## Creating a Dynamic Inventory using the Nautobot GraphQL Dynamic Inventory Plugin

To facilitate the creation of a dynamic inventory using Nautobot, we will provide use Nautobot's GraphQL dynamic inventory plugin for Ansible. GraphQL is a powerful data query language for APIs that allows users to specify the exact data they require for a particular task. Dynamic inventories, as opposed to static inventories, enable us to keep track of all our onboarded devices and facilitate the application of configuration changes as our network expands. By combining the power of GraphQL with Ansible, we can create an inventory that is highly scalable and surpasses the limitations of a static Ansible inventory file.

1. On the desktop-linux-1-vm instance in Google Cloud, open the Terminal and execute the following commands to install Ansible on the instance:
   * sudo apt update
   * sudo apt install software-properties-common
   * sudo add-apt-repository --yes --update ppa:ansible/ansible
   * sudo apt install ansible



1. Install the required packages for the Nautobot Ansible collection:
   * sudo apt install python3-pip
   * sudo pip install netutils

A screen shot of a computer

Description automatically generated

1. Switch to the **/etc/ansible** directory.

A black background with white text

Description automatically generated

1. Use the **chmod** command to change the file permissions of the **ansible.cfg** file.

**A screen shot of a computer

Description automatically generated**

1. Create a complete initial Ansible configuration using the **ansible-config** command.

ansible-config init --disabled -t all > ansible.cfg

1. Install the Nautobot galaxy collection using the command **ansible-galaxy collection install networktocode.nautobot**. This installs our dynamic inventory plugin alongside other Ansible modules for Nautobot.

A screen shot of a computer

Description automatically generated

1. On a web browser, navigate to the internal IP address of the Nautobot instance and login.
2. In the Nautobot GUI, navigate to Admin > Profile > API Tokens

A screenshot of a computer

Description automatically generated

1. Click “**Add a token**”

A screenshot of a computer

Description automatically generated

1. Click **Create** to accept the default settings. This will create an API token so that GraphQL can query Nautobot for information. This token will never expire. Keep this token handy in a text file as we will use it later.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Right-click on the desktop to create a folder named **ansible-gql**. We will use this folder to store our dynamic inventory and Ansible playbooks.

A screenshot of a computer

Description automatically generated

1. Open the ansible-gql folder and create a YAML configuration file for the dynamic inventory plugin. We will name it **inventory.yml**.

A screenshot of a computer

Description automatically generated

1. Open **inventory.yml** in Visual Studio Clde by right-clicking on the file and selecting **Open With “Visual Studio Code”**

A screenshot of a computer

Description automatically generated

1. Enter the following configuration details for the dynamic inventory plugin into the file:

plugin: networktocode.nautobot.gql\_inventory

api\_endpoint: https://<IP\_of\_nautobot>

token: API token created from step 6

validate\_certs: false

This is the minimum required configuration for the dynamic inventory plugin to work. The **validate\_certs** parameter, while optional, is required for our environment because of Nautobot’s use of self-signed SSL/TLS certificates

A screen shot of a computer

Description automatically generated

1. Save the file by pressing CTRL+S
2. Use the command **ansible-inventory -v --list -i inventory.yml** to test the inventory plugin. Ansible will dynamically query Nautobot via GraphQL using the configuration details in inventory.yml and display information for each onboarded device in Nautobot.

A screenshot of a computer

Description automatically generated