Automating Service Providers tasks using Ansible

LAB GUIDE

Introduction

Ansible is an open-source software provisioning, configuration management, and application-deployment tool enabling Infrastructure as Code. Ansible is Python-based and can talk to managed nodes using SSH or HTTPS with REST for broad platform support and compatibility. Ansible supports a wide range of vendors, device types, and actions, so that you can manage your workstations, servers, applications, network devices - and now Cisco Meraki - with a single automation tool.

One of the greatest features of Ansible is that it is agentless. You do not need to install anything on the managed node to control. You can control Linux workstations and servers can be controlled over SSH and Cisco Meraki is controlled using REST APIs. Other protocols are possible using many different Python libraries.

Another feature of Ansible is that when modules run configurations, they set things to a desired state. Additional runs of the same configuration results in no changes. This feature is idempotence, from the Latin idem and potence ("same" and "power"). In mathematics and computer science, it is "the property of certain operations that can be applied multiple times without changing the result beyond the initial application". When you run an Ansible playbook, the result is a consistent configuration state, whether your managed node needed no changes, few changes, or many.
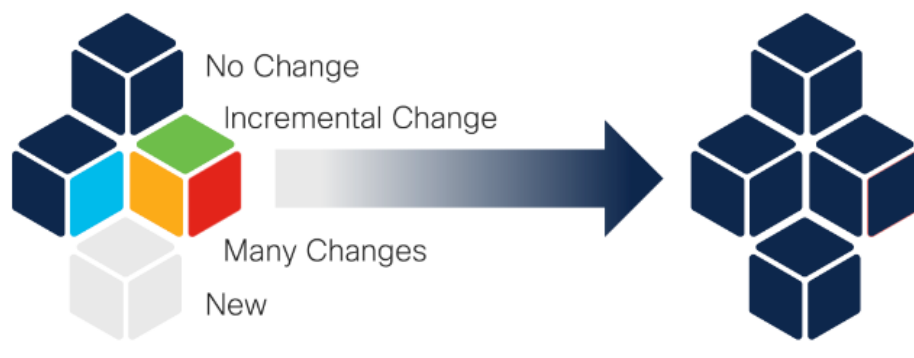
This session has been divided into two different labs:

- **LAB 1: Understanding the Basics: ANSIBLE and MERAKI (90 min)**
    - In this lab, you will follow a self-paced lab that will provide them the basic knowledge of the new Ansible collection and Infrastructure as a Code concepts.
- **LAB2: Manage and create your own playbooks (2hr)**
    - In this lab, you will apply the knowledge gained from the previous lab to create some Ansible modules that can be very valuable for your day-to-day work activities.

LAB 1 | ANSIBLE and MERAKI (90 minutes)

In this Lab, you will learn how to use Ansible with the Cisco Meraki's Dashboard API. This is a completely self-paced lab that have been created specifically to help you to get a basic knowledge and understanding of Ansible concepts.

The lab is available in our cisco Devnet website. Please, follow the next steps to access to the environment:

- **Step 1:** go to **https://developer.cisco.com/**
- **Step2:** at the top right of the page, click on "**SIGN UP FREE**". Choose the login method more convenient for you.
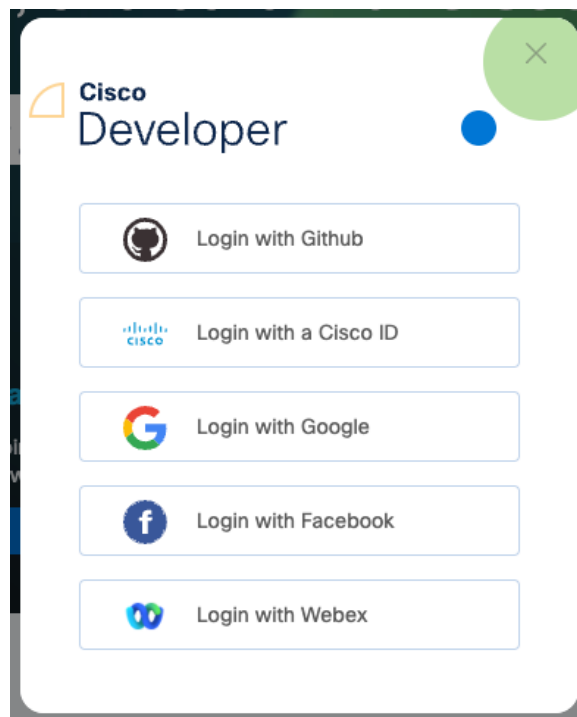
- **Step 3:** Once you have completed the login process go to the search bar and type "**ANSIBLE MERAKI LAB**". You will see a list of outputs. Please, click on this one:

