
Umbrella is active.

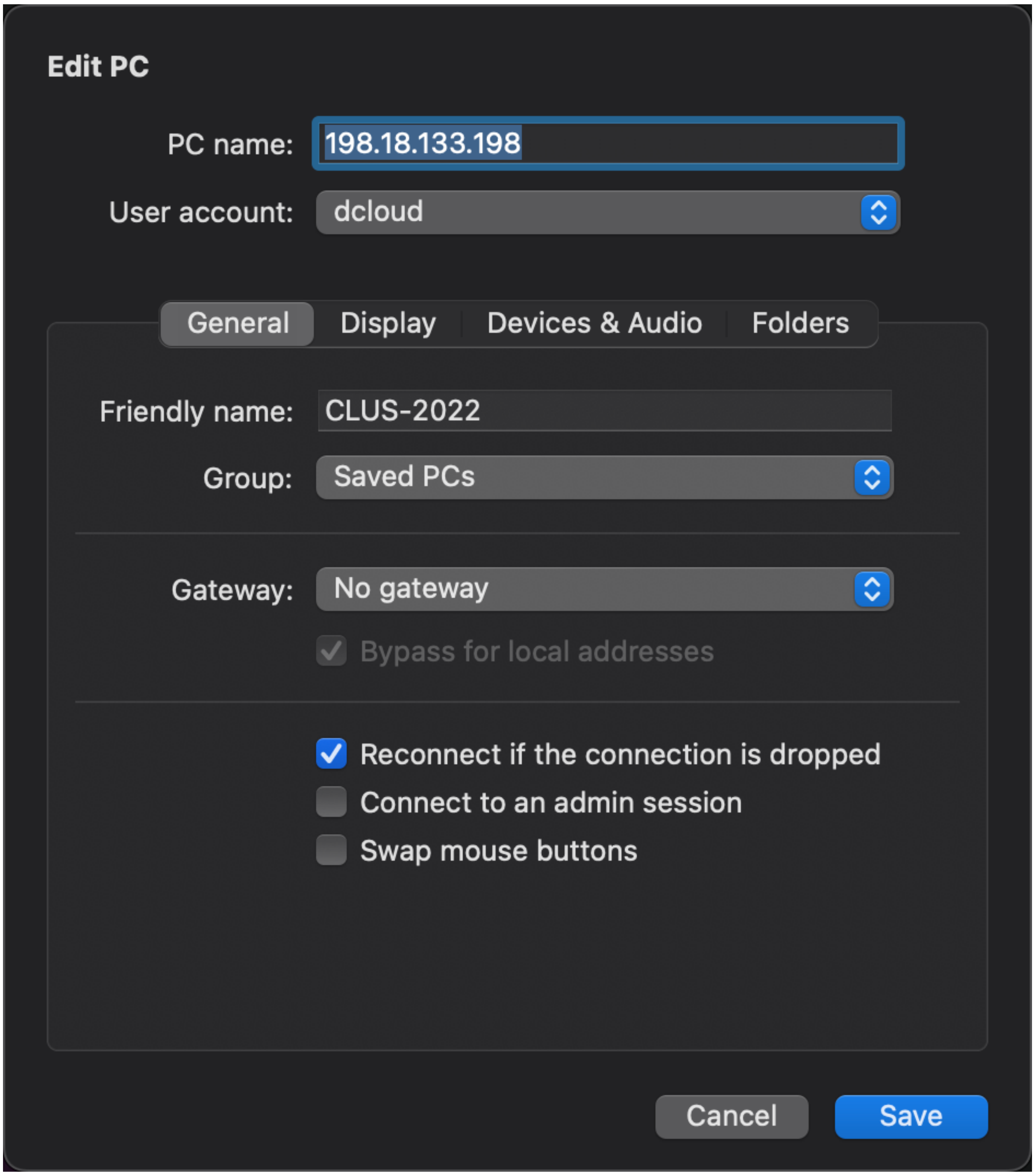
Cisco AnyConnect | dcloud-rtp-anyconnect.cisco.com

Please enter your username and password.

