Ansible AWX Training

Introduction

This document provides an overview on Ansible and AWX features and hands on tutorials.

Installation

Please refer to the following link: https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html

CentOS Red Hat Installation

These are the steps to install AWX/Ansible on top of CentOS/RHEL:

#EPEL Repo

```
yum -y install epel-release
```

# Install Additional Packages

```
yum -y install git gettext ansible docker nodejs npm gcc-c++ bzip2
yum -y install python-docker-py
```

# Run Docker

```
systemctl start docker
systemctl enable docker
```

```
# Clone AWX Repo and Deploy
git clone https://github.com/ansible/awx.git
cd awx/installer/
ansible-playbook -i inventory install.yml
```

Notes

You might need to modify git commands that access remote repositories if your internet access is through a proxy server. Once you have obtained the proxy settings (server URL, port, username and password); you need to configure your git as follows:

```
# git config --global http.proxy http://<username>:<password>@<proxy-server-url>:<port>
```

You would need to replace <username>, <password>, <proxy-server-url>, <port> with the values specific to your proxy server credentials.

If you do not specify "project_data_dir=/var/lib/awx/projects" during the deployment, you can't use Manual SCM. You might face issue while creating the new project.

If you are behind an HTTP or HTTPS proxy server, you need to add the proxy configuration in the Docker systemd service file.

Docker

AWX relies on five Docker containers provided by Red Hat Ansible in Docker Hub (https://hub.docker.com/u/ansible).

The Ansible installation creates a PostgreSQL database that will be in one container and will create the AWX one that contains the web tier, the engine, a cache and a queue:

- AWX Web. Server which provides HTTP GUI access.
- AWX Task. Engine responsible of providing API interworking for solution components.
- RabbitMq. Acting as a messaging broker between solution components.
- Memcached. Memcached is a general-purpose distributed memory caching system. It is used to speed up dynamic database-driven websites by caching data and objects in RAM to reduce the number of times an external data source (such as a database or API) must be read.
- Postgres database. AWX requires access to a PostgreSQL database, and by default, one will be created and deployed in a container, and data will be persisted to a host volume.

Docker Getting Started

Once Docker is installed, open a terminal and type the following commands:

```
#docker info
#docker -v
```

If the installation worked, you will see a bunch of information about your Docker installation and docker version running:

```
# docker -v
# docker info
Containers: 0
 Running: 0
 Paused: 0
 Stopped: 0
Images: 0
Server Version: 1.13.1
Storage Driver: overlay2
 Backing Filesystem: extfs
 Supports d_type: true
 Native Overlay Diff: true
Logging Driver: journald
Cgroup Driver: systemd
Plugins:
 Volume: local
 Network: bridge host macvlan null overlay
 Authorization: rhel-push-plugin
Swarm: inactive
Runtimes: docker-runc runc
Default Runtime: docker-runc
```

You can list the containers that are running by issuing the following command:

```
# docker ps
CONTAINER ID        IMAGE
COMMAND                      CREATED             STATUS
PORTS                                                      NAMES
28143772a881        ansible/awx_task:2.1.2              "/tini -- /bin/
sh ..."              19 minutes ago      Up 19 minutes         8052/
tcp                                              awx_task
ded05efa006d        ansible/awx_web:2.1.2               "/tini -- /bin/
sh ..."              19 minutes ago      Up 19 minutes         0.0.0.0:80->8052/
tcp                                    awx_web
a76b6a798a5b        memcached:alpine                    "docker-entrypoint..."
     43 hours ago        Up 6 hours              11211/
```

Use the command "docker exec -it <container ID> /bin/bash" to get a bash shell in the container:

```
[root@RedhatNewAnsible projects]# docker exec -it ded05efa006d  /bin/bash
[root@awxweb awx]# ls
awxfifo  favicon.ico  projects  public  supervisord.log  supervisord.pid  venv  wsgi.py
[root@awxweb awx]# exit
exit
[root@RedhatNewAnsible projects]#
```

Docker Useful Commands

Stops one or more containers.

```
docker stop my_container
```

Stop all running containers:

```
docker stop $(docker ps -a -q)
```

The following command does not attempt to shut down the process gracefully first:

```
docker kill my_container
```

Start one or more containers:

```
docker start my_container
```

This command displays the logs of a container:

```
docker logs --follow my_container
```

Remove one or more containers:

```
docker rm my_container
```

Remove one or more images:

docker rmi my_image

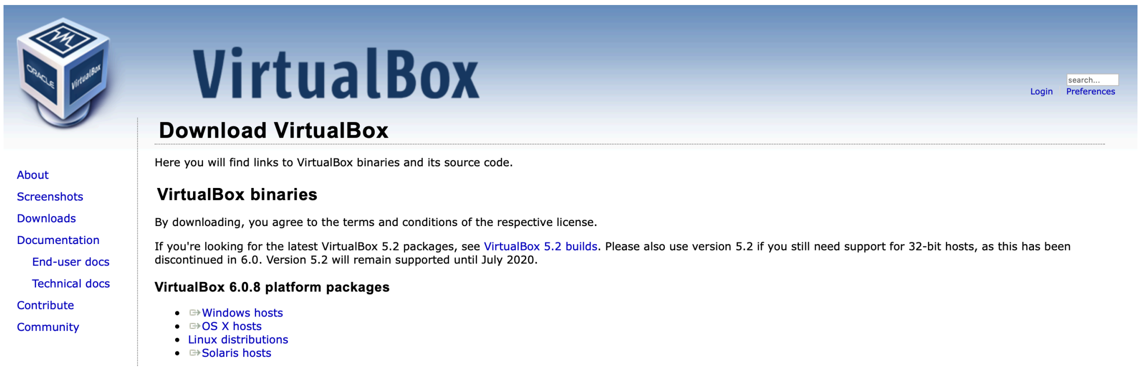
Running Ansible on your laptop

Vagrant is a tool for building and managing virtual machine environments. Vagrant provides easy to configure, reproducible, and portable work environments built on top of industry-standard technology.

Installing Vagrant

First of all, request local admin rights in advance to install the required software.

Download Virtualbox: https://www.virtualbox.org and install it.



Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

VirtualBox binaries

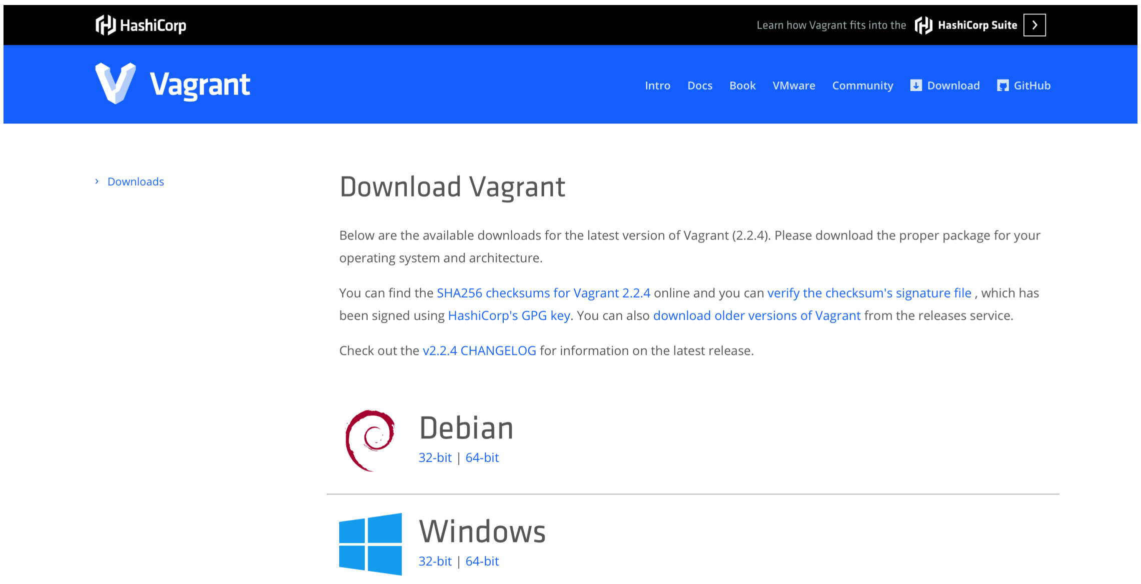
By downloading, you agree to the terms and conditions of the respective license.

If you're looking for the latest VirtualBox 5.2 packages, see VirtualBox 5.2 builds. Please also use version 5.2 if you still need support for 32-bit hosts, as this has been discontinued in 6.0. Version 5.2 will remain supported until July 2020.

VirtualBox 6.0.8 platform packages

- Windows hosts
- OS X hosts
- Linux distributions
- Solaris hosts

Download Vagrant: https://www.vagrantup.com for your operating system.



Download Vagrant

Below are the available downloads for the latest version of Vagrant (2.2.4). Please download the proper package for your operating system and architecture.

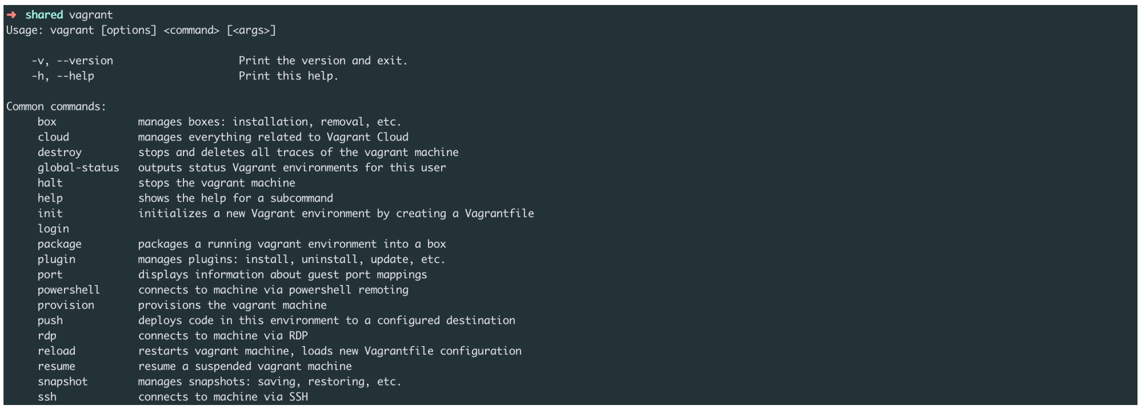
You can find the SHA256 checksums for Vagrant 2.2.4 online and you can verify the checksum's signature file , which has been signed using HashiCorp's GPG key. You can also download older versions of Vagrant from the releases service.

Check out the v2.2.4 CHANGELOG for information on the latest release.

Debian
32-bit | 64-bit

Windows
32-bit | 64-bit

Ensure that Vagrant is properly installed by issuing the "vagrant" command as shown below:



```
➜ shared vagrant
Usage: vagrant [options] <command> [<args>]

    -v, --version                    Print the version and exit.
    -h, --help                       Print this help.

Common commands:
     box             manages boxes: installation, removal, etc.
     cloud           manages everything related to Vagrant Cloud
     destroy         stops and deletes all traces of the vagrant machine
     global-status   outputs status Vagrant environments for this user
     halt            stops the vagrant machine
     help            shows the help for a subcommand
     init            initializes a new Vagrant environment by creating a Vagrantfile
     login
     package         packages a running vagrant environment into a box
     plugin          manages plugins: install, uninstall, update, etc.
     port            displays information about guest port mappings
     powershell      connects to machine via powershell remoting
     provision       provisions the vagrant machine
     push            deploys code in this environment to a configured destination
     rdp             connects to machine via RDP
     reload          restarts vagrant machine, loads new Vagrantfile configuration
     resume          resume a suspended vagrant machine
     snapshot        manages snapshots: saving, restoring, etc.
     ssh             connects to machine via SSH
```

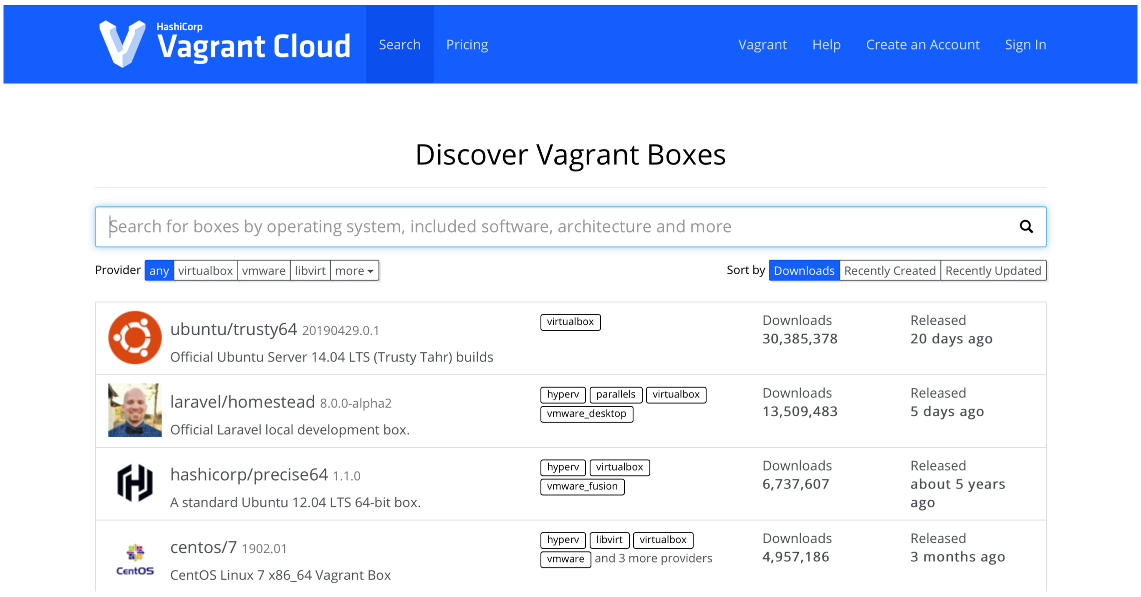
Create a local directory and then make a `shared` folder.

sudo mkdir centos7

cd centos7

sudo mkdir shared

Download the boxes you need to deploy from Vagrant Cloud: https://app.vagrantup.com/boxes/search (i.e. centos/7)


HashiCorp
Vagrant Cloud
Search
Pricing
Vagrant
Help
Create an Account
Sign In
Discover Vagrant Boxes
Search for boxes by operating system, included software, architecture and more
Provider any virtualbox vmware libvirt more
Sort by Downloads Recently Created Recently Updated

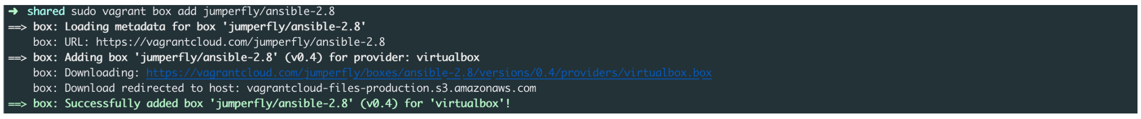
ubuntu/trusty64 20190429.0.1 Official Ubuntu Server 14.04 LTS (Trusty Tahr) builds	virtualbox	Downloads 30,385,378	Released 20 days ago
laravel/homestead 8.0.0-alpha2 Official Laravel local development box.	hyperv parallels virtualbox vmware_desktop	Downloads 13,509,483	Released 5 days ago
hashicorp/precise64 1.1.0 A standard Ubuntu 12.04 LTS 64-bit box.	hyperv virtualbox vmware_fusion	Downloads 6,737,607	Released about 5 years ago
centos/7 1902.01 CentOS Linux 7 x86_64 Vagrant Box	hyperv libvirt virtualbox vmware and 3 more providers	Downloads 4,957,186	Released 3 months ago

The command below downloads a centos 7 box

sudo vagrant box add centos/7

In this case, I am downloading a box called "jumperfly/ansible-2.8" which provides Ansible already installed.