You're now in!

The lab environment is divided into three sections. On the left side of the screen you'll see the self-paced instructions to follow to complete the tasks. On the right-hand side there are:

- A terminal window, that you will use to insert the commands required during the lab and launch de Ansible commands.
- A folder navigator and editor to create/edit Ansible modules.

Before starting the lab, we also encourage you to create a new fresh Dashboard Organization where you, as administrator, will have full read-write privileges. It is also recommended, but not required, that you use a user account (email) that you do not use for your Production Organizations.

- **Step 4: Create a dashboard organization:**

  - **Step 4.1**: go to https://dashboard.meraki.com
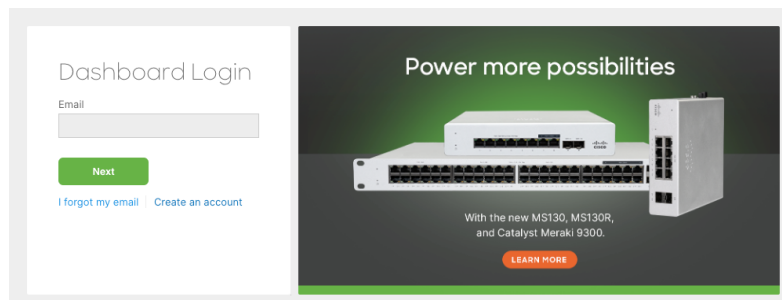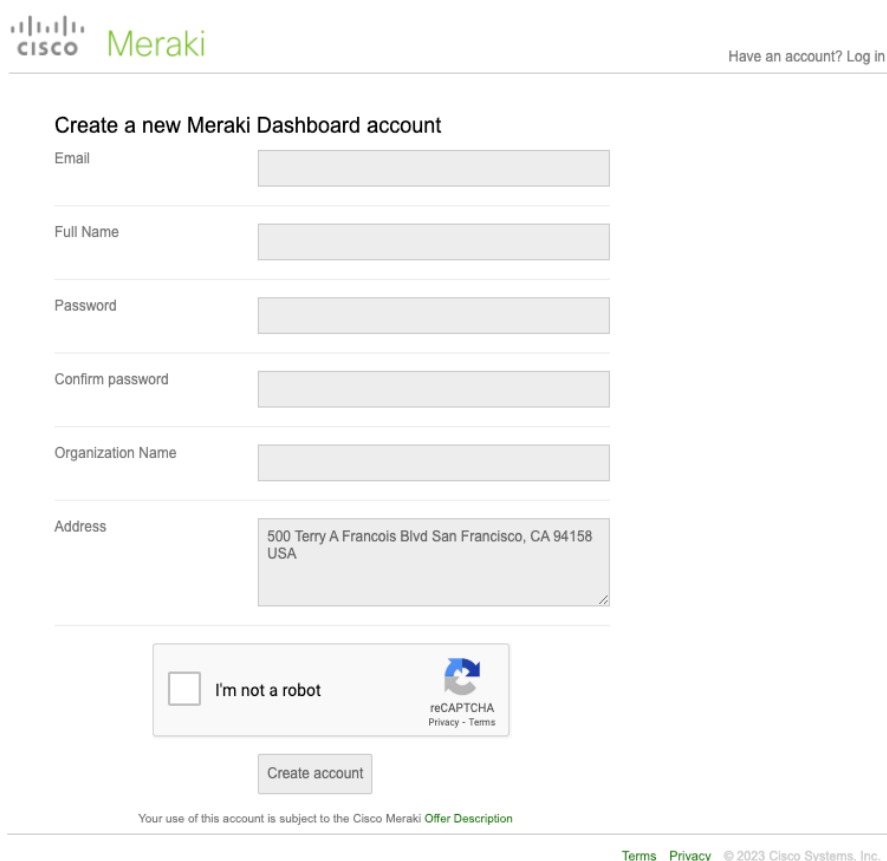
- **Step 4.2**: click on "Create an account". Select the "Europe" region and click "Next".