Group: Anyconnect-to-dCloud
Username: v2878user1
Password: ●●●●●●

Figure 2 Connect to VPN

b. **Using any RDP Client connect to the host ip mentioned in your session.**


Edit PC

PC name: 198.18.133.198

User account: dcloud

Friendly name: CLUS-2022

Group: Saved PCs

Gateway: No gateway

✓ Bypass for local addresses

✓ Reconnect if the connection is dropped

Connect to an admin session

Swap mouse buttons

Figure 3 RDP to Ubuntu Jump host

c. **Connect to Jump host**

3. **Verify following is present in the jumhost.**

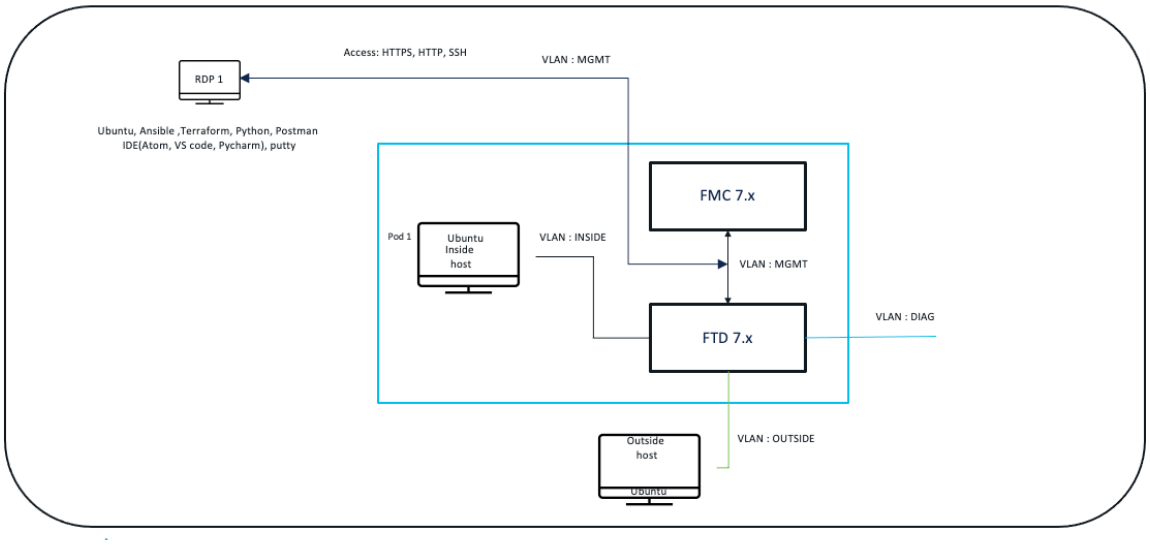
a. **PyCharm (IDE)**

b. **Postman**

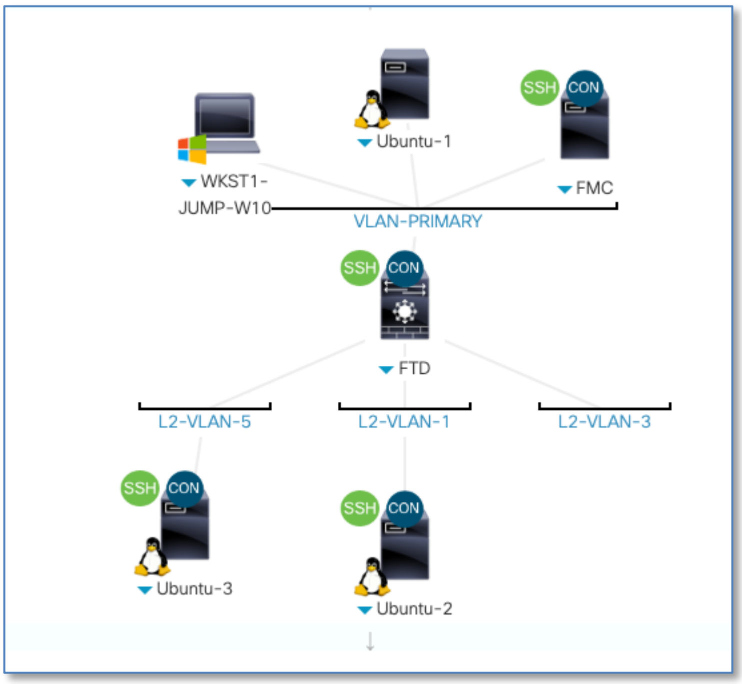
c. **Ansible**

4. Internet connectivity

Network Diagram



Lab Topology



Access Details:

S.No	Device Name	IP	Username	Password
1	FMC	198.18.133.30	admin	C1sco12345
2	FTD	198.18.133.25	admin	C1sco12345
3	Jumphost (MGMT)	198.18.133.45	cisco	C1sco12345
4	Inside Host	198.18.13.20	dcloud	C1sco12345
5	Outside Host	198.18.14.20	dcloud	C1sco12345

Task 1: Hands On Experience for Postman and Cisco FTD

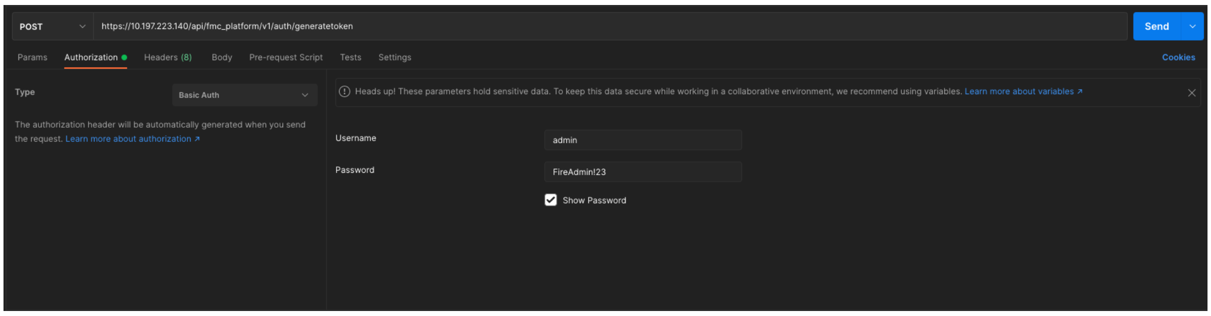
In this task, you are going to Interact with Cisco FTD's Rest API using Postman.

To interact with Cisco FTD's REST API, a user needs to authenticate itself, in order to access various use cases supported by Cisco FTD.

Task 1A: Authenticate User, Get Access Token

To authenticate the user and get the authentication/access token for the session, user needs to send a POST call to Cisco FMC. Access Token is then returned in the header of the response.

Note: https://www.cisco.com/c/en/us/support/docs/security/firepower-management-center/215918-how-to-generate-authentication-token-for.html


POST https://10.197.223.140/api/fmc_platform/v1/auth/generatetoken

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings | Cookies

Type: Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables ↗

Username	admin
Password	FireAdmin!23

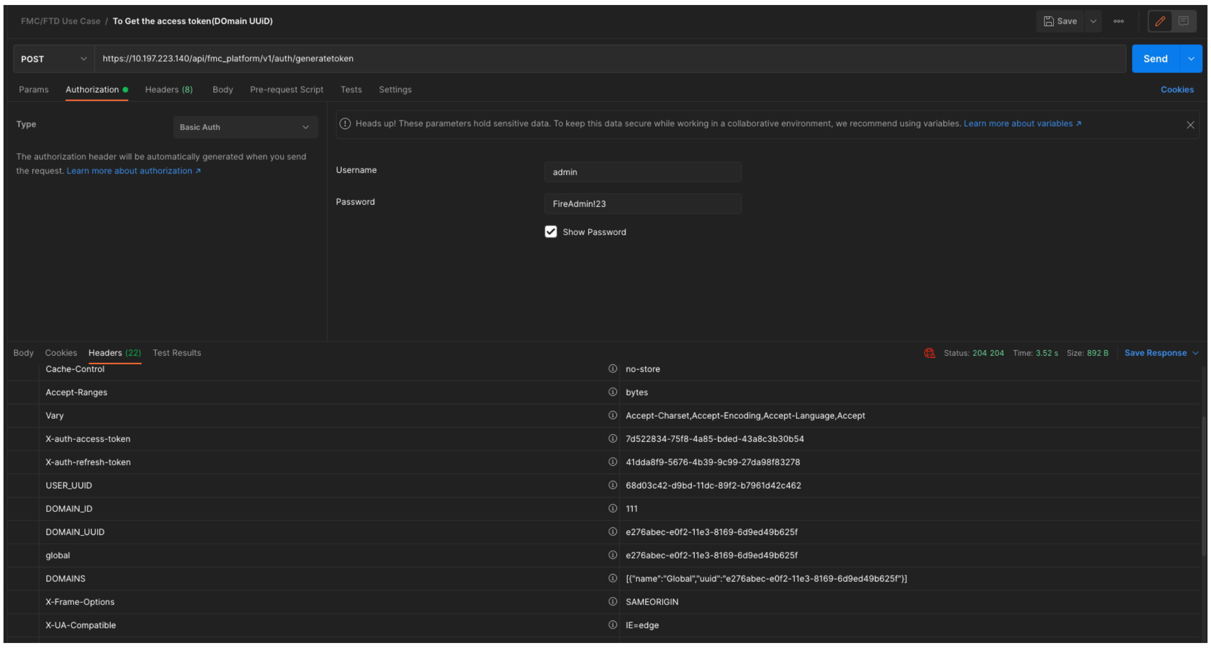
[✓ Show Password]

Action to Take:

1. **Open postman in your jump host.**
2. **Change the HTTP Method to POST and use the URL Error! Hyperlink reference not valid. in the URL section. Replace the {{FMC_IP}} with your own FMC's IP given in the session.**
3. **Under Authorization Tab, select Basic Auth and Provide Username and Password as shown in the snapshot**

Result:

1. **If the authentication is successful , you will get HTTP response of 204 and under header section, you will see the access token under field "x-auth-access-token" and "DOMAIN_UUID"as shown below.**


FMC/FTD Use Case / To Get the access token(DOmain UUiD)

POST https://10.197.223.140/api/fmc_platform/v1/auth/generatetoken

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings | Cookies

Type: Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables ↗

Username	admin
Password	FireAdmin!23

[✓ Show Password]

Body | Cookies | Headers (22) | Test Results — Status: 204 204 Time: 3.52 s Size: 892 B Save Response