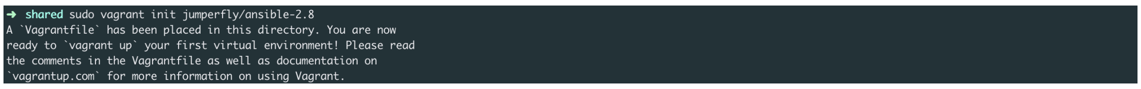
NOTE: This is not an official box so be aware of the security risks.



```
➜  shared sudo vagrant box add jumperfly/ansible-2.8
==> box: Loading metadata for box 'jumperfly/ansible-2.8'
    box: URL: https://vagrantcloud.com/jumperfly/ansible-2.8
==> box: Adding box 'jumperfly/ansible-2.8' (v0.4) for provider: virtualbox
    box: Downloading: https://vagrantcloud.com/jumperfly/boxes/ansible-2.8/versions/0.4/providers/virtualbox.box
    box: Download redirected to host: vagrantcloud-files-production.s3.amazonaws.com
==> box: Successfully added box 'jumperfly/ansible-2.8' (v0.4) for 'virtualbox'!
```

Next step is to initialise the box.

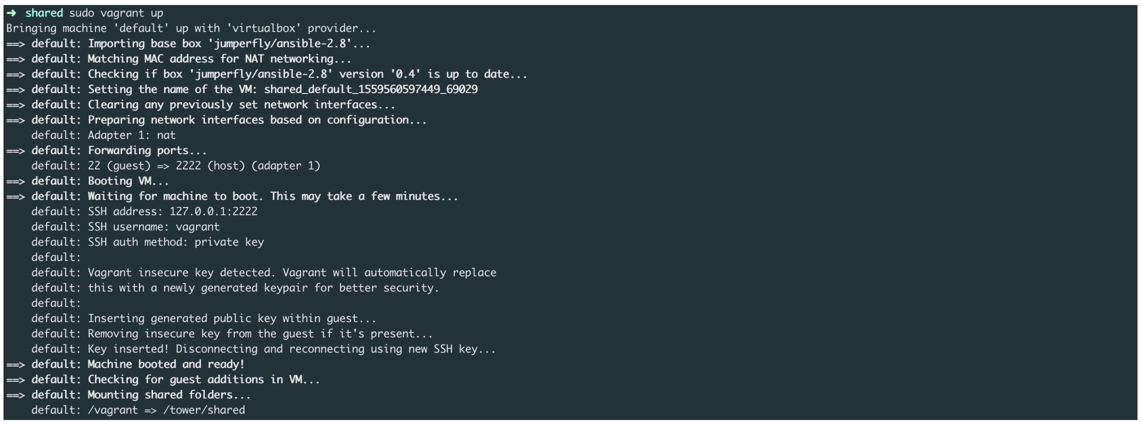
sudo vagrant init centos/7



```
➜  shared sudo vagrant init jumperfly/ansible-2.8
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
```

Launch your VM by issuing the command below:

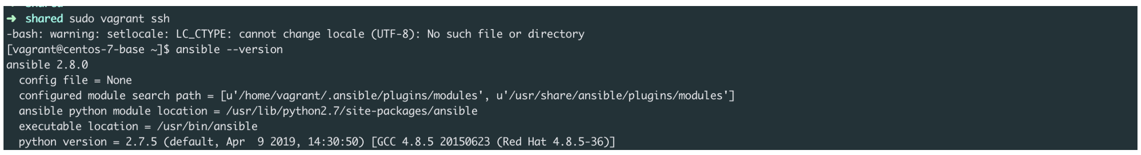
sudo vagrant up



```
➜  shared sudo vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'jumperfly/ansible-2.8'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'jumperfly/ansible-2.8' version '0.4' is up to date...
==> default: Setting the name of the VM: shared_default_1559560597449_69029
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
==> default: Mounting shared folders...
    default: /vagrant => /tower/shared
```

Access the VM initiated. In this case, Ansible is already installed ("ansible --version").

sudo vagrant ssh



```
➜  shared sudo vagrant ssh
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or directory
[vagrant@centos-7-base ~]$ ansible --version
ansible 2.8.0
  config file = None
  configured module search path = [u'/home/vagrant/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/site-packages/ansible
  executable location = /usr/bin/ansible
  python version = 2.7.5 (default, Apr  9 2019, 14:30:50) [GCC 4.8.5 20150623 (Red Hat 4.8.5-36)]
```

Vagrant Basic Commands

Navigate to the folder which stores your Vagrantfile and pause the VM

sudo vagrant suspend

Switch off the VM

```
sudo vagrant halt
```

Destroy the VM

```
sudo vagrant destroy
```

Quick Start

Concepts

Ansible Engine

Inventory

The Ansible inventory file defines the hosts and groups of hosts upon which commands, modules, and tasks in a playbook operate. The file can be in one of many formats depending on your Ansible environment and plugins. The default location for the inventory file is /etc/ansible/hosts. If necessary, you can also create project-specific inventory files in alternate locations.

The inventory file can list individual hosts or user-defined groups of hosts. For example, if you are managing one or more data centers, you can create Ansible groups for those components that require the same set of operations.

YAML

YAML stands for "YAML Ain't Markup Language" (Please refer to: https://yaml.org).

it is basically a human-readable structured data format. It is less complex and ungainly than XML or JSON, but provides similar capabilities.

There are some rules that YAML has in place to avoid issues related to ambiguity. These rules make it possible for a single YAML file to be interpreted consistently, regardless of which library is being used to interpret it.

- YAML files should end in .yaml.
- YAML is case sensitive.
- YAML uses a fixed indentation scheme to represent relationships between data layers.
- Dictionary keys are represented in YAML as strings terminated by a trailing colon. Values are represented by either a string following the colon, separated by a space.
- To represent lists of items, a single dash followed by a space is used. Multiple items are a part of the same list as a function of their having the same level of indentation.

The following example represents a YAML playbook:

```
- name: configure interface settings  ios  config:    lines:      - description
```

Configuration File

Certain settings in Ansible are adjustable via a configuration file (ansible.cfg). The stock configuration should be sufficient for most users, but there may be reasons you would want to change them.

Configuration file which will be processed in the following order:

1. Environment variable: ANSIBLE_CONFIG
2. Actual directory "ansible.cfg" file.
3. Home directory "ansible.cfg" file.
4. Configuration file stored in "/etc/ansible/ansible.cfg".

Option	Default Value	Description
inventory	/etc/ansible/hosts	Inventory location
forks	5	Specify number of parallel processes to use
remote_port	22	Remote SSH port
host_key_checking	true	Check host key installed
timeout	10	SSH connection timeout in seconds
remote_user	root	Remote connection user
become	false	Run operations with become (does not imply password prompting)
become_method	sudo	Privilege escalation method to use
pipelining	false	Reduces the number of network operations required to execute a module on the remote server, by executing many Ansible modules without actual file transfer.This can result in a very significant performance improvement when enabled

Please refer to Ansible documentation for more details: https://docs.ansible.com/ansible/latest/reference_appendices/config.html

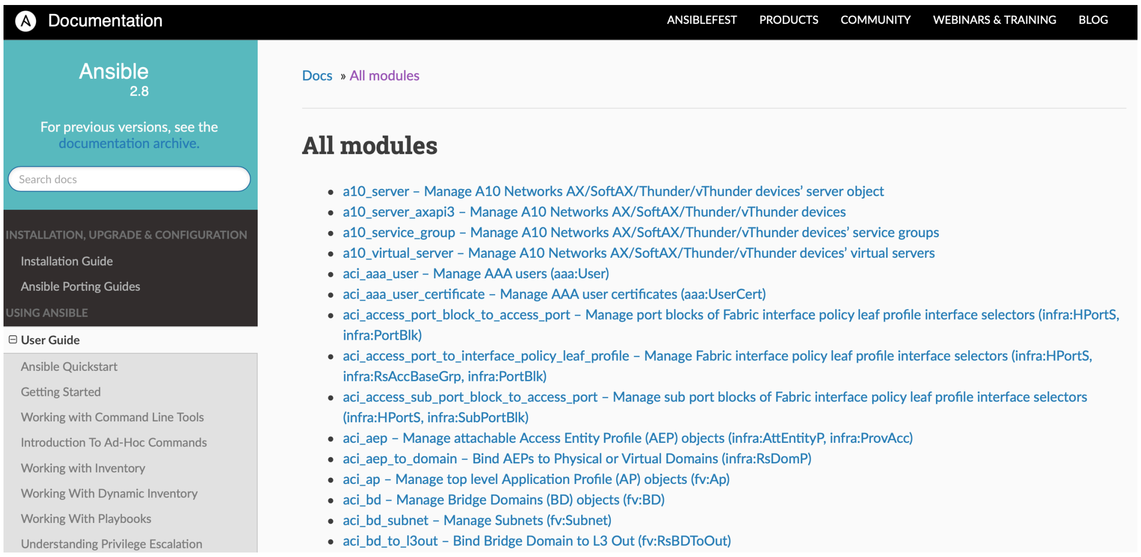
Playbooks

Playbooks are Ansible's configuration, deployment, and orchestration language. Playbooks are designed to be human-readable and are developed in a basic text language. There are multiple ways to organize playbooks and the files they include.

Modules

Ansible ships with a number of modules (called the 'module library') that can be executed directly on remote hosts or through playbooks. Users can also write their own modules. These modules can control system resources, like services, packages, or files , or handle executing system commands.

Ansible modules: https://docs.ansible.com/ansible/latest/modules/list_of_all_modules.html

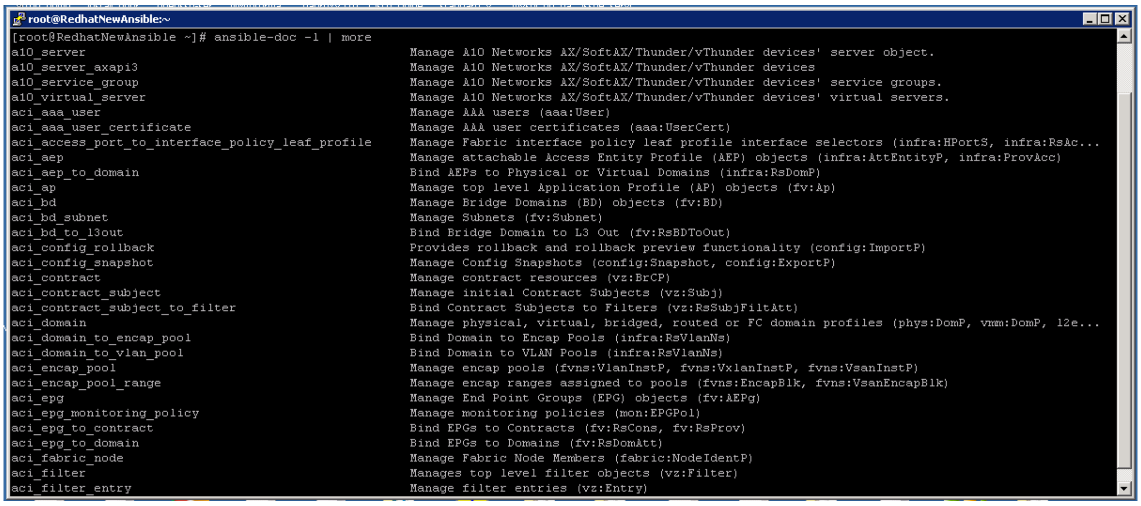


All modules

- a10_server – Manage A10 Networks AX/SoftAX/Thunder/vThunder devices' server object
- a10_server_axapi3 – Manage A10 Networks AX/SoftAX/Thunder/vThunder devices
- a10_service_group – Manage A10 Networks AX/SoftAX/Thunder/vThunder devices' service groups
- a10_virtual_server – Manage A10 Networks AX/SoftAX/Thunder/vThunder devices' virtual servers
- aci_aaa_user – Manage AAA users (aaa:User)
- aci_aaa_user_certificate – Manage AAA user certificates (aaa:UserCert)
- aci_access_port_block_to_access_port – Manage port blocks of Fabric interface policy leaf profile interface selectors (infra:HPortS, infra:PortBlk)
- aci_access_port_to_interface_policy_leaf_profile – Manage Fabric interface policy leaf profile interface selectors (infra:HPortS, infra:RsAccBaseGrp, infra:PortBlk)
- aci_access_sub_port_block_to_access_port – Manage sub port blocks of Fabric interface policy leaf profile interface selectors (infra:HPortS, infra:SubPortBlk)
- aci_aep – Manage attachable Access Entity Profile (AEP) objects (infra:AttEntityP, infra:ProvAcc)
- aci_aep_to_domain – Bind AEPs to Physical or Virtual Domains (infra:RsDomP)
- aci_ap – Manage top level Application Profile (AP) objects (fv:Ap)
- aci_bd – Manage Bridge Domains (BD) objects (fv:BD)
- aci_bd_subnet – Manage Subnets (fv:Subnet)
- aci_bd_to_l3out – Bind Bridge Domain to L3 Out (fv:RsBDToOut)

The following command displays the list of available modules:

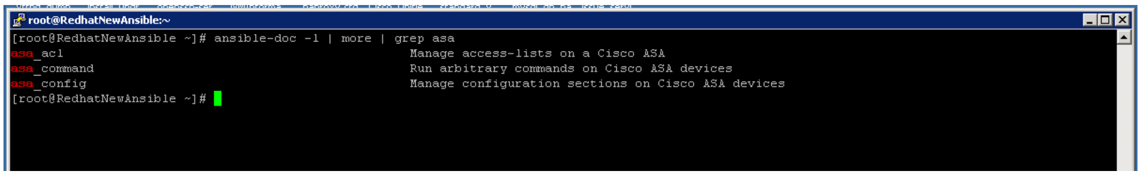
#ansible-doc -l | more



```
[root@RedhatNewAnsible ~]# ansible-doc -l | more
a10_server                                           Manage A10 Networks AX/SoftAX/Thunder/vThunder devices' server object.
a10_server_axapi3                                    Manage A10 Networks AX/SoftAX/Thunder/vThunder devices
a10_service_group                                    Manage A10 Networks AX/SoftAX/Thunder/vThunder devices' service groups.
a10_virtual_server                                   Manage A10 Networks AX/SoftAX/Thunder/vThunder devices' virtual servers.
aci_aaa_user                                         Manage AAA users (aaa:User)
aci_aaa_user_certificate                             Manage AAA user certificates (aaa:UserCert)
aci_access_port_to_interface_policy_leaf_profile     Manage Fabric interface policy leaf profile interface selectors (infra:HPortS, infra:RsAc...
aci_aep                                              Manage attachable Access Entity Profile (AEP) objects (infra:AttEntityP, infra:ProvAcc)
aci_aep_to_domain                                    Bind AEPs to Physical or Virtual Domains (infra:RsDomP)
aci_ap                                               Manage top level Application Profile (AP) objects (fv:Ap)
aci_bd                                               Manage Bridge Domains (BD) objects (fv:BD)
aci_bd_subnet                                        Manage Subnets (fv:Subnet)
aci_bd_to_l3out                                      Bind Bridge Domain to L3 Out (fv:RsBDToOut)
aci_config_rollback                                  Provides rollback and rollback preview functionality (config:ImportP)
aci_config_snapshot                                  Manage Config Snapshots (config:Snapshot, config:ExportP)
aci_contract                                         Manage contract resources (vz:BrCP)
aci_contract_subject                                 Manage initial Contract Subjects (vz:Subj)
aci_contract_subject_to_filter                       Bind Contract Subjects to Filters (vz:RsSubjFiltAtt)
aci_domain                                           Manage physical, virtual, bridged, routed or FC domain profiles (phys:DomP, vmm:DomP, l2e...
aci_domain_to_encap_pool                             Bind Domain to Encap Pools (infra:RsVlanNs)
aci_domain_to_vlan_pool                              Bind Domain to VLAN Pools (infra:RsVlanNs)
aci_encap_pool                                       Manage encap pools (fvns:VlanInstP, fvns:VxlanInstP, fvns:VsanInstP)
aci_encap_pool_range                                 Manage encap ranges assigned to pools (fvns:EncapBlk, fvns:VsanEncapBlk)
aci_epg                                              Manage End Point Groups (EPG) objects (fv:AEPg)
aci_epg_monitoring_policy                            Manage monitoring policies (mon:EPGPol)
aci_epg_to_contract                                  Bind EPGs to Contracts (fv:RsCons, fv:RsProv)
aci_epg_to_domain                                    Bind EPGs to Domains (fv:RsDomAtt)
aci_fabric_node                                      Manage Fabric Node Members (fabric:NodeIdentP)
aci_filter                                           Manages top level filter objects (vz:Filter)
aci_filter_entry                                     Manage filter entries (vz:Entry)
```

Narrow down modules related to ASA devices:

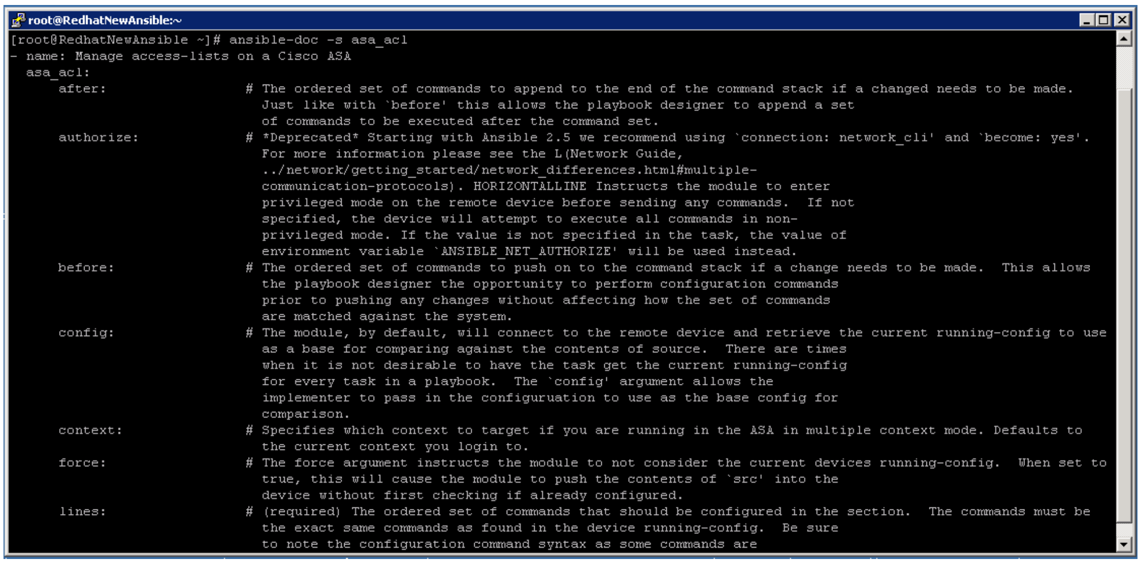
#ansible-doc -l | more | grep asa



```
[root@RedhatNewAnsible ~]# ansible-doc -l | more | grep asa
asa_acl                                              Manage access-lists on a Cisco ASA
asa_command                                          Run arbitrary commands on Cisco ASA devices
asa_config                                           Manage configuration sections on Cisco ASA devices
[root@RedhatNewAnsible ~]#
```

Show actions that a module can perform:

ansible-doc -s module_name



```
[root@RedhatNewAnsible ~]# ansible-doc -s asa_acl
- name: Manage access-lists on a Cisco ASA
  asa_acl:
      after:                 # The ordered set of commands to append to the end of the command stack if a changed needs to be made.
                               Just like with `before' this allows the playbook designer to append a set
                               of commands to be executed after the command set.
      authorize:             # *Deprecated* Starting with Ansible 2.5 we recommend using `connection: network_cli' and `become: yes'.
                               For more information please see the L(Network Guide,
                               ../network/getting_started/network_differences.html#multiple-
                               communication-protocols). HORIZONTALLINE Instructs the module to enter
                               privileged mode on the remote device before sending any commands.  If not
                               specified, the device will attempt to execute all commands in non-
                               privileged mode. If the value is not specified in the task, the value of
                               environment variable `ANSIBLE_NET_AUTHORIZE' will be used instead.
      before:                # The ordered set of commands to push on to the command stack if a change needs to be made.  This allows
                               the playbook designer the opportunity to perform configuration commands
                               prior to pushing any changes without affecting how the set of commands
                               are matched against the system.
      config:                # The module, by default, will connect to the remote device and retrieve the current running-config to use
                               as a base for comparing against the contents of source.  There are times
                               when it is not desirable to have the task get the current running-config
                               for every task in a playbook.  The `config' argument allows the
                               implementer to pass in the configuruation to use as the base config for
                               comparison.
      context:               # Specifies which context to target if you are running in the ASA in multiple context mode. Defaults to
                               the current context you login to.
      force:                 # The force argument instructs the module to not consider the current devices running-config.  When set to
                               true, this will cause the module to push the contents of `src' into the
                               device without first checking if already configured.
      lines:                 # (required) The ordered set of commands that should be configured in the section.  The commands must be
                               the exact same commands as found in the device running-config.  Be sure
                               to note the configuration command syntax as some commands are
```

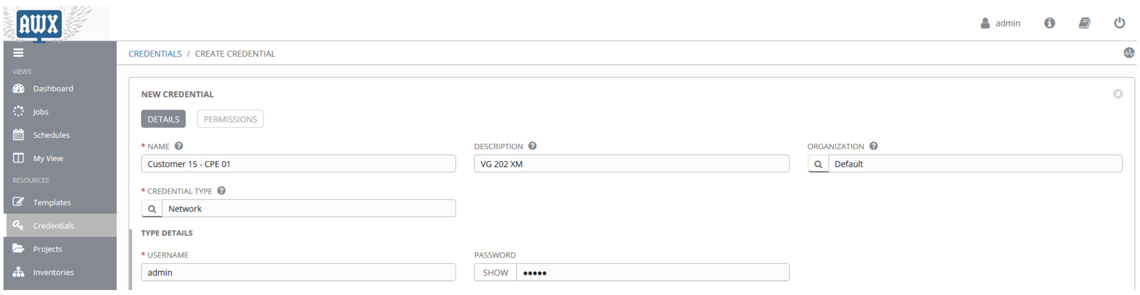
Ansible Tower - AWX

The concepts across Ansible Engine and AWX (Ansible Tower) are the same. However, the GUI provided by AWX makes workflows being configured in a different way.

For running an Ansible Playbook with AWX, you need to configure the following items:

1. Credentials: User name/password or ssh key to connect to remote component.
2. Project: It contains the Ansible playbook, config, roles, templates etc.
3. Inventories: What servers the playbook will run against and connection specific configuration.
4. Templates: Job template to associate all of the above and run the playbook
5. Launch Templates: Launching current project.

For instance, credentials (1) for a CPE device (Credential Type = Network) are created by typing the username and password value pair.



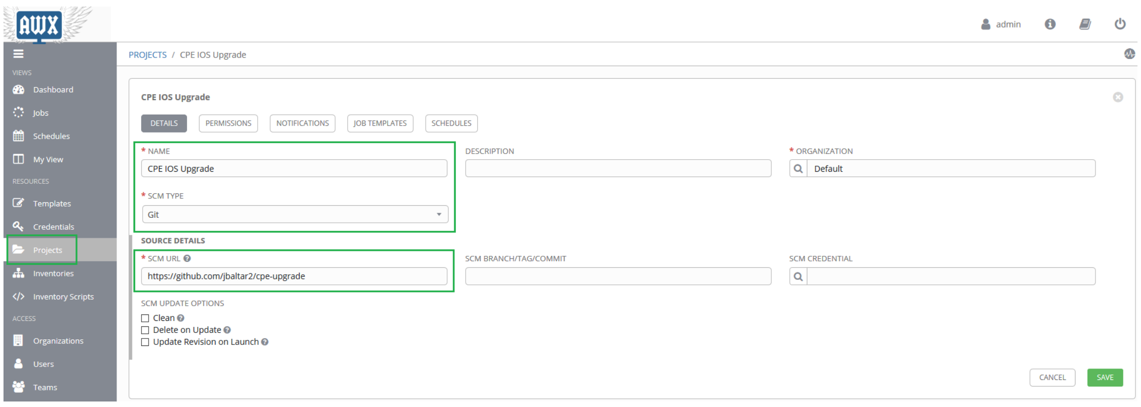
NEW CREDENTIAL

* NAME: Customer 15 - CPE 01
DESCRIPTION: VG 202 XM
ORGANIZATION: Default
* CREDENTIAL TYPE: Network

TYPE DETAILS

* USERNAME: admin
PASSWORD: •••••

Next step requires creating a new Project (2) which is using Git as source.



PROJECTS / CPE IOS Upgrade

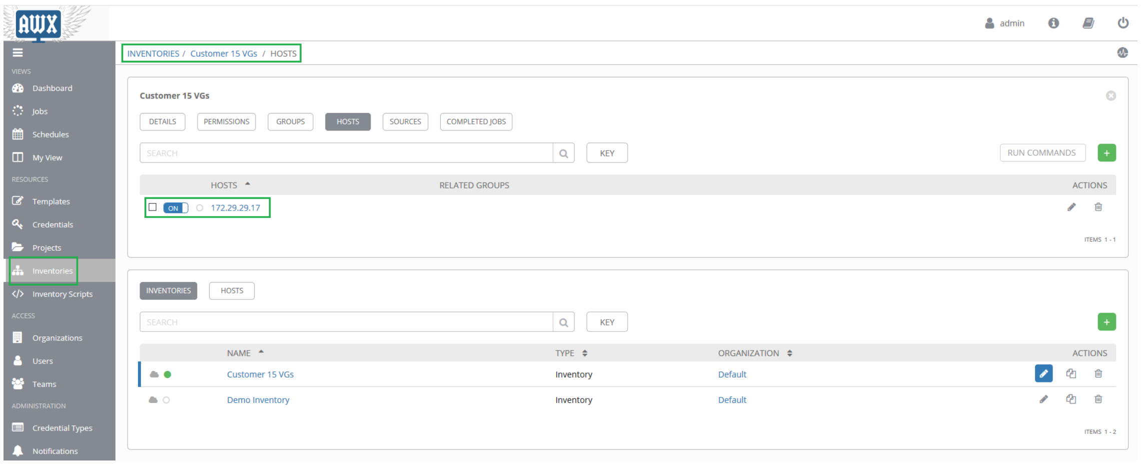
CPE IOS Upgrade

* NAME: CPE IOS Upgrade
DESCRIPTION:
* ORGANIZATION: Default
* SCM TYPE: Git

SOURCE DETAILS
* SCM URL: https://github.com/jbaltar2/cpe-upgrade
SCM BRANCH/TAG/COMMIT:
SCM CREDENTIAL:

SCM UPDATE OPTIONS
- [ ] Clean
- [ ] Delete on Update
- [ ] Update Revision on Launch

Once the Project is saved, click on Inventories (3), create the corresponding one and add Hosts to it as shown below:



INVENTORIES / Customer 15 VGs / HOSTS

Customer 15 VGs

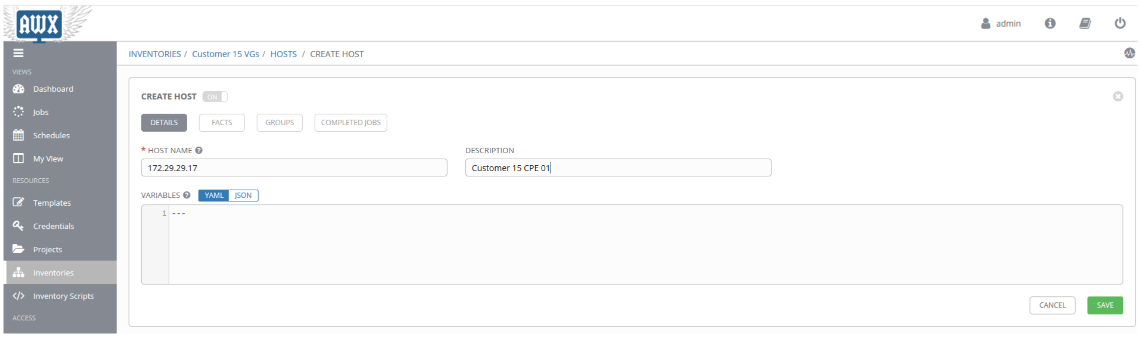
HOSTS	RELATED GROUPS	ACTIONS
ON 172.29.29.17		

ITEMS 1 - 1

NAME	TYPE	ORGANIZATION	ACTIONS
Customer 15 VGs	Inventory	Default	
Demo Inventory	Inventory	Default	

ITEMS 1 - 2

A new Hosts (i.e CPE IP address) is added to the "Customer 15 VGs" inventory.