- **Step 4.3**: Fill in the form:

Please, remember that:

- The email address must be valid. Avoid, whenever possible, using any email you're already using for your production environments.

- Choose any Organization name of your convenience.

- **Step 5: Activate API access to the Organization.**

By default, API access to the Organization is disabled so, let's activate it. Go to **Organization → Settings** and check the API access flag. Save your changes.

Dashboard API access

API Access ⓘ   ☑ Enable access to the Cisco Meraki Dashboard API

- **Step 6: Create your dashboard API key:**

To create the API key associated with you user account, please login into the dashboard using your email address and selected password. Then, move to **Organization→Configure→API & webhooks.**
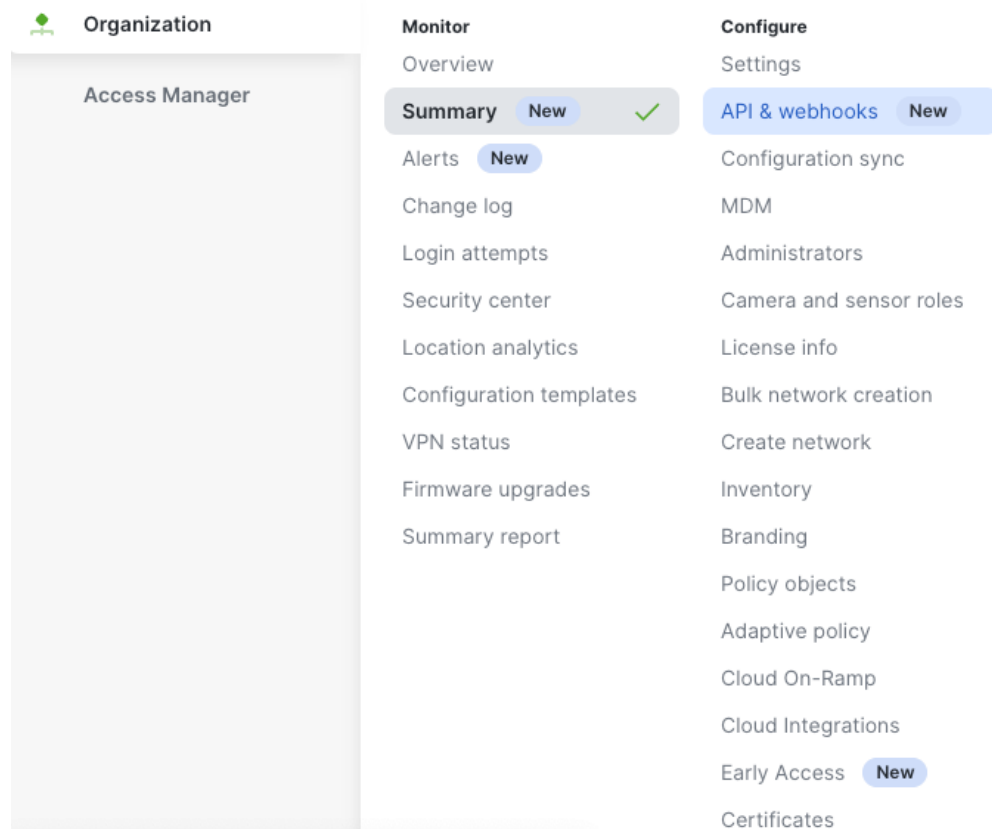
Now click on the "Generate API key" button:

On the next screen, you'll see the API Keys associated to your account. It is expected that, if you're using your email account for the first time, no API keys will exist. So, one more time, click on "Generate API Key".