Cache-Control	no-store
Accept-Ranges	bytes
Vary	Accept-Charset,Accept-Encoding,Accept-Language,Accept
X-auth-access-token	7d522834-75f8-4a85-bded-43a8c3b30b54
X-auth-refresh-token	41dda8f9-5676-4b39-9c99-27da98f83278
USER_UUID	68d03c42-d9bd-11dc-89f2-b7961d42c462
DOMAIN_ID	111
DOMAIN_UUID	e276abec-e0f2-11e3-8169-6d9ed49b625f
global	e276abec-e0f2-11e3-8169-6d9ed49b625f
DOMAINS	[{"name":"Global","uuid":"e276abec-e0f2-11e3-8169-6d9ed49b625f"}]
X-Frame-Options	SAMEORIGIN
X-UA-Compatible	IE=edge

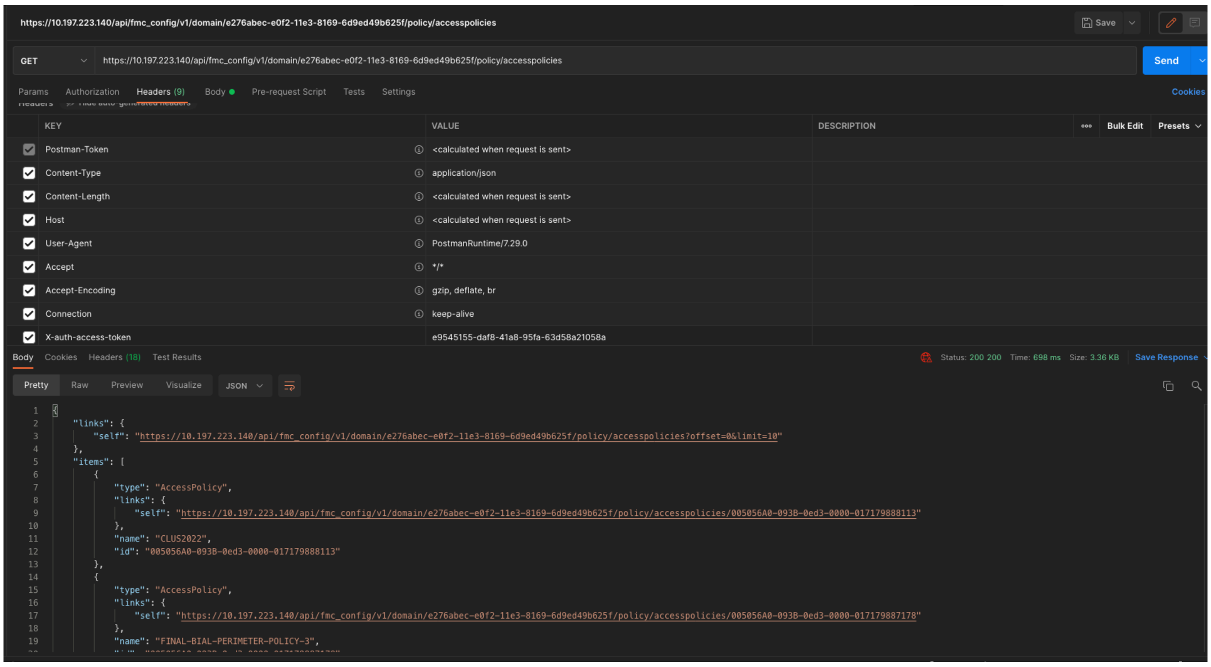
Task 1B: Get the list of Access Policy Configured in this FMC.

Action to Take:

1. **Open postman in your jump host.**
2. **Change the HTTP Method to GET, and use the URL Error! Hyperlink reference not valid.in the URL section. Replace the {{FMC_IP}} with your own FMC's IP , {{DomainUUID}} with the value as obtained in Task1A.**
3. **Since each HTTP call needs to be authenticated, hence use the access token obtained in Task 1A. Navigate to Headers Tab and add a new field "x-auth-access-token" and use the value as obtained in Task1A.**
4. **Click Send**

Result

1. **Ensure you get a response of HTTP 200**
2. **You will receive the Policy Configured in Cisco FMC in JSON format, as shown in snapshot.**


https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies

GET https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies

	KEY	VALUE	DESCRIPTION
✓	Postman-Token	<calculated when request is sent>	
✓	Content-Type	application/json	
✓	Content-Length	<calculated when request is sent>	
✓	Host	<calculated when request is sent>	
✓	User-Agent	PostmanRuntime/7.29.0	
✓	Accept	*/*	
✓	Accept-Encoding	gzip, deflate, br	
✓	Connection	keep-alive	
✓	X-auth-access-token	e9545155-daf8-41a8-95fa-63d58a21058a	

Status: 200 200 Time: 698 ms Size: 3.36 KB

```
{
    "links": {
        "self": "https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies?offset=0&limit=10"
    },
    "items": [
        {
            "type": "AccessPolicy",
            "links": {
                "self": "https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies/005056A0-093B-0ed3-0000-017179888113"
            },
            "name": "CLUS2022",
            "id": "005056A0-093B-0ed3-0000-017179888113"
        },
        {
            "type": "AccessPolicy",
            "links": {
                "self": "https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies/005056A0-093B-0ed3-0000-017179887178"
            },
            "name": "FINAL-BIAL-PERIMETER-POLICY-3",
```

Task 1C: Create an Access Policy in this FMC.

Action to Take:

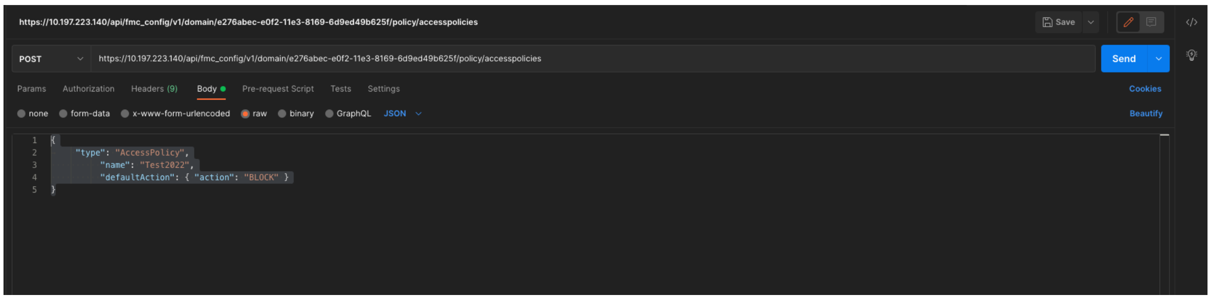
1. **Open postman in your jumphost.**
2. **Change the HTTP Method to POST, and use the URL**

Error! Hyperlink reference not valid.in the URL section. Replace the {{FMC_IP}} with your own FMC's IP , {{DomainUUID}} with the value as obtained in Task1A.