INVENTORIES / Customer 15 VGs / HOSTS / CREATE HOST

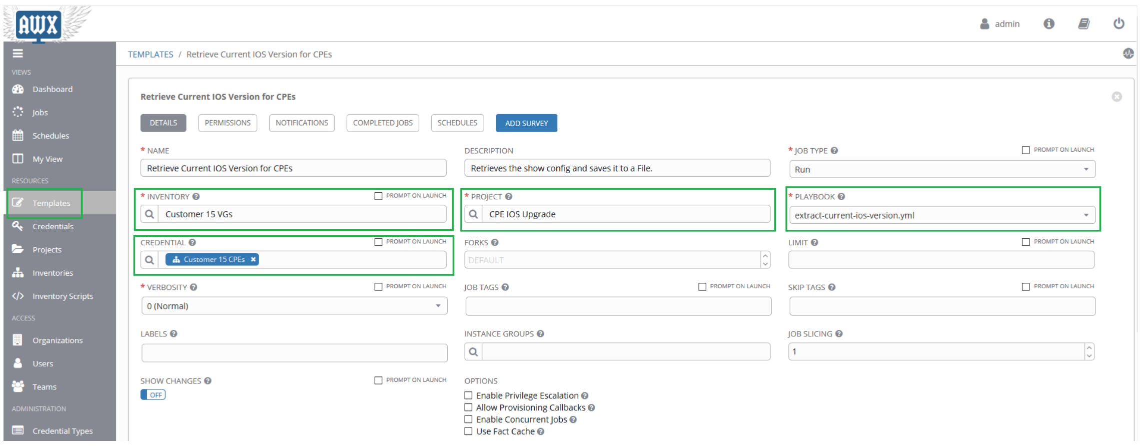
CREATE HOST ON

* HOST NAME: 172.29.29.17
DESCRIPTION: Customer 15 CPE 01

VARIABLES: YAML JSON

```
---
```

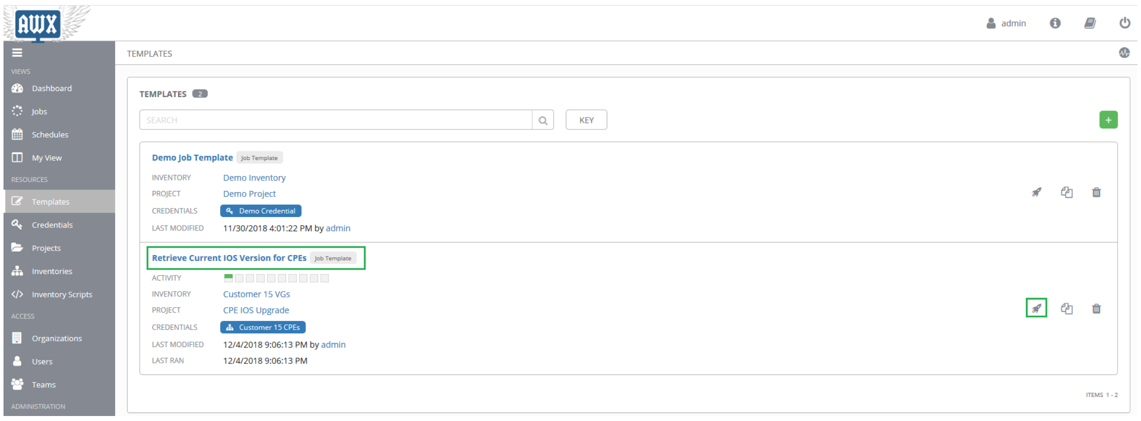
Lets create a Job Template (4) which associated all the above: the inventory, credential and the Project which stores the list of playbooks.



Retrieve Current IOS Version for CPEs

* NAME: Retrieve Current IOS Version for CPEs	DESCRIPTION: Retrieves the show config and saves it to a File.	* JOB TYPE: Run (☐ PROMPT ON LAUNCH)
* INVENTORY: Customer 15 VGs (☐ PROMPT ON LAUNCH)	* PROJECT: CPE IOS Upgrade	* PLAYBOOK: extract-current-ios-version.yml
CREDENTIAL: Customer 15 CPEs (☐ PROMPT ON LAUNCH)	FORKS: DEFAULT	LIMIT: (☐ PROMPT ON LAUNCH)
* VERBOSITY: 0 (Normal) (☐ PROMPT ON LAUNCH)	JOB TAGS: (☐ PROMPT ON LAUNCH)	SKIP TAGS: (☐ PROMPT ON LAUNCH)
LABELS:	INSTANCE GROUPS:	JOB SLICING: 1
SHOW CHANGES: OFF (☐ PROMPT ON LAUNCH)	OPTIONS: ☐ Enable Privilege Escalation ☐ Allow Provisioning Callbacks ☐ Enable Concurrent Jobs ☐ Use Fact Cache	

Finally, click on the Templates tab and launch it.



TEMPLATES 2

Demo Job Template (Job Template)

INVENTORY	Demo Inventory
PROJECT	Demo Project
CREDENTIALS	Demo Credential
LAST MODIFIED	11/30/2018 4:01:22 PM by admin

Retrieve Current IOS Version for CPEs (Job Template)

ACTIVITY	
INVENTORY	Customer 15 VGs
PROJECT	CPE IOS Upgrade
CREDENTIALS	Customer 15 CPEs
LAST MODIFIED	12/4/2018 9:06:13 PM by admin
LAST RAN	12/4/2018 9:06:13 PM

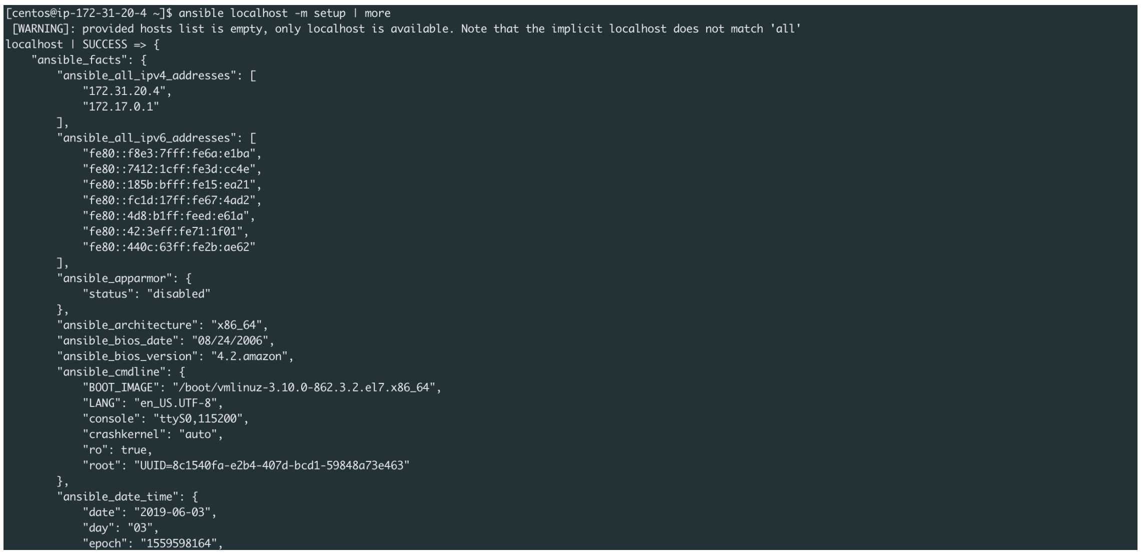
ITEMS 1 - 2

Ad-hoc Commands

An Ansible Ad-hoc command is a one-liner Ansible command that performs a single task on the target host. It allows you to execute simple one-line task against one or group of hosts defined on the inventory file configuration.

An Ad-Hoc command will only have two parameters, the group of a host that you want to perform the task and the Ansible module to run.

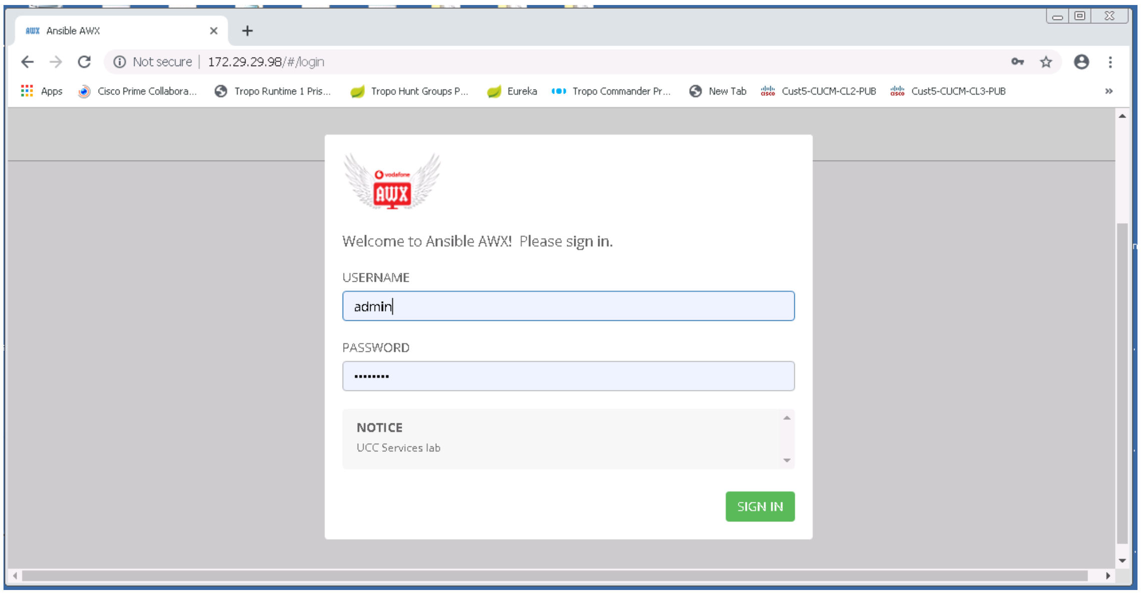
Command	Description
#ansible all -m ping	The basic command of ansible ad-hoc against 'all' hosts on the inventory file and using the 'ping' module
#ansible localhost -m copy -a 'src=/home/myfile dest=/home/mydestinationfolder/myfilecopied'	The command copies a file to a destination folder
#ansible all -m yum -a "name=telnet state=present" -- become	The command installs the "telnet" package against 'all' hosts on the inventory file
#ansible all -m service -a "name=nginx state=started enabled=yes" —-become	The commands starts the nginx service
#ansible localhost -m setup | more	The command retrieves a bunch of system data from the remote host



```
[centos@ip-172-31-20-4 ~]$ ansible localhost -m setup | more
 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
localhost | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "172.31.20.4",
            "172.17.0.1"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::f8e3:7fff:fe6a:e1ba",
            "fe80::7412:1cff:fe3d:cc4e",
            "fe80::185b:bfff:fe15:ea21",
            "fe80::fc1d:17ff:fe67:4ad2",
            "fe80::4d8:b1ff:feed:e61a",
            "fe80::42:3eff:fe71:1f01",
            "fe80::440c:63ff:fe2b:ae62"
        ],
        "ansible_apparmor": {
            "status": "disabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "08/24/2006",
        "ansible_bios_version": "4.2.amazon",
        "ansible_cmdline": {
            "BOOT_IMAGE": "/boot/vmlinuz-3.10.0-862.3.2.el7.x86_64",
            "LANG": "en_US.UTF-8",
            "console": "ttyS0,115200",
            "crashkernel": "auto",
            "ro": true,
            "root": "UUID=8c1540fa-e2b4-407d-bcd1-59848a73e463"
        },
        "ansible_date_time": {
            "date": "2019-06-03",
            "day": "03",
            "epoch": "1559598164",
```

AWX

Browse the corresponding URL as shown below.



Welcome to Ansible AWX! Please sign in.

USERNAME

admin

PASSWORD

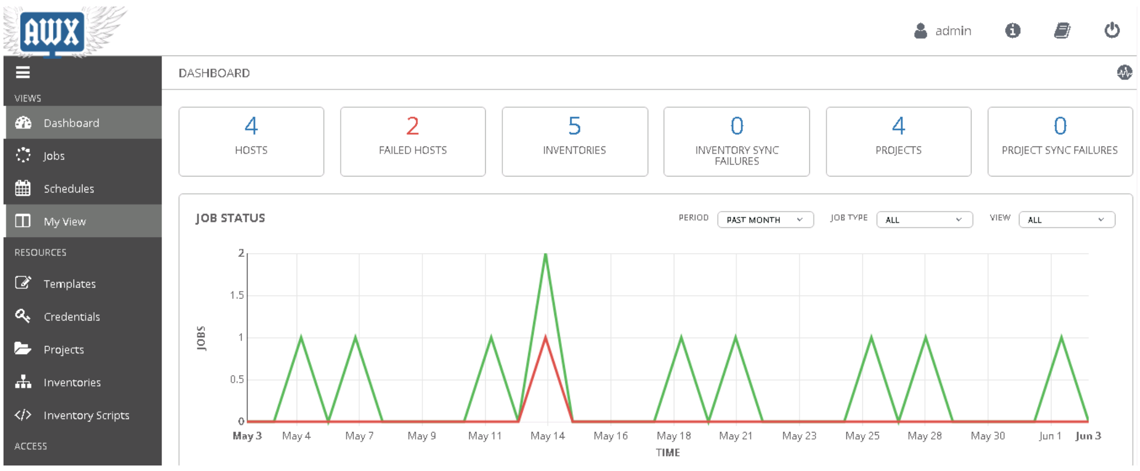
••••••••

NOTICE

UCC Services lab

SIGN IN

Sign in with your credentials and you will be redirected to the AWX dashboard.



4 HOSTS | 2 FAILED HOSTS | 5 INVENTORIES | 0 INVENTORY SYNC FAILURES | 4 PROJECTS | 0 PROJECT SYNC FAILURES

PERIOD: PAST MONTH   JOB TYPE: ALL   VIEW: ALL

Click on Users tab, choose your username and click on Edit.

Edit your password, add you to specific organizations and teams, manage permissions and click on Save.

Ansible Engine

You can access Ansible Engine via SSH using any SSH client.

As a brief example, the /home/AnsibleTraining/ folder contains a inventory file called “hosts”.