You will automatically see a pop-up window with the API KEY generated. Please, **copy this APY KEY** in your notepad (you need to click and accept that the API has been stored to proceed). Bear in mind that the API KEY is shown once, and you won't be able to see it again in the dashboard, but you can revoke the KEY and create a new one anytime you need.

Then, your API key is ready for being used. The figure below shows how your dashboard should look like after completing all the above steps (the API characters you see on this screenshot will differ from your output).

API & Webhooks

Overview   **API keys and access**   Webhooks

Personal API keys ⓘ

Maximum of 2 keys   **Generate API Key**

API key	Time created	Actions
*****************11c2	Nov 02 2023 12:32	**Revoke key**

To proceed with the lab activities, we recommend that you use the Devnet Sandbox API KEY **for activities 1-5**. The API KEY is:

MERAKI DEVNET SANDBOX LAB API KEY: 6bec40cf957de430a6f1f2baa056b99a4fac9ea0

When your reach the activity number 5 of the Lab ("*Ansible Collection Help and Query Examples*"), there are 2 playbooks to query for data from the Devnet Sandbox Organization. There's also a folder with more ansible Playbooks that you can look at, but let's focus on the 2 Playbooks that you can see in your Lab environment:

- Get Organization Lists
- Fetch all devices in your organization's inventory.

To execute these Playbooks, just create 2 yaml files by clicking on the "+" sign in your Lab editor:
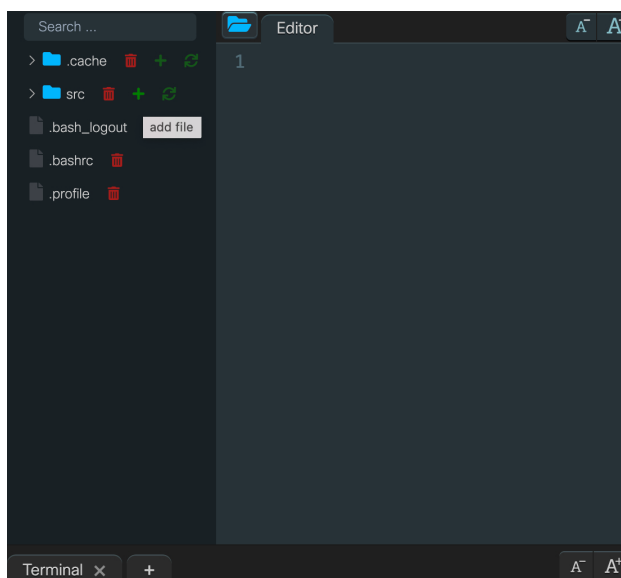
Name the files:

developer.cisco.com says

Enter a new filename:

myplaybook.yaml

Cancel   OK

Once you hit "OK", the file is creted. As you can see, it's empty:
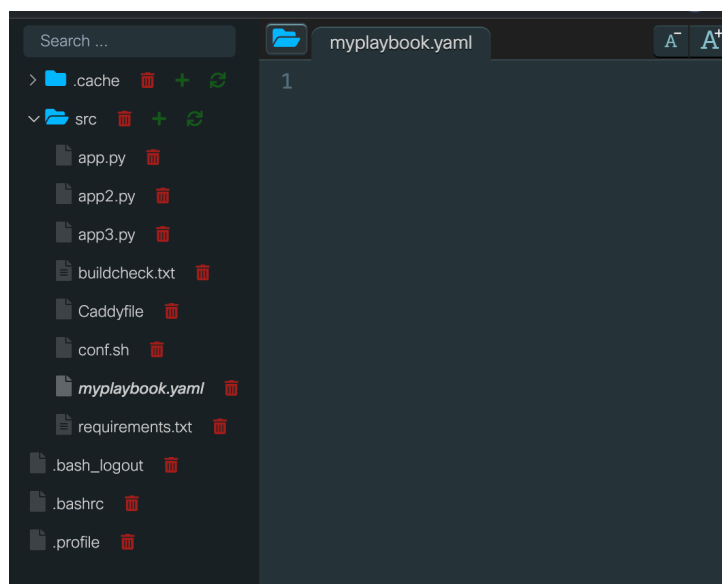
You can then copy and paste the playbook. The file will be automatically saved and ready to be executed.

```yaml
---
- hosts: meraki_servers
  gather_facts: false
  tasks:
    - name: update ports
      cisco.meraki.devices_switch_ports:
        meraki_suppress_logging: true
        state: present
        serial: "AAAA-BBBB-CCCC"
        vlan: 10
        portId: 1
```