3. **Under Body , provide following json to create this in the FMC**

```
{
    "type": "AccessPolicy",
         "name": "Test2022",
         "defaultAction": { "action": "BLOCK" }
}
```

4. **Since each HTTP call needs to be authenticated, hence use the access token obtained in Task 1A. Navigate to Headers Tab and add a new field "x-auth-access-token" and use the value as obtained in Task1A.**
5. **Click Send**



https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies

POST https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies

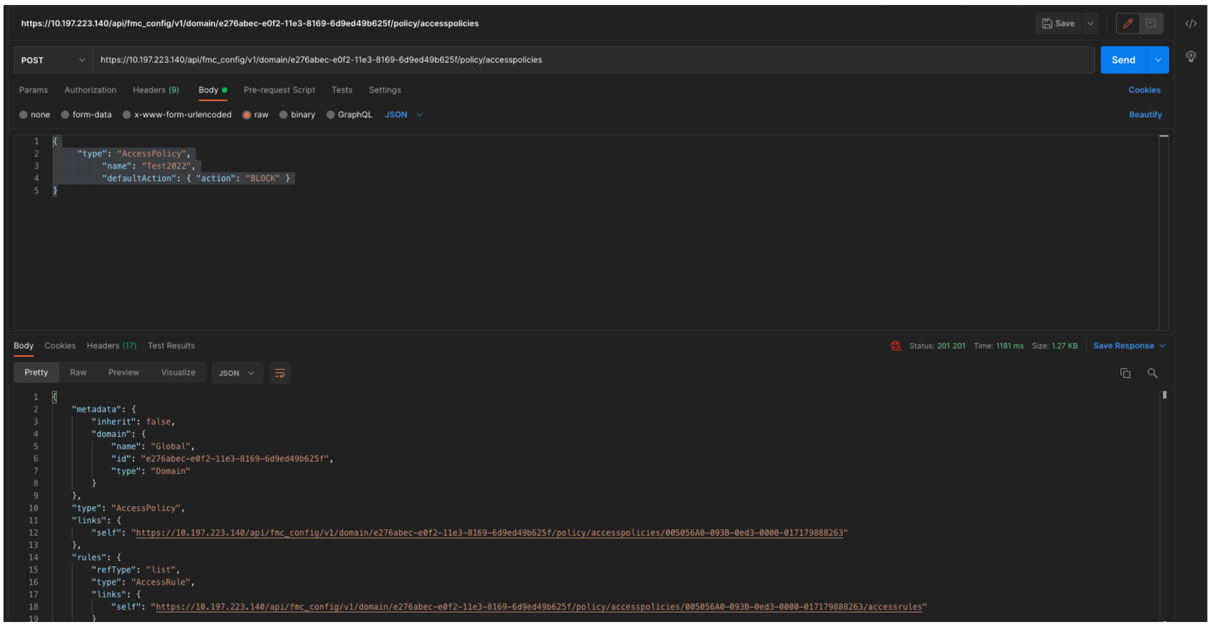
Params | Authorization | Headers (9) | Body | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL | JSON

```
{
    "type": "AccessPolicy",
        "name": "Test2022",
        "defaultAction": { "action": "BLOCK" }
}
```

Result

3. **Ensure you get a response of HTTP 201**
4. **You will receive the Policy Configured in Cisco FMC in JSON format, as shown in snapshot.**



https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies

POST https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies

Params | Authorization | Headers (9) | Body | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL | JSON

```
{
    "type": "AccessPolicy",
        "name": "Test2022",
        "defaultAction": { "action": "BLOCK" }
}
```

Body | Cookies | Headers (17) | Test Results — Status: 201 201 Time: 1181 ms Size: 1.27 KB

Pretty | Raw | Preview | Visualize | JSON

```
{
    "metadata": {
        "inherit": false,
        "domain": {
            "name": "Global",
            "id": "e276abec-e0f2-11e3-8169-6d9ed49b625f",
            "type": "Domain"
        }
    },
    "type": "AccessPolicy",
    "links": {
        "self": "https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies/005056A0-093B-0ed3-0000-017179888263"
    },
    "rules": {
        "refType": "list",
        "type": "AccessRule",
        "links": {
            "self": "https://10.197.223.140/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies/005056A0-093B-0ed3-0000-017179888263/accessrules"
        }
```

----End of Task 1----

Task 2: Hands On Experience for Ansible and Cisco FTD

In this task, you are going to Interact with Cisco FTD's Rest API using Ansible.

Ansible will be running in the Jump host which should be having access over HTTPS to Cisco FMC.

Use cases where Ansible can be leveraged

1. **Physical Device initial configuration of the Device**
2. **Non-Bulk Configuration append**
3. **Fetching Information from the device**
4. **Zero Touch Configuration(Initial) for Cloud Based Cisco FTD**

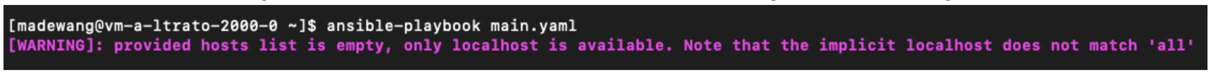
Task 2A: Register Newly Build Cisco FTD to Cisco FMC

To register the newly build Cisco FTD to Targeted FMC we need following pre-requisites to be met for this task to successfully accomplish

1. **Cisco FTD is ready with day-0 configuration**
2. **Cisco FTD is configured with its manager[This can be automated, for lab we are relying on manual configuration]**
3. **Network Reachability between Cisco FMC, Cisco FTD and Jump host where Ansible is running.**

Actions to take:
Note: For the sake of the Lab Execution time, I have already prepared Ansible playbook, which is available at the GitHub, you can clone the playbook and execute. During execution of the playbook this will create a default policy to register the FTD to FMC.