The /home/AnsibleTraining/playbooks/ folder stores the playbook below:

```
[root@RedhatNewAnsible playbooks]# cat asa_show_version.yml
---

- hosts: asa-lab
  gather_facts: false
  connection: network_cli

tasks:
- name: SHOW ASA VERSION
   asa_command:
   commands:
      - show version
   register: version

- debug: var=version.stdout_lines
```

Let´s analize the anatomy of the playbook above:

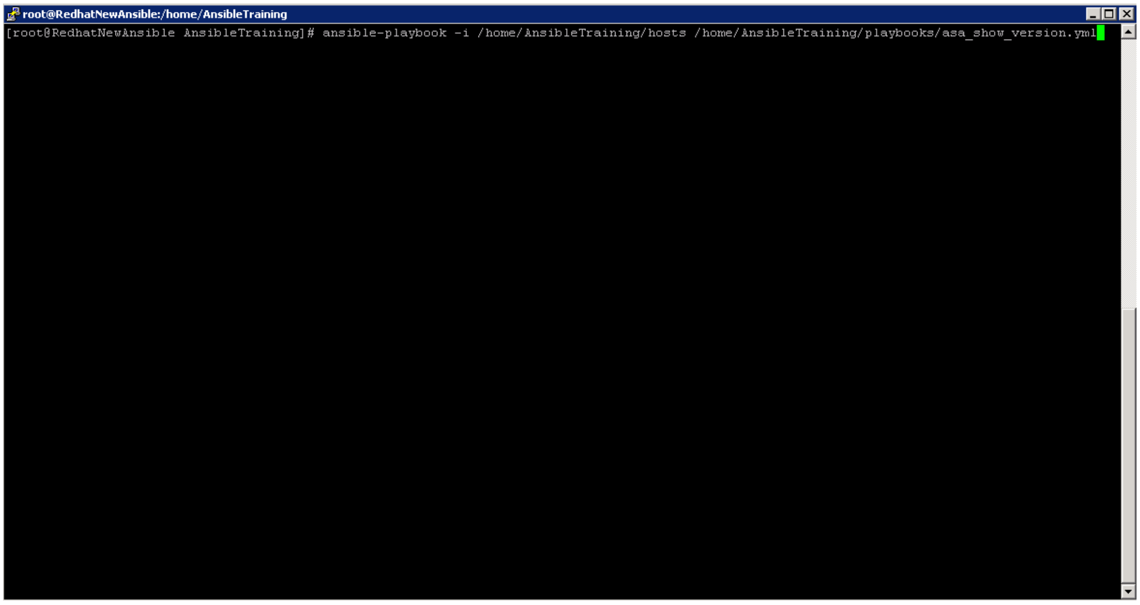
- hosts: Instructs the playbook to be run againsts the "asa-lab" group in the inventory file.
- gather_facts will connect to the managed host, run a script that collects a bunch of data (system, version, environment variables, etc.).
- Ansible uses the "connection" setting to determine how to connect to a remote device. When working with Ansible Networking, set this to network_cli so Ansible treats the remote

node as a network device with a limited execution environment. Without this setting, Ansible would attempt to use ssh to connect to the remote and execute the Python script on the network device, which would fail because Python generally isn't available on network devices.

- The tasks section includes two modules: "asa command" and "debug".
- The "asa_command" module is called in order to issue a "show version" command and store the fetched data in a register.
- "The "debug" commands prints the register output.

The following command runs the mentioned playbook:

#ansible-playbook -i /home/AnsibleTraining/hosts /home/AnsibleTraining/ playbooks/asa_show_version.yml


root@RedhatNewAnsible:/home/AnsibleTraining

```
[root@RedhatNewAnsible AnsibleTraining]# ansible-playbook -i /home/AnsibleTraining/hosts /home/AnsibleTraining/playbooks/asa_show_version.yml
```

Inventories

The inventory file defines the hosts and groups of hosts upon which commands, modules, and tasks in a playbook operate. The inventory file can be in one of many formats depending on your Ansible environment and plugins.

The default location for the inventory file is /etc/ansible/hosts. If necessary, you can also create project-specific inventory files in alternate locations.

The inventory file can list individual hosts or user-defined groups of hosts. This enables you to define groups of devices with similar roles upon which to perform the same operational and configuration tasks. For example, if you are managing one or more data centers, you can create Ansible groups for those elements that require the same set of operations.

Ansible Engine

The following INI-formatted sample inventory file defines an individual host, called "webserver", and two groups of devices, ios and nxos.

$ cat /etc/ansible/hosts

The inventory file includes variables also:

```
[cisco-ios-devices:vars]
ansible_port=22
ansible_user=netadmin
```

Groups of groups are also supported:

```
[cisco-ios-devices]
10.10.0.1
10.10.0.2

[cisco-nxos-devices]
10.10.20.1
10.10.20.2

[network-devices:children]
cisco-ios-devices
cisco-nxos-devices
```

Patterns can be used as shown below:

```
[cisco-nexus-7000]
nexus[01:04].companydomain.com
```

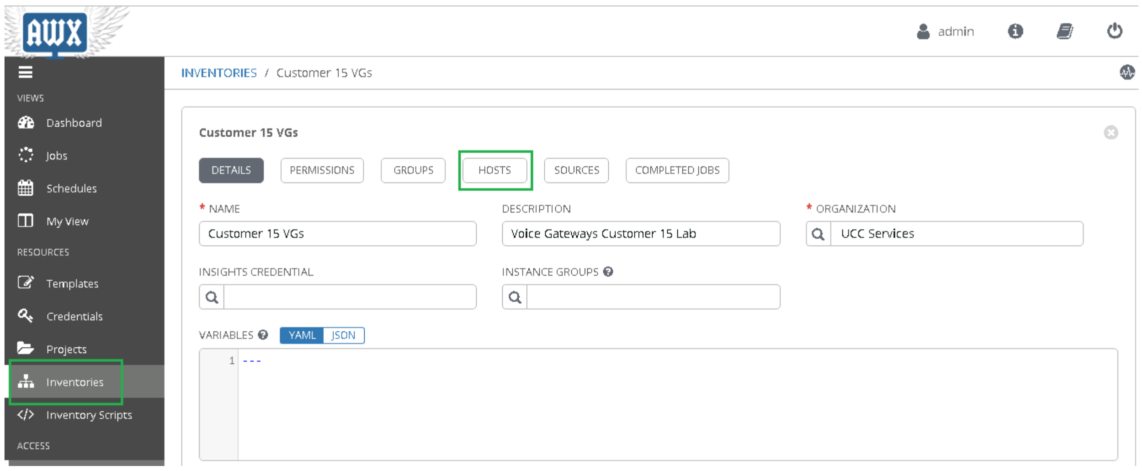
The pattern above corresponds with the following bunch of hosts:

```
nexus01.companydomain.com
nexus02.companydomain.com
nexus03.companydomain.com
nexus04.companydomain.com
```

AWX

AWX stores the inventory in the backend database. To create an inventory list, use the option “Inventories” and click Add and name the Inventory list and save. Click the Add Host button to add a host, the host name can be a DNS resolvable name or an IP address.

Create a new inventory by navigating to Inventories tab and clicking on ⊕.

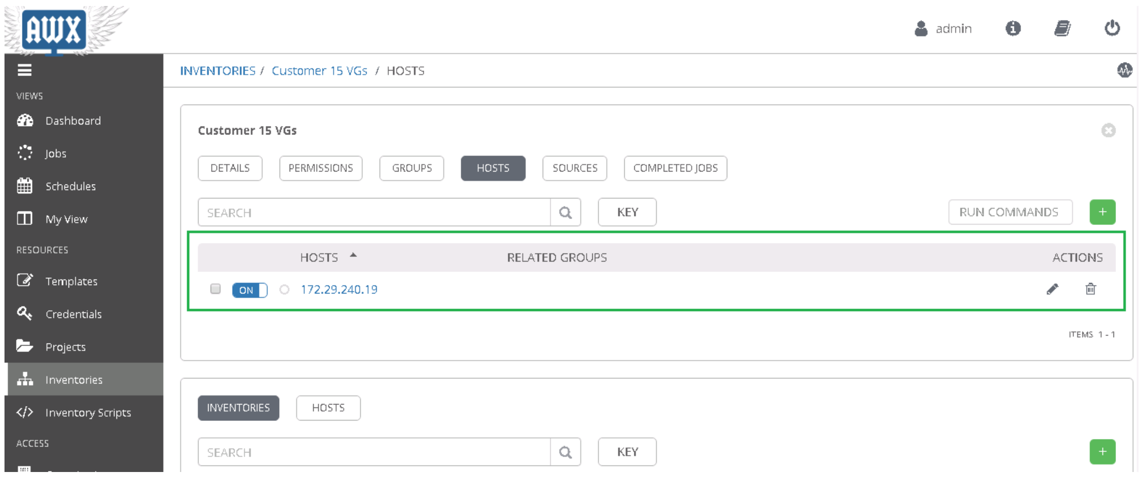


Customer 15 VGs

* NAME: Customer 15 VGs
DESCRIPTION: Voice Gateways Customer 15 Lab
* ORGANIZATION: UCC Services
INSIGHTS CREDENTIAL:
INSTANCE GROUPS:
VARIABLES:

```
---
```

Once the new inventory is saved , click on the “Hosts” tab and click ➕ to add new hosts.



HOSTS	RELATED GROUPS	ACTIONS
ON 172.29.240.19		

Bulk Import

AWX offers a utility called “awx-manage” to perform several of operations via CLI . One of the most important features is bulk hosts import.

Login to the awx task container and execute the following command to import hosts to inventory:

You can list the containers that are running by issuing the following command:

```
# docker ps

CONTAINER ID        IMAGE
COMMAND                          CREATED             STATUS
PORTS                                                          NAMES

28143772a881        ansible/awx_task:2.1.2             "/tini -- /bin/
sh ..."             19 minutes ago       Up 19 minutes          8052/
tcp                                                    awx_task

ded05efa006d        ansible/awx_web:2.1.2              "/tini -- /bin/
sh ..."             19 minutes ago       Up 19 minutes          0.0.0.0:80->8052/
tcp                                     awx_web
```

Use the command "docker exec -it <container ID> /bin/bash" to get a bash shell in the container:

```
[root@RedhatNewAnsible projects]# docker exec -it ded05efa006d  /bin/bash
[root@awxweb awx]# ls
awxfifo  favicon.ico  projects  public  supervisord.log  supervisord.pid  venv  wsgi.py
[root@awxweb awx]# exit
exit
[root@RedhatNewAnsible projects]#
```

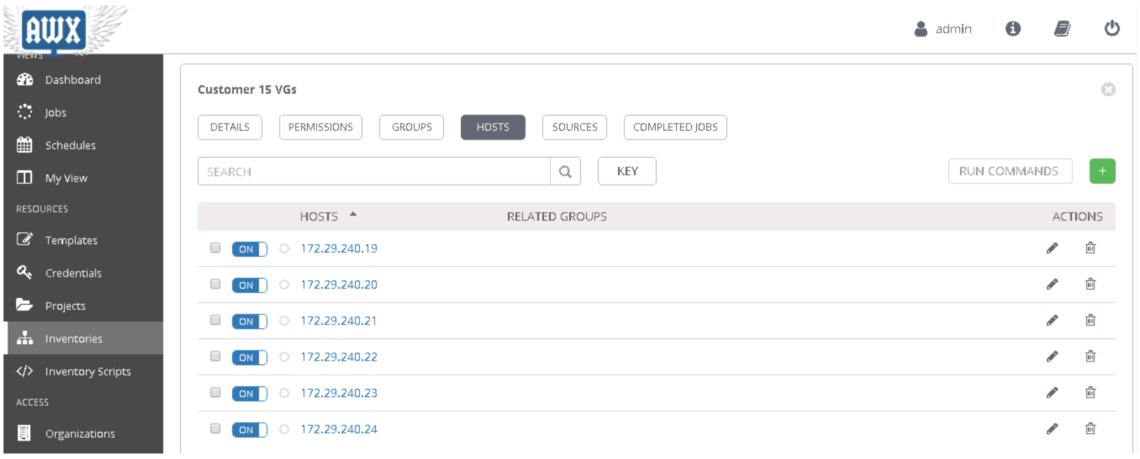
Create a file which contains the list of hosts to add:

```
[root@awx awx]# more customer_15_vgs
172.29.240.20
172.29.240.21
172.29.240.22
172.29.240.23
172.29.240.24
```

Once the list is ready in the file, identify the inventory name in which you need to add the hosts from AWX GUI. In the example, I am adding the hosts to “Customer 15 VGs” inventory.

```
[root@awx awx]# awx-manage inventory_import --inventory-name 'Customer 15 VGs' --source customer_15_vgs
    2.326 INFO     Updating inventory 2: Customer 15 VGs
    2.496 INFO     Reading Ansible inventory source: /var/lib/awx/customer_15_vgs
    3.532 INFO     Processing JSON output...
    3.533 INFO     Loaded 0 groups, 5 hosts
2018-12-17 21:57:24,015 DEBUG    awx.main.models.inventory Going to update inventory computed fields
2018-12-17 21:57:24,114 DEBUG    awx.main.models.inventory Finished updating inventory computed fields
    3.702 INFO     Inventory import completed for  (Customer 15 VGs - 12) in 1.4s
```

The imported hosts are displayed now in AWX GUI.