Activities 6 and 7 from the Lab requires write permissions. Therefore, the Devnet Sandbox API Key is not valid. We're going to use your new Organization to make this activity. To do so:

- **Step1:** Change the API key that you're using in your lab environment. Remember, up to exercise 5 you've been using the Devnet API KEY. Please change the environment variable to use the API key that you have just generated in your personal dashboard Organization by running this command:

```
export MERAKI_DASHBOARD_API_KEY=<your API key>
```

- **Step 2:** Login into the Organization by using your email account and password.
- **Step 3:** Create a new (combined hardware) network (*Organization→ Configure→ Create Network*)

Activity 6 requires you know some additional parameters, like your Organization ID and the Network ID of the network you have just created. The Organization ID is shown at the bottom of any Dashboard page. As an example, you will see something like this (your ID number will be different from the one shown here)

Data for Ansible Lab (organization ID: 714946440845067383) is hosted in **Europe**

Finally, the "*networkId*" parameter is also required to execute one of the Playbooks. To know the ID of the network that you have just created, please create a yaml file in your lab environment and copy the following Ansible task:

```yaml
---
- hosts: meraki_servers
  gather_facts: false
  tasks:
   - name: Get Networks
     cisco.meraki.networks_info:
       meraki_suppress_logging: true
       organizationId: xxxxxxxxxxxxx # replace with your organization ID
     register: result
   - name: Show result
     ansible.builtin.debug:
       msg: "{{ result }}"
```

Save the file (for example, you can name it as **getOrganizationNetworks.yaml**) and execute it. You should see an output with the Network information from your organization including the ID of your network. Copy it.

You will also see that the activity 6 of the Lab contains 2 Ansible files. The first one modifies the settings from a port of a switch. Since you don't have a device in your organization, you will see that the Ansible task will fail. This is normal, since the serial number of the switch does not exist. In any case, create the Ansible file and execute it to see the error message. You should see the following output:

```
TASK [update ports] ********************************************************
Error:  An error occured when executing operation.The error was: switch, getDeviceSwitchPort – 404 N
ot Found, please wait a minute if the key or org was just newly created.
An exception occurred during task execution. To see the full traceback, use -vvv. The error was: Non
eType: None
fatal: [meraki_server]: FAILED! => {"changed": false, "msg": "Object does not exists, plugin only ha
s update"}

PLAY RECAP ********************************************************
meraki_server              : ok=0    changed=0    unreachable=0    failed=1    skipped=0    rescued=
0    ignored=0
```

The second task of exercise 6 is about creating and configure a wireless SSID. Even if you don't have any access point in the Organization and the Network, this task can be accomplished successfully. Please note that you must use your network ID as parameter.

Once you have successfully executed the Ansible playbook, check in the dashboard the result (go to the network created and then to *Wireless* → *Configure* → *SSIDs*). You will see that the SSID has been successfully created.