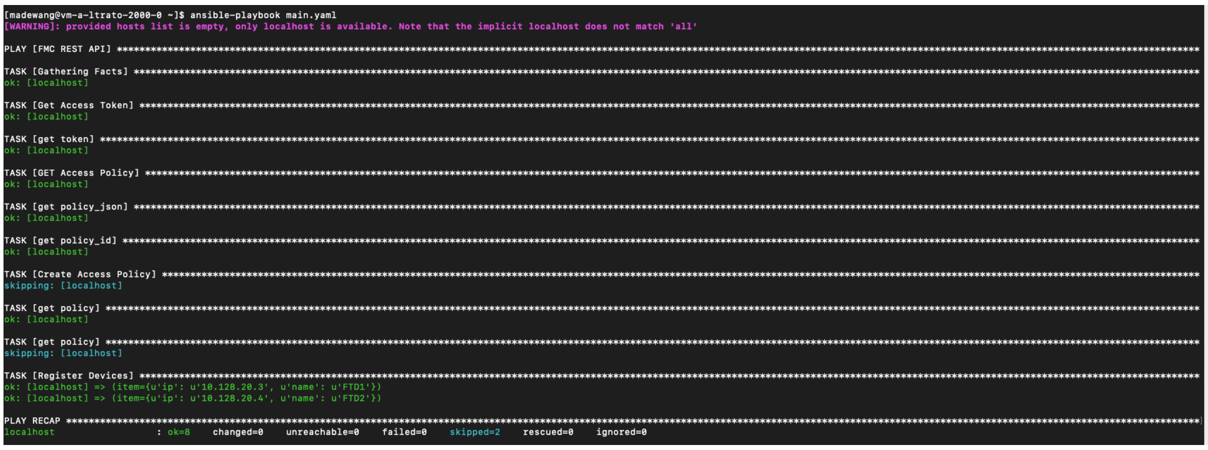
1. **Git clone {{URL for git repository}}**
2. **Open the playbook using vi and edit the Cisco FMC IP as provided in the session.**
3. **Save the playbook.**
4. **Run the playbook to for the setup. "ansible-playbook main.yaml"**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook main.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
```

Result:

1. **All the FTD mentioned in the vars.yml file will get registered to the FMC. Expected outcome is as follows in the snapshot.**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook main.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [FMC REST API] ****************************************************************************

TASK [Gathering Facts] *************************************************************************
ok: [localhost]

TASK [Get Access Token] ************************************************************************
ok: [localhost]

TASK [get token] *******************************************************************************
ok: [localhost]

TASK [GET Access Policy] ***********************************************************************
ok: [localhost]

TASK [get policy_json] *************************************************************************
ok: [localhost]

TASK [get policy_id] ***************************************************************************
ok: [localhost]

TASK [Create Access Policy] ********************************************************************
skipping: [localhost]

TASK [get policy] ******************************************************************************
ok: [localhost]

TASK [get policy] ******************************************************************************
skipping: [localhost]

TASK [Register Devices] ************************************************************************
ok: [localhost] => (item={u'ip': u'10.128.20.3', u'name': u'FTD1'})
ok: [localhost] => (item={u'ip': u'10.128.20.4', u'name': u'FTD2'})

PLAY RECAP *************************************************************************************
localhost                  : ok=8    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

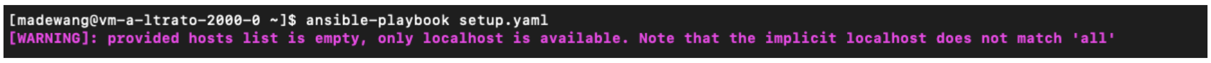
Task 2B: Configure each interface with the name and security zone

To configure each interface in all the FTDs here are my assumptions:

- **All the FTD's interface GigabitEthernet 0/0 will be connected to Internal LAN and configured as Inside zone**
- **All the FTD's interface GigabitEthernet 0/1 will be connected to Outside zone and configured as Outside zone**
- **Cisco FTD is already registered to its manager.**

Actions to take:
Note: For the sake of the Lab Execution time, I have already prepared Ansible playbook, which is available at the GitHub, you can clone the playbook and execute. During execution of the playbook this will create a default policy to register the FTD to FMC.

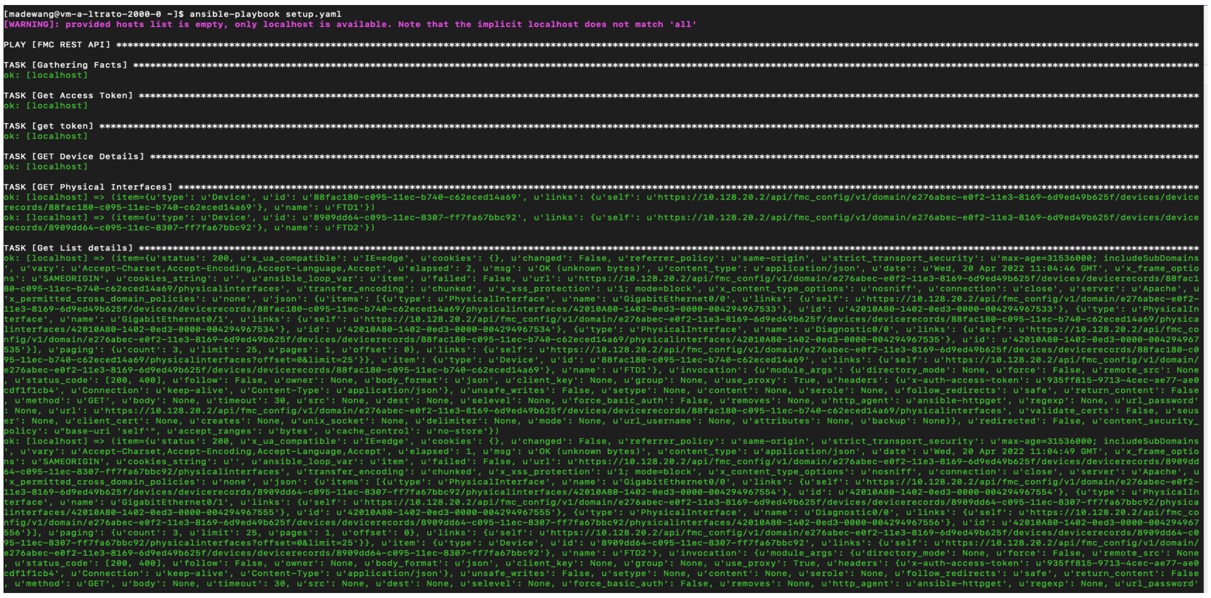
1. **Git clone {{URL for git repository}}**
2. **Open the playbook using vi and edit the Cisco FMC IP as provided in the session.**
3. **Save the playbook.**
4. **Run the playbook to for the setup. "ansible-playbook setup_device.yaml"**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook setup.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
```

Result:

1. **All the interfaces across all the FTD mentioned in the vars.yml file will be configured as per the assumption, and all the changes are deployed to the box as well.**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook setup.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [FMC REST API] ***********************************************************************************************************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************************************************************************************************************
ok: [localhost]

TASK [Get Access Token] *******************************************************************************************************************************************************************************************
ok: [localhost]

TASK [get token] **************************************************************************************************************************************************************************************************
ok: [localhost]

TASK [GET Device Details] *****************************************************************************************************************************************************************************************
ok: [localhost]

TASK [GET Physical Interfaces] ************************************************************************************************************************************************************************************
ok: [localhost] => (item={u'type': u'Device', u'id': u'88fac180-c095-11ec-b740-c62eced14a69', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/device
records/88fac180-c095-11ec-b740-c62eced14a69'}, u'name': u'FTD1'})
ok: [localhost] => (item={u'type': u'Device', u'id': u'8909dd64-c095-11ec-8307-ff7fa67bbc92', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/device
records/8909dd64-c095-11ec-8307-ff7fa67bbc92'}, u'name': u'FTD2'})