Customer 15 VGs

HOSTS	RELATED GROUPS	ACTIONS
ON 172.29.240.19		
ON 172.29.240.20		
ON 172.29.240.21		
ON 172.29.240.22		
ON 172.29.240.23		
ON 172.29.240.24		

Dynamic Inventory

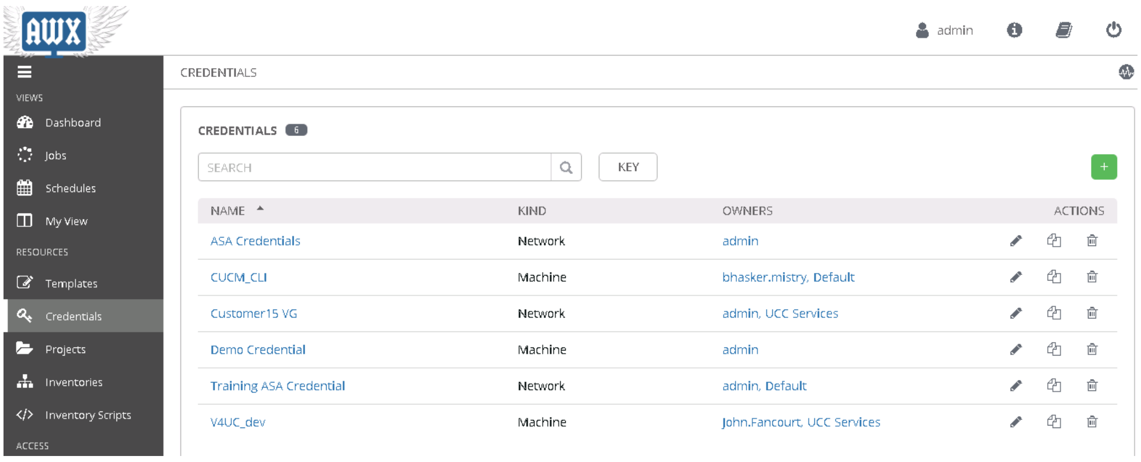
Please refer to the Ansible documentation: https://docs.ansible.com/ansible/latest/user_guide/intro_dynamic_inventory.html

vmware

https://docs.ansible.com/ansible/latest/scenario_guides/vmware_scenarios/vmware_inventory.html

Credentials

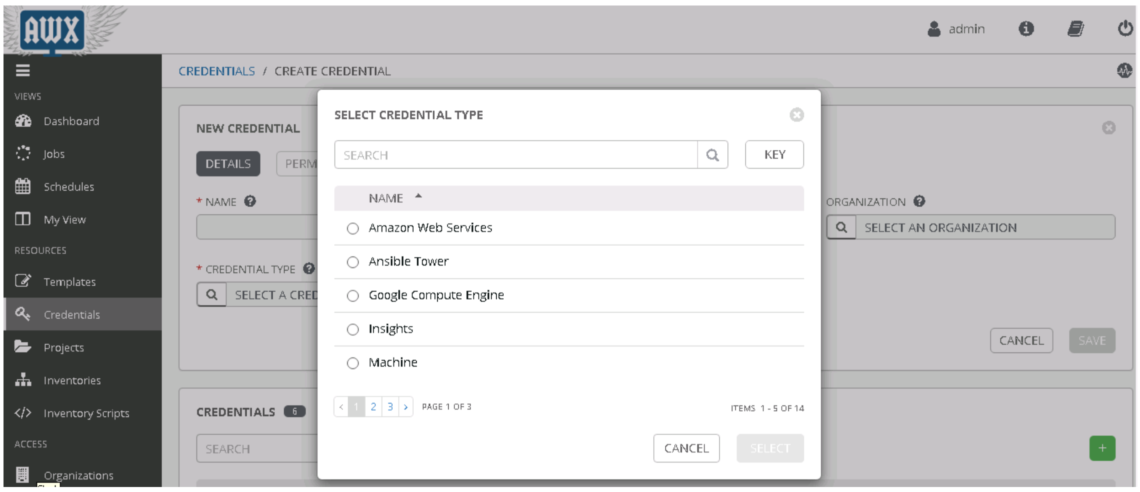
For AWX, click on Credentials tab and add button .



CREDENTIALS 6

NAME	KIND	OWNERS	ACTIONS
ASA Credentials	Network	admin	
CUCM_CLI	Machine	bhasker.mistry, Default	
Customer15 VG	Network	admin, UCC Services	
Demo Credential	Machine	admin	
Training ASA Credential	Network	admin, Default	
V4UC_dev	Machine	John.Fancourt, UCC Services	

A bunch of credentials type are supported such as AWS, Machine, Google Cloud, Azure, Network, VMware, Vault, etc.



Playbooks

Playbooks are the heart of Ansible. They can contain other playbooks, roles and/or tasks.

```
---
- hosts: webservers
  become: yes
  tasks:
    - name: Install Apache
      apt:
        name: apache2
        state: latest
      notify: restart apache

    - name: Create an index
       file:
        state: touch
        path: /var/www/html/index.html

    - name: Add line to index if not present
      lineinfile:
        state: present
        path: /var/www/html/index.html
        line: '<html>My hostname is {{ansible_hostname}}</html>'

  handlers:
```

Let’s review the code line by line.

```
---
```

This line starts every playbook. The three dashes tell the interpreter that this is a YAML document.

```
- hosts: webservers
```

At this point starts the first play in the playbook. Each play is defined in a YAML as a list entry and has to have at least two keys, hosts and tasks (or roles).

These entries can be in any order as long as they are present. The hosts entry can either be a individual host in the inventory or an inventory group.

```
become: yes
```

This line will have the playbook run as root.

```
tasks:
  - name: Install Apache
    apt:
      name: apache2
      state: latest
    notify: restart apache
```

The tasks section contains all of the tasks that will run on the hosts defined earlier. Each separate task is a list entry under this section and contain one or more modules.

```
- name: Add line to index if not present
  lineinfile:
    state: present
    path: /var/www/html/index.html
    line: '<html>My hostname is {{ansible_hostname}}</html>'
```

This is an example of a variable ({{ansible_hostname}}). Variables can be set by a variety of sources including the command line and the inventory file.

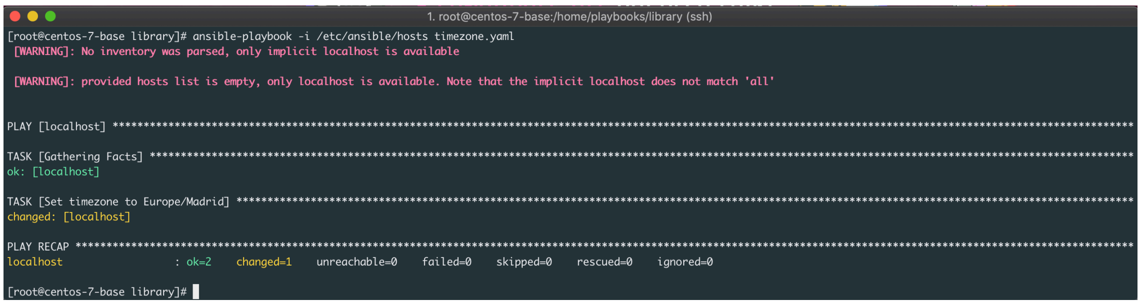
This particular variable will be set when Ansible runs through the fact gathering phase of the host, which provides the playbook run with a wide range of information that you can use in your playbooks.

Finally, the playbook uses handlers. The line "notify: restart apache" in the first task triggers this handler if there was a change that was required to be made by that task. This means if Apache wasn't installed or a new version was found the handler will be run and apache will be restarted.

Running Playbooks Deep Dive

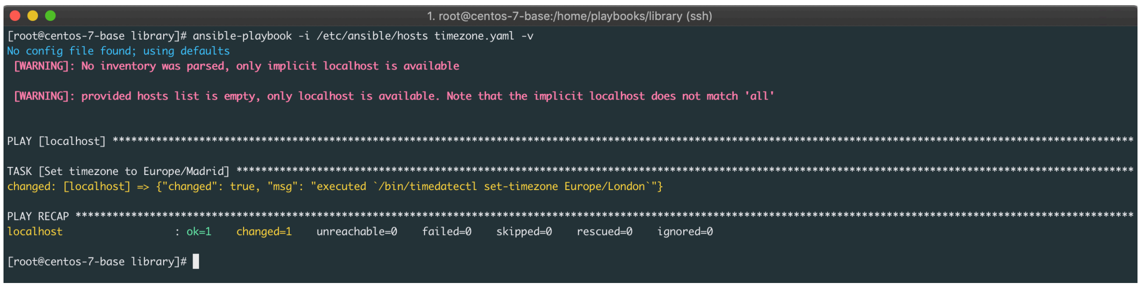
Ansible Engine

#ansible-playbook -i /path/inventory_file /path/playbook.yaml

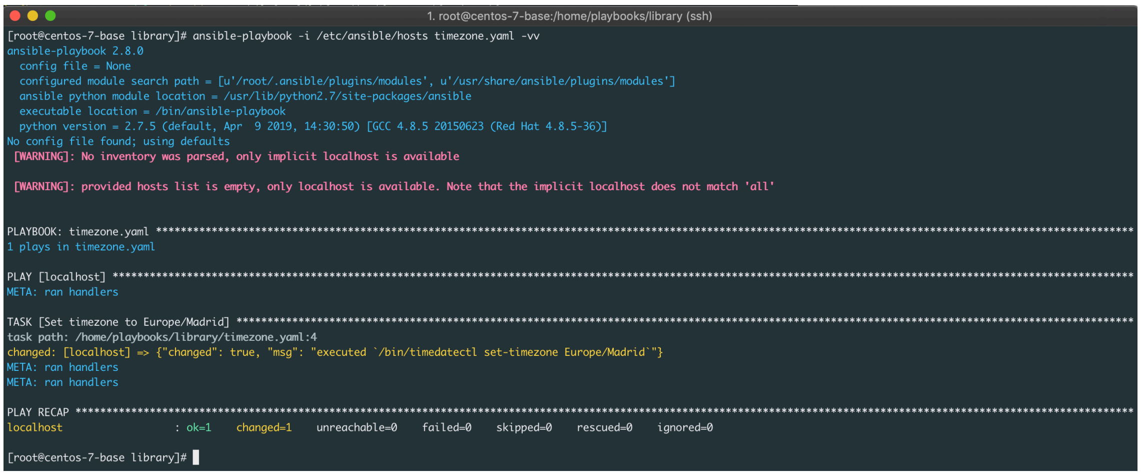


```
[root@centos-7-base library]# ansible-playbook -i /etc/ansible/hosts timezone.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ******************************************************************************************************************************************

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

TASK [Set timezone to Europe/Madrid] **********************************************************************************************************************
changed: [localhost]

PLAY RECAP ************************************************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@centos-7-base library]#
```

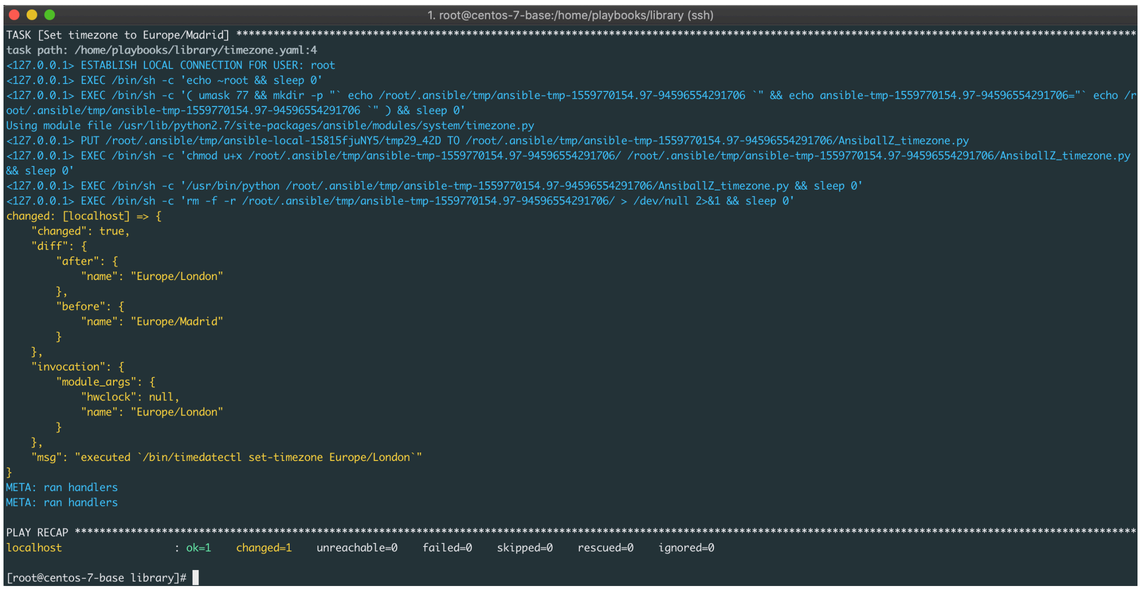
One of the first options anyone picks up is the debug option. To understand what is happening when you run the playbook, you can run it with the verbose (-v) option. Every extra v will provide the end user with more debug output.



```
[root@centos-7-base library]# ansible-playbook -i /etc/ansible/hosts timezone.yaml -v
No config file found; using defaults
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ******************************************************************************************************************************************

TASK [Set timezone to Europe/Madrid] **********************************************************************************************************************
changed: [localhost] => {"changed": true, "msg": "executed `/bin/timedatectl set-timezone Europe/London`"}

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

[root@centos-7-base library]#
```

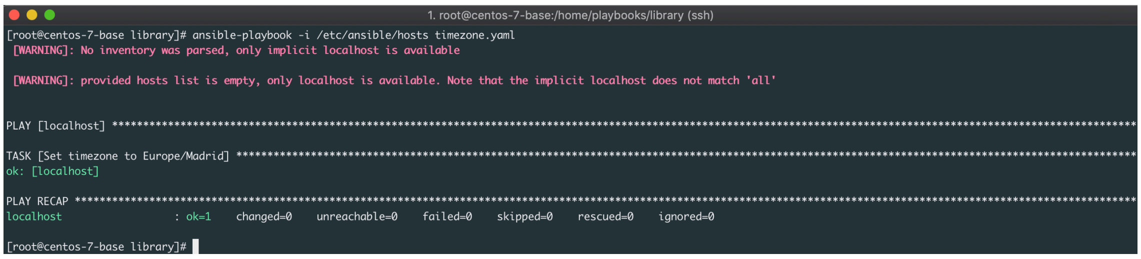


```
[root@centos-7-base library]# ansible-playbook -i /etc/ansible/hosts timezone.yaml -vv
ansible-playbook 2.8.0
  config file = None
  configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/site-packages/ansible
  executable location = /bin/ansible-playbook
  python version = 2.7.5 (default, Apr  9 2019, 14:30:50) [GCC 4.8.5 20150623 (Red Hat 4.8.5-36)]
No config file found; using defaults
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAYBOOK: timezone.yaml *************************************************************************************************************
1 plays in timezone.yaml

PLAY [localhost] ********************************************************************************************************************
META: ran handlers

TASK [Set timezone to Europe/Madrid] ************************************************************************************************
task path: /home/playbooks/library/timezone.yaml:4
changed: [localhost] => {"changed": true, "msg": "executed `/bin/timedatectl set-timezone Europe/Madrid`"}
META: ran handlers
META: ran handlers

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

[root@centos-7-base library]# 
```



```
TASK [Set timezone to Europe/Madrid] ************************************************************************************************
task path: /home/playbooks/library/timezone.yaml:4
<127.0.0.1> ESTABLISH LOCAL CONNECTION FOR USER: root
<127.0.0.1> EXEC /bin/sh -c 'echo ~root && sleep 0'
<127.0.0.1> EXEC /bin/sh -c '( umask 77 && mkdir -p "` echo /root/.ansible/tmp/ansible-tmp-1559770154.97-94596554291706 `" && echo ansible-tmp-1559770154.97-94596554291706="` echo /r
oot/.ansible/tmp/ansible-tmp-1559770154.97-94596554291706 `" ) && sleep 0'
Using module file /usr/lib/python2.7/site-packages/ansible/modules/system/timezone.py
<127.0.0.1> PUT /root/.ansible/tmp/ansible-local-15815fjuNY5/tmp29_42D TO /root/.ansible/tmp/ansible-tmp-1559770154.97-94596554291706/AnsiballZ_timezone.py
<127.0.0.1> EXEC /bin/sh -c 'chmod u+x /root/.ansible/tmp/ansible-tmp-1559770154.97-94596554291706/ /root/.ansible/tmp/ansible-tmp-1559770154.97-94596554291706/AnsiballZ_timezone.py
&& sleep 0'
<127.0.0.1> EXEC /bin/sh -c '/usr/bin/python /root/.ansible/tmp/ansible-tmp-1559770154.97-94596554291706/AnsiballZ_timezone.py && sleep 0'
<127.0.0.1> EXEC /bin/sh -c 'rm -f -r /root/.ansible/tmp/ansible-tmp-1559770154.97-94596554291706/ > /dev/null 2>&1 && sleep 0'
changed: [localhost] => {
    "changed": true,
    "diff": {
        "after": {
            "name": "Europe/London"
        },
        "before": {
            "name": "Europe/Madrid"
        }
    },
    "invocation": {
        "module_args": {
            "hwclock": null,
            "name": "Europe/London"
        }
    },
    "msg": "executed `/bin/timedatectl set-timezone Europe/London`"
}
META: ran handlers
META: ran handlers

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

[root@centos-7-base library]# 
```