Configuration overview

SSIDs	Showing 4 of 15 SSIDs. Show all my SSIDs.
	ACME1
Enabled	enabled ∨
Name	rename
Enterprise Admins	access enabled ∨
Access control	edit settings
Encryption	PSK (WPA2)
Sign-on method	None
Bandwidth limit	unlimited
Client IP assignment	Local LAN
Clients blocked from using LAN	yes
Wired clients are part of Wi-Fi network	no
VLAN tag ⓘ	100
VPN	Disabled
Splash page	
Splash page enabled	no
Splash theme	n/a

You can also look at the change log to see the API call made to create the SSID (***Organization → Monitor → Change Log***)

In the activity 7 you are going to create a few additional SSIDs using an Ansible script that utilizes loops. Just copy and paste the ansible playbook and insert the ORG ID of your dashboard organization as parameter. As you did on task 6, check the results on the SSID page in dashboard (***Wireless → Configure → SSIDs)*** and in the change log.

CONGRATULATIONS! YOU HAVE SUCCESSFULLY COMPLETED LAB 1

LAB 2 | Manage and create your own playbooks (2hr)

As indicated in the introduction, this laboratory aims to put into practice the knowledge acquired during the first part of this session. The idea is that, at the end of this laboratory, you can take with you a series of playbooks that can be useful for your daily life and that, based on what you learned today, you will be able to create your own playbooks.

Since you would want to operate on your personal Organization, once of the first things you must do (if you have not done it already) is to modify the current API KEY in use for your personal API KEY that you've just generated at the beginning of this session. So, in the terminal window please type:

```
export MERAKI_DASHBOARD_API_KEY=<your API key>
```

From now on, all the Ansible modules you run will change (via REST API) the configurations and settings on your personal Meraki Dashboard Organization.

- **Activity 1: Get familiar with Ansible documentation.**

One of the great benefits of using Ansible over Programming languages like Python is that Ansible is very specific in the syntax and way of creating a Playbook, so that there is one, and only one, way to do and complete a task. With Python, for example, the same problem can be solved in multiple ways, depending on the experience and capabilities of the programmer. This makes the learning curve of Ansible, compared to Python, much smaller and allows users with very little or no experience, and with the help of documentation, to create their own Playbooks in a very short time.

Assuming that most of you in this lab are new to Ansible, one of the first things you should do is familiarize yourself with the documentation. You can not only find all the modules currently available for Meraki's Ansible collection, but also how to use them and examples that you can copy and paste to create your tasks easily. In this way, and with the help of the documentation, you will be able to create some basic automation scripts that will help you build the foundations to be able to perform more complex tasks in the future.

The official documentation of the Ansible collection for Meraki is available at the following URL. Look for a few minutes at some of the available modules.