TASK [Get List details] *******************************************************************************************************************************************************************************************
ok: [localhost] => (item={u'status': 200, u'x_ua_compatible': u'IE=edge', u'cookies': {}, u'changed': False, u'referrer_policy': u'same-origin', u'strict_transport_security': u'max-age=31536000; includeSubDomains
', u'vary': u'Accept-Charset,Accept-Encoding,Accept-Language,Accept', u'elapsed': 2, u'msg': u'OK (unknown bytes)', u'content_type': u'application/json', u'date': u'Wed, 20 Apr 2022 11:04:46 GMT', u'x_frame_optio
ns': u'SAMEORIGIN', u'cookies_string': u'', u'ansible_loop_var': u'item', u'failed': False, u'url': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/88fac1
80-c095-11ec-b740-c62eced14a69/physicalinterfaces', u'transfer_encoding': u'chunked', u'x_xss_protection': u'1; mode=block', u'x_content_type_options': u'nosniff', u'connection': u'close', u'server': u'Apache', u
'x_permitted_cross_domain_policies': u'none', u'json': {u'items': [{u'type': u'PhysicalInterface', u'name': u'GigabitEthernet0/0', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-
11e3-8169-6d9ed49b625f/devices/devicerecords/88fac180-c095-11ec-b740-c62eced14a69/physicalinterfaces/42010A80-1402-0ed3-0000-004294967533'}, u'id': u'42010A80-1402-0ed3-0000-004294967533'}, {u'type': u'PhysicalIn
terface', u'name': u'GigabitEthernet0/1', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/88fac180-c095-11ec-b740-c62eced14a69/physica
linterfaces/42010A80-1402-0ed3-0000-004294967534'}, u'id': u'42010A80-1402-0ed3-0000-004294967534'}, {u'type': u'PhysicalInterface', u'name': u'Diagnostic0/0', u'links': {u'self': u'https://10.128.20.2/api/fmc_co
nfig/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/88fac180-c095-11ec-b740-c62eced14a69/physicalinterfaces/42010A80-1402-0ed3-0000-004294967535'}, u'id': u'42010A80-1402-0ed3-0000-004294967
535'}], u'paging': {u'count': 3, u'limit': 25, u'pages': 1, u'offset': 0}, u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/88fac180-c0
95-11ec-b740-c62eced14a69/physicalinterfaces?offset=0&limit=25'}}, u'item': {u'type': u'Device', u'id': u'88fac180-c095-11ec-b740-c62eced14a69', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/
e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/88fac180-c095-11ec-b740-c62eced14a69'}, u'name': u'FTD1'}, u'invocation': {u'module_args': {u'directory_mode': None, u'force': False, u'remote_src': None
, u'status_code': [200, 400], u'follow': False, u'owner': None, u'body_format': u'json', u'client_key': None, u'group': None, u'use_proxy': True, u'headers': {u'x-auth-access-token': u'935ff815-9713-4cec-ae77-ae0
cdf1f1cb4', u'Connection': u'keep-alive', u'Content-Type': u'application/json'}, u'unsafe_writes': False, u'setype': None, u'content': None, u'serole': None, u'follow_redirects': u'safe', u'return_content': False
, u'method': u'GET', u'body': None, u'timeout': 30, u'src': None, u'dest': None, u'selevel': None, u'force_basic_auth': False, u'removes': None, u'http_agent': u'ansible-httpget', u'regexp': None, u'url_password'
: None, u'url': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/88fac180-c095-11ec-b740-c62eced14a69/physicalinterfaces', u'validate_certs': False, u'seus
er': None, u'client_cert': None, u'creates': None, u'unix_socket': None, u'delimiter': None, u'mode': None, u'url_username': None, u'attributes': None, u'backup': None}}, u'redirected': False, u'content_security_
policy': u"base-uri 'self'", u'accept_ranges': u'bytes', u'cache_control': u'no-store'})
ok: [localhost] => (item={u'status': 200, u'x_ua_compatible': u'IE=edge', u'cookies': {}, u'changed': False, u'referrer_policy': u'same-origin', u'strict_transport_security': u'max-age=31536000; includeSubDomains
', u'vary': u'Accept-Charset,Accept-Encoding,Accept-Language,Accept', u'elapsed': 1, u'msg': u'OK (unknown bytes)', u'content_type': u'application/json', u'date': u'Wed, 20 Apr 2022 11:04:49 GMT', u'x_frame_optio
ns': u'SAMEORIGIN', u'cookies_string': u'', u'ansible_loop_var': u'item', u'failed': False, u'url': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/8909dd
64-c095-11ec-8307-ff7fa67bbc92/physicalinterfaces', u'transfer_encoding': u'chunked', u'x_xss_protection': u'1; mode=block', u'x_content_type_options': u'nosniff', u'connection': u'close', u'server': u'Apache', u
'x_permitted_cross_domain_policies': u'none', u'json': {u'items': [{u'type': u'PhysicalInterface', u'name': u'GigabitEthernet0/0', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-
11e3-8169-6d9ed49b625f/devices/devicerecords/8909dd64-c095-11ec-8307-ff7fa67bbc92/physicalinterfaces/42010A80-1402-0ed3-0000-004294967554'}, u'id': u'42010A80-1402-0ed3-0000-004294967554'}, {u'type': u'PhysicalIn
terface', u'name': u'GigabitEthernet0/1', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/8909dd64-c095-11ec-8307-ff7fa67bbc92/physica
linterfaces/42010A80-1402-0ed3-0000-004294967555'}, u'id': u'42010A80-1402-0ed3-0000-004294967555'}, {u'type': u'PhysicalInterface', u'name': u'Diagnostic0/0', u'links': {u'self': u'https://10.128.20.2/api/fmc_co
nfig/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/8909dd64-c095-11ec-8307-ff7fa67bbc92/physicalinterfaces/42010A80-1402-0ed3-0000-004294967556'}, u'id': u'42010A80-1402-0ed3-0000-004294967
556'}], u'paging': {u'count': 3, u'limit': 25, u'pages': 1, u'offset': 0}, u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/8909dd64-c0
95-11ec-8307-ff7fa67bbc92/physicalinterfaces?offset=0&limit=25'}}, u'item': {u'type': u'Device', u'id': u'8909dd64-c095-11ec-8307-ff7fa67bbc92', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/
e276abec-e0f2-11e3-8169-6d9ed49b625f/devices/devicerecords/8909dd64-c095-11ec-8307-ff7fa67bbc92'}, u'name': u'FTD2'}, u'invocation': {u'module_args': {u'directory_mode': None, u'force': False, u'remote_src': None
, u'status_code': [200, 400], u'follow': False, u'owner': None, u'body_format': u'json', u'client_key': None, u'group': None, u'use_proxy': True, u'headers': {u'x-auth-access-token': u'935ff815-9713-4cec-ae77-ae0
cdf1f1cb4', u'Connection': u'keep-alive', u'Content-Type': u'application/json'}, u'unsafe_writes': False, u'setype': None, u'content': None, u'serole': None, u'follow_redirects': u'safe', u'return_content': False
, u'method': u'GET', u'body': None, u'timeout': 30, u'src': None, u'dest': None, u'selevel': None, u'force_basic_auth': False, u'removes': None, u'http_agent': u'ansible-httpget', u'regexp': None, u'url_password'
```

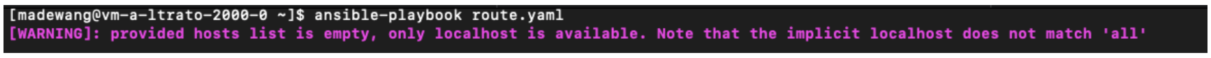
Task 2C: Configure Basic Static Route on each FTD.

To configure each interface in all the FTDs here are my assumptions:

- **All the FTD's interface GigabitEthernet 0/0 will be connected to Internal LAN and configured as Inside zone**
- **All the FTD's interface GigabitEthernet 0/1 will be connected to Outside zone and configured as Outside zone**
- **Cisco FTD is already registered to its manager.**

Actions to take:
Note: For the sake of the Lab Execution time, I have already prepared Ansible playbook, which is available at the GitHub, you can clone the playbook and execute. During execution of the playbook this will create a default policy to register the FTD to FMC.

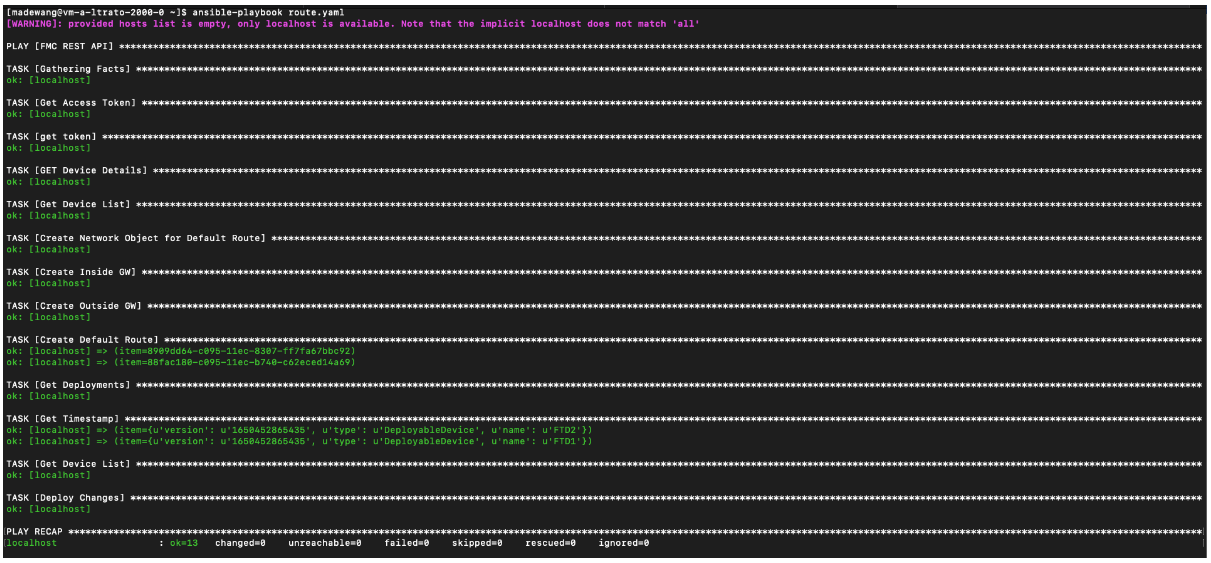
1. **Git clone {{URL for git repository}}**
2. **Open the playbook using vi and edit the Cisco FMC IP as provided in the session.**
3. **Save the playbook.**
4. **Run the playbook to for the setup. "ansible-playbook route.yaml"**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook route.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
```