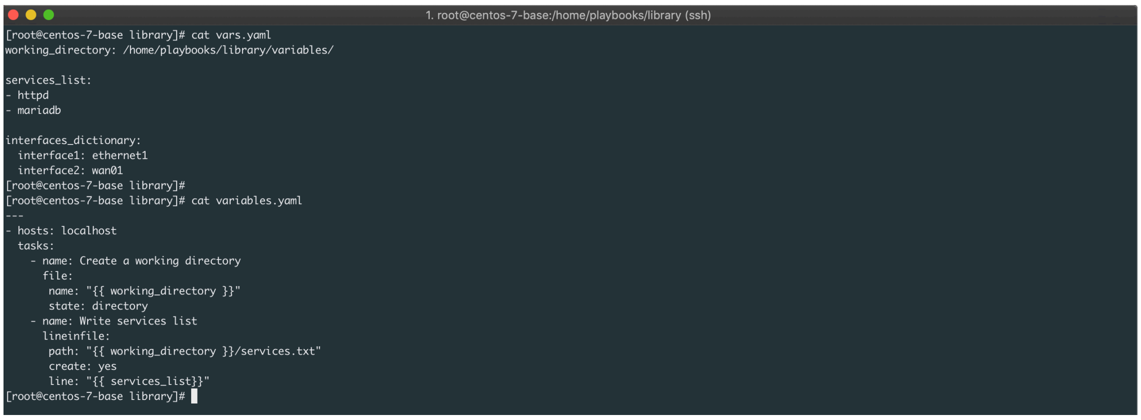
If you run the playbook again, the execution is successful but no changes are made (changed=0).



```
[root@centos-7-base library]# ansible-playbook -i /etc/ansible/hosts timezone.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ********************************************************************************************************************

TASK [Set timezone to Europe/Madrid] ************************************************************************************************
ok: [localhost]

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

[root@centos-7-base library]# 
```

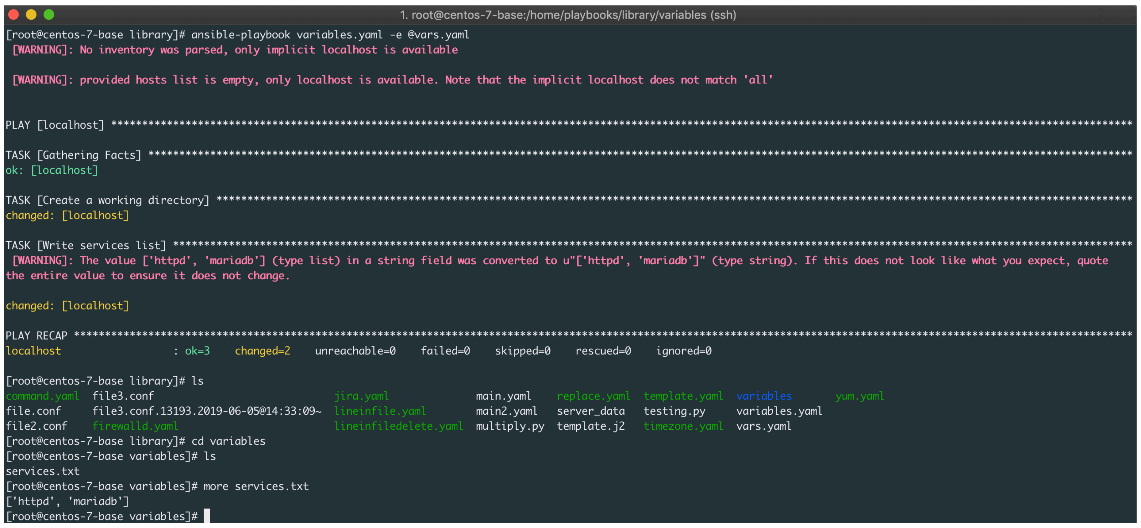
Variables

You define variable in files as mentioned above when using ansible-vault and in the vars section of a play, where you are defining the variable for the set of hosts in the play.



```
[root@centos-7-base library]# cat vars.yaml
working_directory: /home/playbooks/library/variables/

services_list:
- httpd
- mariadb

interfaces_dictionary:
  interface1: ethernet1
  interface2: wan01
[root@centos-7-base library]#
[root@centos-7-base library]# cat variables.yaml
---
- hosts: localhost
  tasks:
    - name: Create a working directory
      file:
        name: "{{ working_directory }}"
        state: directory
    - name: Write services list
      lineinfile:
        path: "{{ working_directory }}/services.txt"
        create: yes
        line: "{{ services_list}}"
[root@centos-7-base library]#
```

This playbooks just fetches the variables from the file and displays the output.



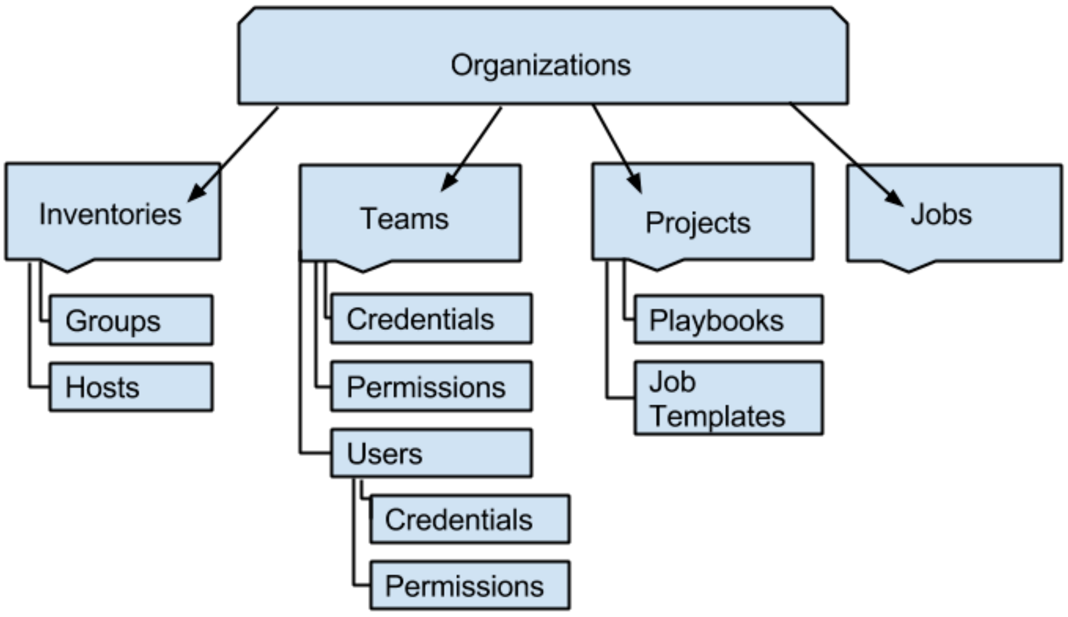
```
[root@centos-7-base library]# ansible-playbook variables.yaml -e @vars.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ***********************************************************************************************************************

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

TASK [Create a working directory] ******************************************************************************************************
changed: [localhost]

TASK [Write services list] *************************************************************************************************************
 [WARNING]: The value ['httpd', 'mariadb'] (type list) in a string field was converted to u"['httpd', 'mariadb']" (type string). If this does not look like what you expect, quote
the entire value to ensure it does not change.

changed: [localhost]

PLAY RECAP *****************************************************************************************************************************
localhost                  : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@centos-7-base library]# ls
command.yaml  file3.conf                          jira.yaml              main.yaml    replace.yaml  template.yaml  variables       yum.yaml
file.conf     file3.conf.13193.2019-06-05@14:33:09~  lineinfile.yaml        main2.yaml   server_data   testing.py     variables.yaml
file2.conf    firewalld.yaml                      lineinfiledelete.yaml  multiply.py  template.j2   timezone.yaml  vars.yaml
[root@centos-7-base library]# cd variables
[root@centos-7-base variables]# ls
services.txt
[root@centos-7-base variables]# more services.txt
['httpd', 'mariadb']
[root@centos-7-base variables]#
```

AWX

Let´s introduce some concepts before getting deeper on how to run playbooks with AWX.

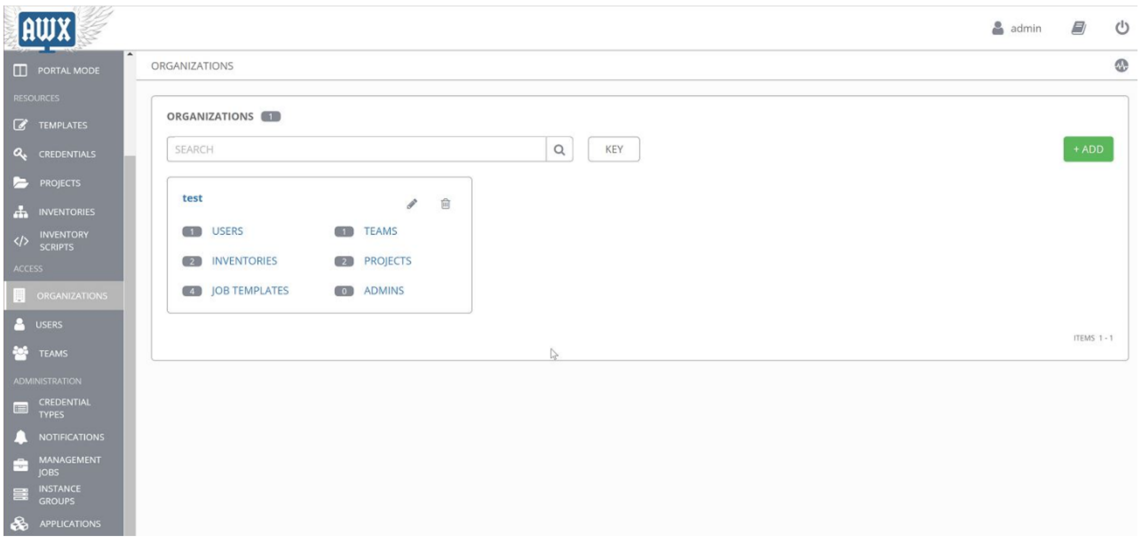
Organizations

Ansible AWX offers multilevel access delegation and role-based access control to the organization. An organization is a logical collection of Users, Teams, Projects, and Inventories. It is the highest level in the AWX object hierarchy. To support multiple clients from one umbrella, you could create an organization for each client and manage multiple teams under that. The picture below depicts the AWX organization hierarchy:

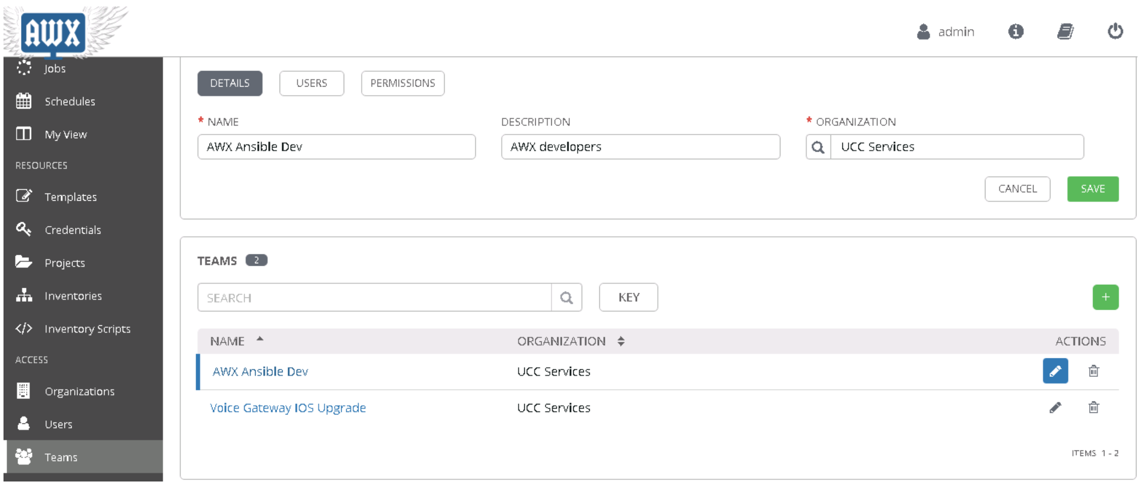


In order to create a new organization, login to AWX console with admin privileges.

Select “Organization” from the navigation and click on add . Enter the organization name and click on Save. Selecting the instance group (requires clustering) might be useful on large deployment.



In order to create a new team, navigate to “Teams” from the menu. Click on to add a new team. Enter the team name and select the organization (you can add multiple teams based on the requirements).