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/

You can also find there the installation instructions:
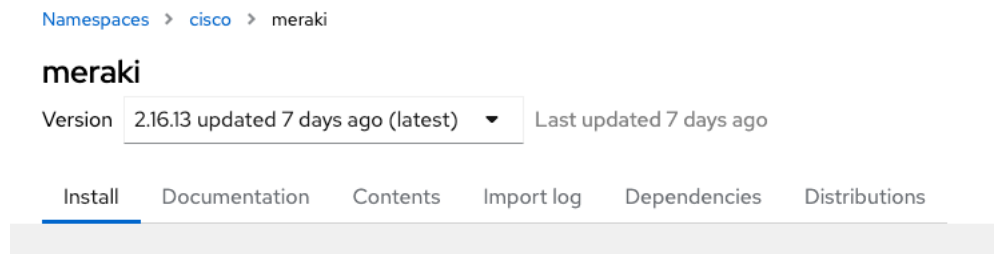
In our case, it is not necessary to install anything since you are going to use the Laboratory environment to run your Playbooks. Therefore, click on "**Documentation**", where you can see all the Ansible modules currently available.
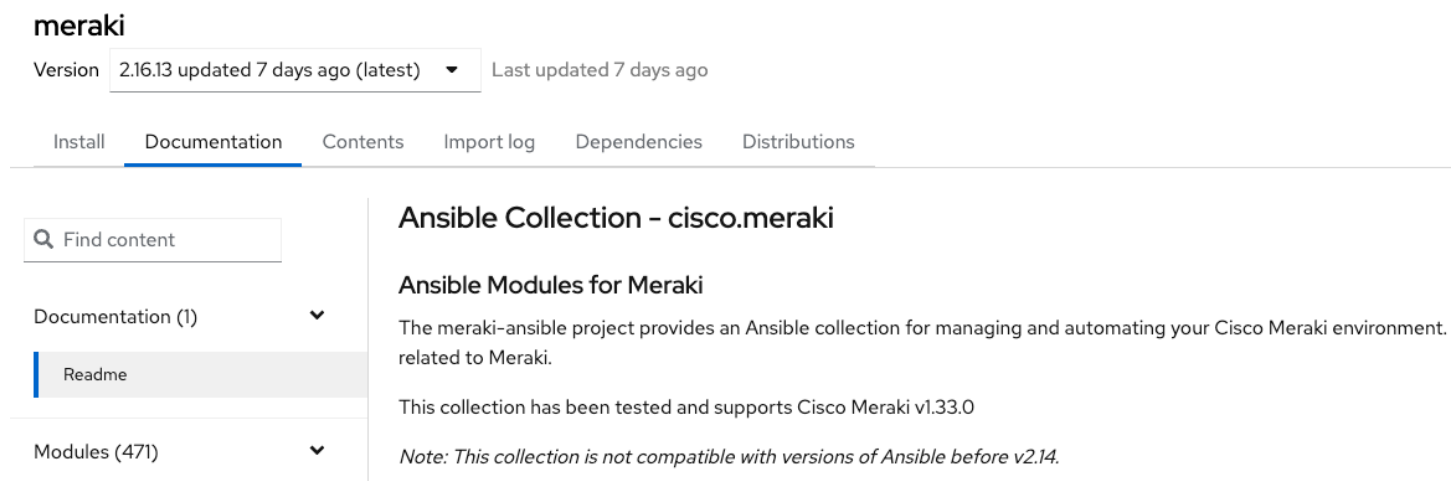
As you may have seen, the current number of Ansible modules (471) are like the API calls available in the Dashboard API. If you click on any of the modules, you will see the same structure in all of them:

o A brief explanation of what the module does

o Minimum requirements

o Parameters accepted by the module

o an example

Those examples will be useful for you to start building your own Playbooks. With a little practice and patience, you will notice that in a short time you can automate numerous tasks in your daily job.

Let's try!

- **Activity 2: Create your own Playbooks.**

Before start, please do the following:

- **Step 1:** Login into your Dashboard Organization and create a new, empty combined-type network. Choose any network name you like.
- **Step 2:** With the help of the Ansible playbook that you already used on the Lab 1(see page 9 of this document), run it again and get the Network ID of the network you've just created. Copy this value.

Let's now to automate the configuration of this network. Here's the desired outcome:

- **MX Security & SD-WAN appliance:**

  o Enable VLANs

  o Create a **Corporate VLAN** with the following configuration:
    - VLAN ID: **100**
    - Subnet: **192.168.100.0/24**
    - Appliance IP: **192.168.1.1**
    - Name: **Corporate VLAN**
    - IPv6: **Disabled**

  o Create a **Guest VLAN** with the following configuration:
    - VLAN ID: **200**
    - Subnet: **192.168.200.0/24**
    - Appliance IP: **192.168.200.1**
    - Name: **Guest VLAN**
    - IPv6: **Disabled**

  o Create a **L3 Firewall rule** to prevent the Access to "*internetbadguys.com*"

  o Create a **traffic shaping rule** to limit the client's bandwidth to *5Mbps (down)/2Mbps (up)*

  o Enable the **Anti-Malware** protection.

  o Enable the **IDS /IPS** with the following configuration:

    - IDS/IPS mode: **prevention** mode
    - Ruleet: **Balanced**

- **MR SSID Configuration**

  o Create 2 SSIDs with the following settings:

    - **Corporate SSID**
      - Name: **Corporate WiFi**
      - Radius Authentication with the following parameters:
        - Radius IP: **1.2.3.4**
        - Port: **1812**
        - Secret: **Super$ecret**
      - IP assigment mode: **bridge mode**
      - VLAN tagging **enabled.**
      - VLAN Tag: **100**

    - **Guest SSID**
      - Name: **Guest WiFi**
      - Autenticación mode: **open**
      - Splash Page: **Click-Through**
      - IP assigment mode: **bridge mode**
      - VLAN tagging **enabled**.
      - VLAN Tag: **200**

  o On the guest SSID, add a Firewall rule to block the guest traffic from reaching internal destinations.