Result:

1. **Default route configured on all the Cisco FTDs registered to Cisco FMC.**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook route.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [FMC REST API] ************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************
ok: [localhost]

TASK [Get Access Token] ********************************************************************************************************************
ok: [localhost]

TASK [get token] ***************************************************************************************************************************
ok: [localhost]

TASK [GET Device Details] ******************************************************************************************************************
ok: [localhost]

TASK [Get Device List] *********************************************************************************************************************
ok: [localhost]

TASK [Create Network Object for Default Route] *********************************************************************************************
ok: [localhost]

TASK [Create Inside GW] ********************************************************************************************************************
ok: [localhost]

TASK [Create Outside GW] *******************************************************************************************************************
ok: [localhost]

TASK [Create Default Route] ****************************************************************************************************************
ok: [localhost] => (item=8909dd64-c095-11ec-8307-ff7fa67bbc92)
ok: [localhost] => (item=88fac180-c095-11ec-b740-c62eced14a69)

TASK [Get Deployments] *********************************************************************************************************************
ok: [localhost]

TASK [Get Timestamp] ***********************************************************************************************************************
ok: [localhost] => (item={u'version': u'1650452865435', u'type': u'DeployableDevice', u'name': u'FTD2'})
ok: [localhost] => (item={u'version': u'1650452865435', u'type': u'DeployableDevice', u'name': u'FTD1'})

TASK [Get Device List] *********************************************************************************************************************
ok: [localhost]

TASK [Deploy Changes] **********************************************************************************************************************
ok: [localhost]

PLAY RECAP *********************************************************************************************************************************
localhost                  : ok=13   changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

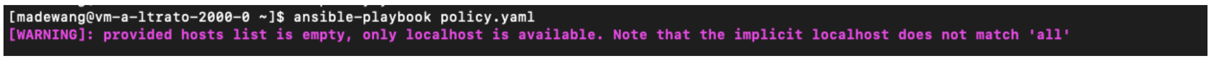
Task 2D: Configure Access Rule on Cisco FMC and deploy to devices.

To configure each interface in all the FTDs here are my assumptions:

- **All the FTD's interface GigabitEthernet 0/0 will be connected to Internal LAN and configured as Inside zone**
- **All the FTD's interface GigabitEthernet 0/1 will be connected to Outside zone and configured as Outside zone**
- **Cisco FTD is already registered to its manager.**

Actions to take:
Note: For the sake of the Lab Execution time, I have already prepared Ansible playbook, which is available at the GitHub, you can clone the playbook and execute. During execution of the playbook this will create a default policy to register the FTD to FMC.

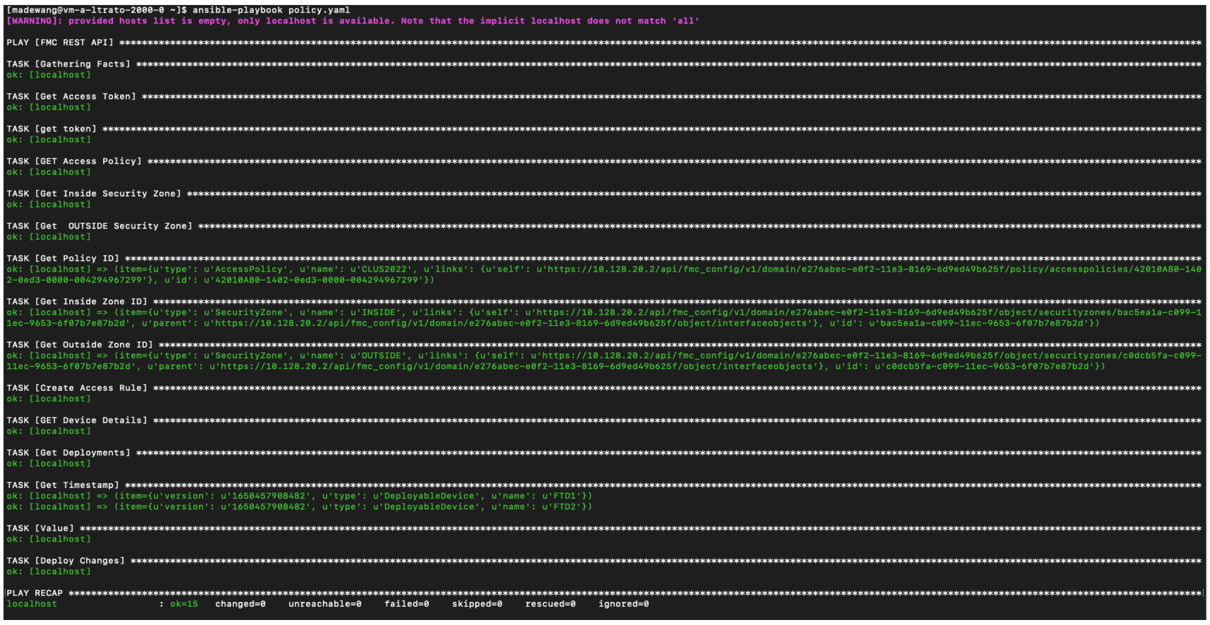
1. **Git clone {{URL for git repository}}**
2. **Open the playbook using vi and edit the Cisco FMC IP as provided in the session.**
3. **Save the playbook.**
4. **Run the playbook to for the setup. "ansible-playbook policy.yaml"**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook policy.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
```

Result:

1. **Provided access rule will be configured in Cisco FMC and will be deployed.**



```
[madewang@vm-a-ltrato-2000-0 ~]$ ansible-playbook policy.yaml
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [FMC REST API] ***********************************************************************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************************************************************************
ok: [localhost]

TASK [Get Access Token] *******************************************************************************************************************************************************
ok: [localhost]

TASK [get token] **************************************************************************************************************************************************************
ok: [localhost]

TASK [GET Access Policy] ******************************************************************************************************************************************************
ok: [localhost]

TASK [Get Inside Security Zone] ***********************************************************************************************************************************************
ok: [localhost]

TASK [Get  OUTSIDE Security Zone] *********************************************************************************************************************************************
ok: [localhost]

TASK [Get Policy ID] **********************************************************************************************************************************************************
ok: [localhost] => (item={u'type': u'AccessPolicy', u'name': u'CLUS2022', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/policy/accesspolicies/42010A80-140
2-0ed3-0000-004294967299'}, u'id': u'42010A80-1402-0ed3-0000-004294967299'})

TASK [Get Inside Zone ID] *****************************************************************************************************************************************************
ok: [localhost] => (item={u'type': u'SecurityZone', u'name': u'INSIDE', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/object/securityzones/bac5ea1a-c099-1
1ec-9653-6f07b7e87b2d', u'parent': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/object/interfaceobjects'}, u'id': u'bac5ea1a-c099-11ec-9653-6f07b7e87b2d'})

TASK [Get Outside Zone ID] ****************************************************************************************************************************************************
ok: [localhost] => (item={u'type': u'SecurityZone', u'name': u'OUTSIDE', u'links': {u'self': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/object/securityzones/c0dcb5fa-c099-
11ec-9653-6f07b7e87b2d', u'parent': u'https://10.128.20.2/api/fmc_config/v1/domain/e276abec-e0f2-11e3-8169-6d9ed49b625f/object/interfaceobjects'}, u'id': u'c0dcb5fa-c099-11ec-9653-6f07b7e87b2d'})

TASK [Create Access Rule] *****************************************************************************************************************************************************
ok: [localhost]

TASK [GET Device Details] *****************************************************************************************************************************************************
ok: [localhost]

TASK [Get Deployments] ********************************************************************************************************************************************************
ok: [localhost]

TASK [Get Timestamp] **********************************************************************************************************************************************************
ok: [localhost] => (item={u'version': u'1650457908482', u'type': u'DeployableDevice', u'name': u'FTD1'})
ok: [localhost] => (item={u'version': u'1650457908482', u'type': u'DeployableDevice', u'name': u'FTD2'})

TASK [Value] ******************************************************************************************************************************************************************
ok: [localhost]

TASK [Deploy Changes] *********************************************************************************************************************************************************
ok: [localhost]

PLAY RECAP ********************************************************************************************************************************************************************
localhost                  : ok=15   changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

----End of Task 2---

Task 3: Hands On Experience for Python and Cisco FTD

In this task, you are going to Interact with Cisco FTD's Rest API using Python.

Python will be running in the Jumphost which should be having access over HTTPS to Cisco FMC. Under this exercise we will be mostly working with requests library of Python which provides all the necessary functions to access the REST API of any given host.

Use cases where Python can be leveraged

1. **Large scale object configuration from any input source**
2. **Merge two Access Policies together**
3. **Enable IPS policy in Specific Policies specific rules.**

Task 3A: Enable Logging at the End for every Rule in given access rule

Automation can be of real help when there is repetition of same task with different values. For example under this task, if we are asked to enable logging at the end for all the rules inside a given ACP.

Such tasks can be easily automated using Python.

Every Python code for accessing the REST API will include the following:

1. **Code to fetch the authentication token from Cisco FMC**
2. **Code to read through the input file.**
3. **Prepare the object which can be used to create the objects in Cisco FMC**
4. **Code to POST the object to Cisco FMC.**

This can further, when enabled with logs and notification can help the user to audit and troubleshoot the issue when encountered.

Actions to take:
Note: For the sake of the Lab Execution time, I have already prepared Python code, which is available at the GitHub, you can clone the playbook and execute.

1. **Git clone {{URL for git repository}}**
2. **Open the playbook using vi and edit the Cisco FMC IP (FMCDetail.py)as provided in the session and add the details of the FMC ACP.**
3. **Save the code.**
4. **Run the code to for the setup. In following order**
   "python GET.py"
   "python Filter.py"
   "python PUTLoggingatTheEND.py"

Result:

1. **All the given objects will be posted to Cisco FMC.**

Task 3B: Merge two access Policies together

There are certain situation when customer need to merge two access policy into single access policy.

One such example will be: when Firepower Migration tool is used, it migrates different configs into different policies, which needs to be merged into one, and each policy can be merged with different category of the Access Policy.

Such use cases can be easily catered by Python.

Such Python code would require following

1. **Code to fetch the authentication token from Cisco FMC**
2. **Code to fetch all the details of access rules into a file**
3. **Massage the fetched file to be used further to post**
4. **Code to POST the rules to Cisco FMC.**

This can further, when enabled with logs and notification can help the user to audit and troubleshoot the issue when encountered.

Actions to take:
Note: For the sake of the Lab Execution time, I have already prepared Python code, which is available at the GitHub, you can clone the playbook and execute.

1. **Git clone {{URL for git repository}}**
2. **Open the playbook using vi and edit the Cisco FMC IP as provided in the session.**
3. **Save the code.**

4. Run the code to for merging. “python main.py”

Result:

1. Access Policy will be merged into one.

Task 3C: Enable IPS is selected rule

After greenfield deployment or after migration, IPS policy is not enabled on all of the access rule rather based on the customer input it will be enabled on the specific rules. Hence if the customer can provide list of rules where IPS needs to be enable, Python script can be leveraged to edit to the rule and enable the IPS rule.

Such use cases can be easily catered by Python.

Such Python code would require following

1. Code to fetch the authentication token from Cisco FMC
2. Code to fetch all the details of access rules into a file
3. Massage the fetched file to be used further to edit and put
4. Code to PUT the rules to Cisco FMC.

This can further, when enabled with logs and notification can help the user to audit and troubleshoot the issue when encountered.

Actions to take:
Note: For the sake of the Lab Execution time, I have already prepared Python code, which is available at the GitHub, you can clone the playbook and execute.

1. Git clone {{URL for git repository}}
2. Open the playbook using vi and edit the Cisco FMC IP as provided in the session.
3. Save the playbook.
4. Run the code to for the setup. In following order
   “python GET.py”
   “python Filter.py”
   “python PUTIPS_Policy.py”

Result:

2. **Selected rules will have IPS enabled**

----End of Task 3----