- **MS 802.1x access policy**

For users connecting via a wired port on MS switch, we want to create an access policy with the following settings:

  - Policy Name: **MyAccessPolicy**
  - Access Policy Type: **Hybrid Authentication**
  - Dot1x control direction: **inbound**
  - Host mode: **single host**
  - Data VLAN ID: **100**
  - Guest VLAN ID: **200**
  - Radius **Accounting enabled**.
  - Radius **CoA enabled**.
  - Radius IP: **1.2.3.4**
  - Port: **1812**
  - Secret: **Super$ecret**
  - Radius **testing disabled**
  - Walled Garden **Enabled**
  - Walled Garden Range IP: **192.168.128.0/24**

As you can see, there are a lot of different things to configure. Let's divide the problem into small pieces. So, create 3 different Ansible playbooks (we will merge them later):

- One playbook to accomplish the MX tasks.
- One playbook to accomplish the MS tasks.
- One playbook to accomplish the MR tasks.

Since the documentation and the number of modules currently available are very high, and the time to complete the lab is limited, we'll put you here all the modules that can help you to create the playbooks. Look at them, see the parameters required and try to discard those that may not be valid for your task. Finally, check the examples of how to use them and copy & paste anything that can be useful for you.

Reference MX modules:

- **Create MX VLANS:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_appliance_vlans/

- **Enable MX VLANS:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_appliance_vlans_settings/

- **Configure VLANs on MX:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_appliance_vlans/

- **Configure Traffic Shaping on MX:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_appliance_traffic_shaping/

- **Configure Firewall Rules on MX:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_appliance_firewall_l3_firewall_rules/

- **Configure AMP Settings:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_appliance_security_malware/

- **Configure IDP/IPS Settings:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_appliance_security_intrusion/

Reference MS modules:

- **Create 8021.x Access Policy:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_switch_access_policies/

Reference MR modules:

- **Create SSIDs:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_wireless_ssids/

- **Update Firewall rules on SSIDs:**

https://galaxy.ansible.com/ui/repo/published/cisco/meraki/content/module/networks_wireless_ssids_firewall_l3_firewall_rules/

Once the Playbooks are executed successfully, go to your Dashboard Organization and check the changes made on the network and on the device's configurations. Confirm that the current device's configurations are consistent with the Playbook's tasks.

Problems? Questions? Let us know!

You can check and download the results here: https://github.com/isantosg/Ansible

- **Activity 3: Configuration consistency and rollback.**

In this final exercise, we are going to see how Ansible can help us to maintain equipment configurations and ensure their consistency. The Playbook, in short, collects the desired state that we want for the network and the device, in such a way that, if an administrator makes any unwanted or erroneous changes to the dashboard, the Playbook will allow us to make a rollback of those modifications to the desired state.

To proceed with the exercise, we are first going to merge the three playbooks that you have created previously into a single Playbook. This Playbook will be our "golden configuration" for the network.

Task 1: Combine the three Playbooks into a single Playbook. Save it with whatever name you want.

Task 2: Access to your dashboard Organization and make some changes on the settings. As an example:

- You can disable AMP on the MX
- Change Firewall settings for the SSDID Guest
- Change the IP address of the Radius Server on the MS 802.1x access policy.
- Change the corporate VLAN IP address, or its VLAN ID on the MX

Task 3: Finally, run the Playbook containing the "golden" configuration. Observe how Ansible reviews the consistency between the desired state (the one that is included in the Playbook) and the real one (the current dashboard configuration) in such a way that the network will be reversed back according to our Golden Playbook, reverting the changes that we have incorrectly made in the settings.

CONGRATULATIONS! YOU HAVE SUCCESSFULLY COMPLETED LAB 2