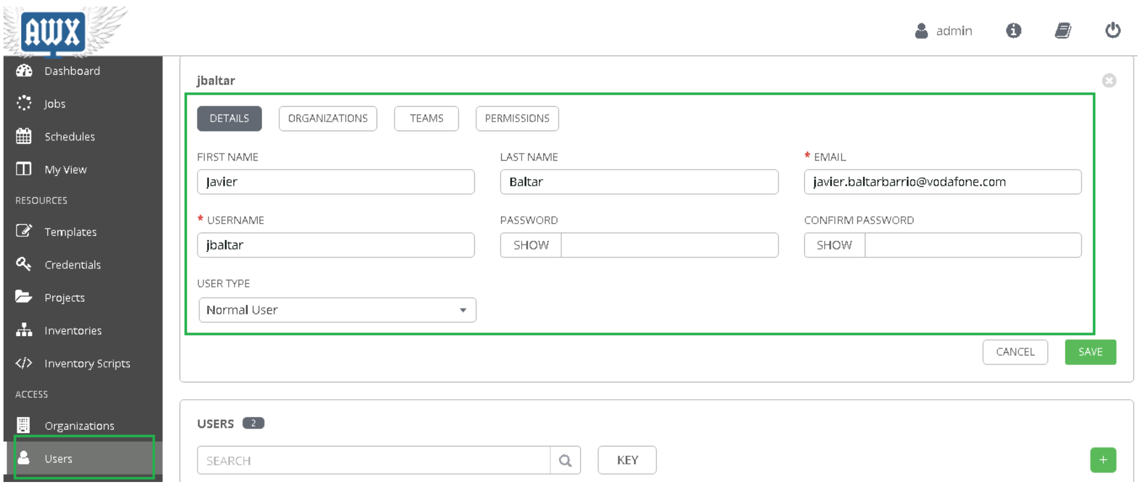
* NAME	DESCRIPTION	* ORGANIZATION
AWX Ansible Dev	AWX developers	UCC Services

TEAMS 2

NAME	ORGANIZATION	ACTIONS
AWX Ansible Dev	UCC Services	
Voice Gateway IOS Upgrade	UCC Services	

User Management

From the AWX main menu,you can aad users to current organization. Select “users” and click on ➕. Enter the user details and select the destination organization.



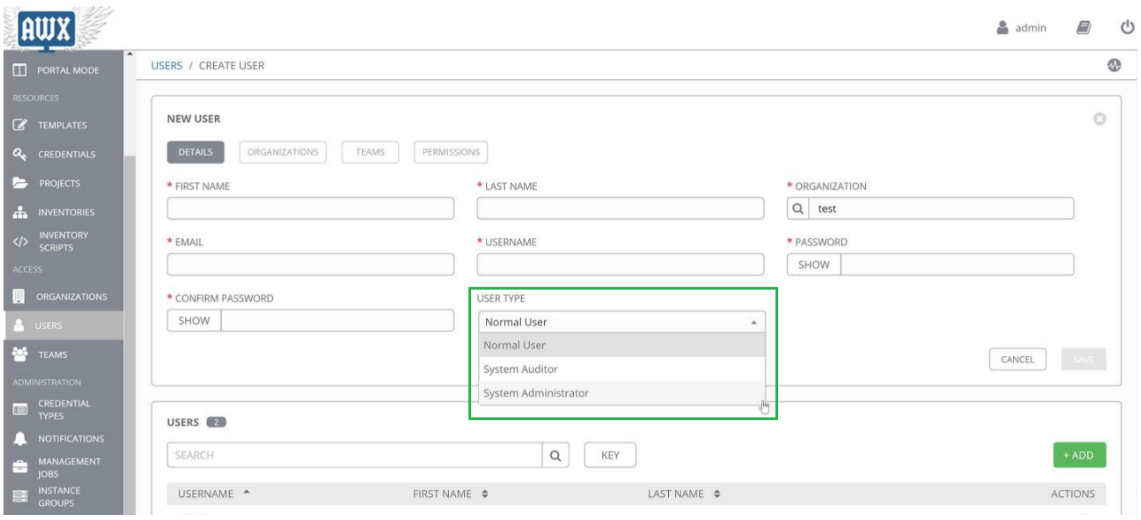
jbaltar

FIRST NAME	LAST NAME	* EMAIL
Javier	Baltar	javier.baltarbarrio@vodafone.com

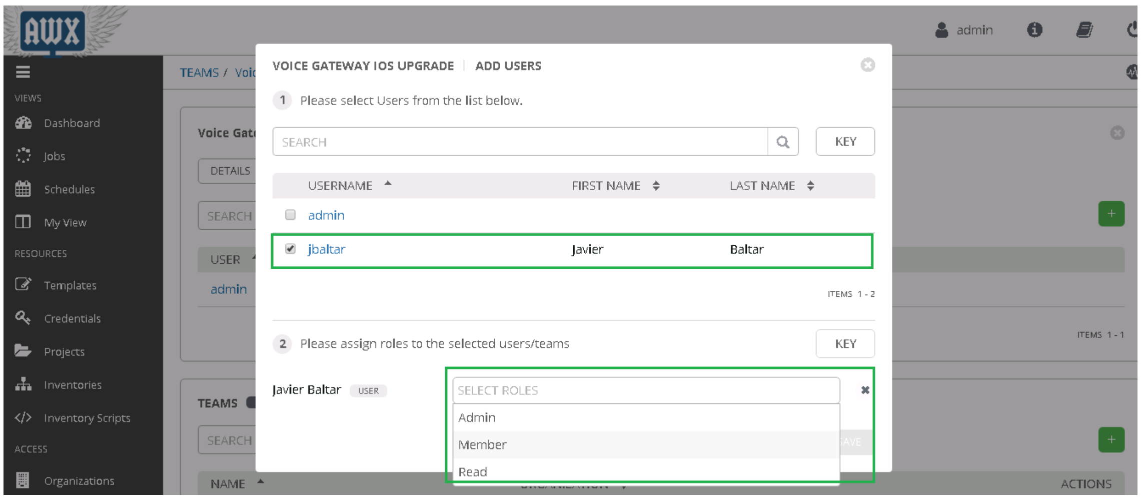
* USERNAME	PASSWORD	CONFIRM PASSWORD
jbaltar	SHOW	SHOW

USER TYPE: Normal User

There are three user types: Normal User, System Auditor and System Administrator as shown below:



From the Teams menu, you can add users to teams assigning the proper role (admin, member or read only).

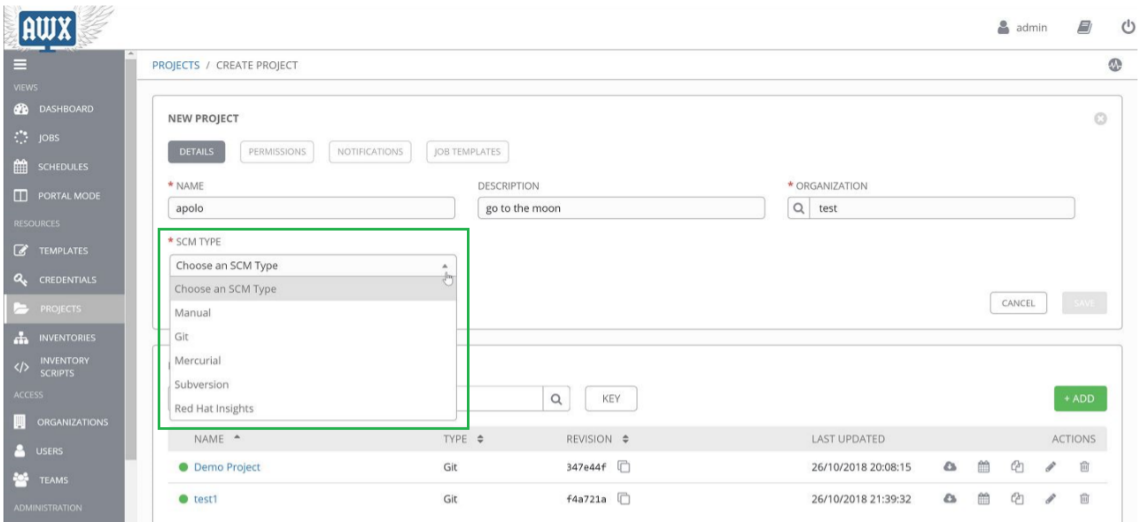

VOICE GATEWAY IOS UPGRADE
ADD USERS
1 Please select Users from the list below.
2 Please assign roles to the selected users/teams
Javier Baltar USER
SELECT ROLES
Admin
Member
Read

USERNAME	FIRST NAME	LAST NAME
admin		
✓ jbaltar	Javier	Baltar

Project

Next, you configure a project, the project contains the SCM type, and playbook directory which you must select. The SCM type can be manual but other options such as Git are available for central management of scripts.

The playbook directory drop down should contain the sub directories in the root projects folder. Select the relevant folder for this project that contains all the playbooks required for jobs connected to this project.



NEW PROJECT

* NAME	DESCRIPTION	* ORGANIZATION
apolo	go to the moon	test

* SCM TYPE

Choose an SCM Type
- Choose an SCM Type
- Manual
- Git
- Mercurial
- Subversion
- Red Hat Insights

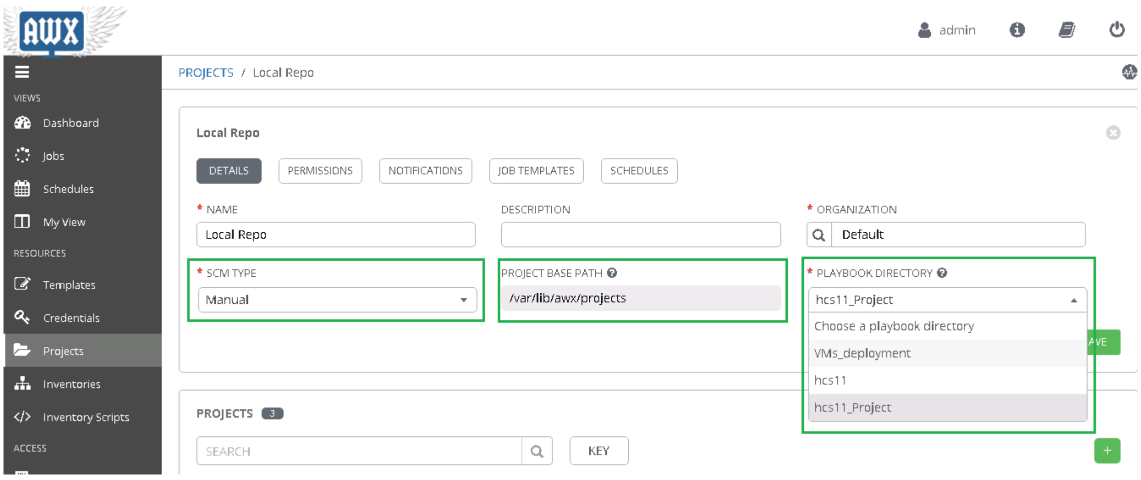
NAME	TYPE	REVISION	LAST UPDATED	ACTIONS
Demo Project	Git	347e44f	26/10/2018 20:08:15	
test1	Git	f4a721a	26/10/2018 21:39:32	

Manual Local Repo

You can create the project folder to store the playbooks in the following local path:

cd /var/lib/awx/projects/

mkdir my_project_name



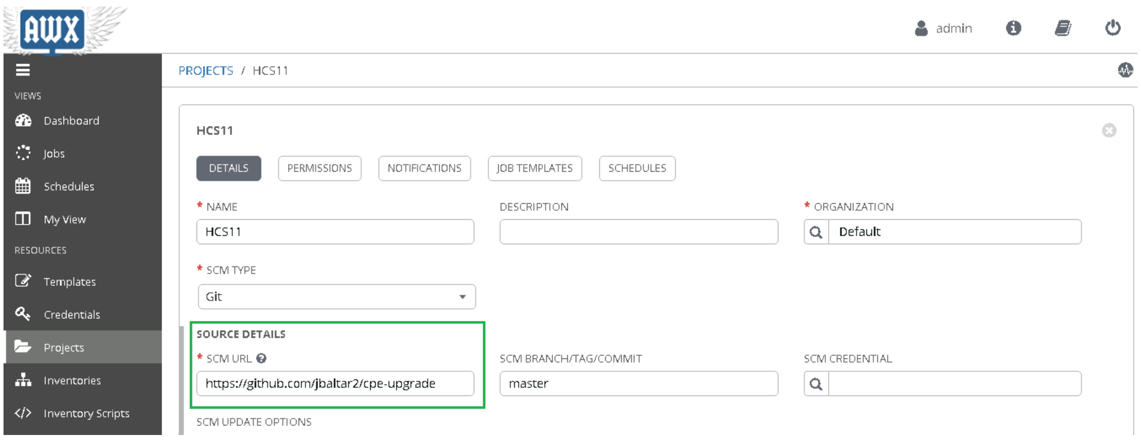
Local Repo

* NAME	DESCRIPTION	* ORGANIZATION
Local Repo		Default
* SCM TYPE	PROJECT BASE PATH	* PLAYBOOK DIRECTORY
Manual	/var/lib/awx/projects	hcs11_Project

PLAYBOOK DIRECTORY options:
- Choose a playbook directory
- VMs_deployment
- hcs11
- hcs11_Project

Git Repo

A externat Git repository can be configured at Project level as shown below:



HCS11

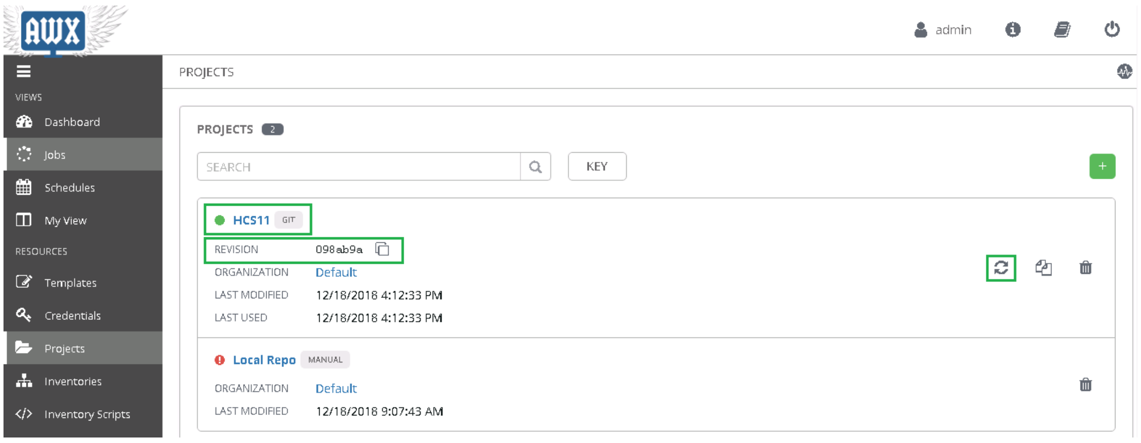
* NAME: HCS11	DESCRIPTION:	* ORGANIZATION: Default
* SCM TYPE: Git		

SOURCE DETAILS

* SCM URL: https://github.com/jbaltar2/cpe-upgrade	SCM BRANCH/TAG/COMMIT: master	SCM CREDENTIAL:

SCM UPDATE OPTIONS

The Projects tab displays the status of the repo, last revision and the sync button.



PROJECTS 2

HCS11 GIT

REVISION	098ab9a
ORGANIZATION	Default
LAST MODIFIED	12/18/2018 4:12:33 PM
LAST USED	12/18/2018 4:12:33 PM

Local Repo MANUAL

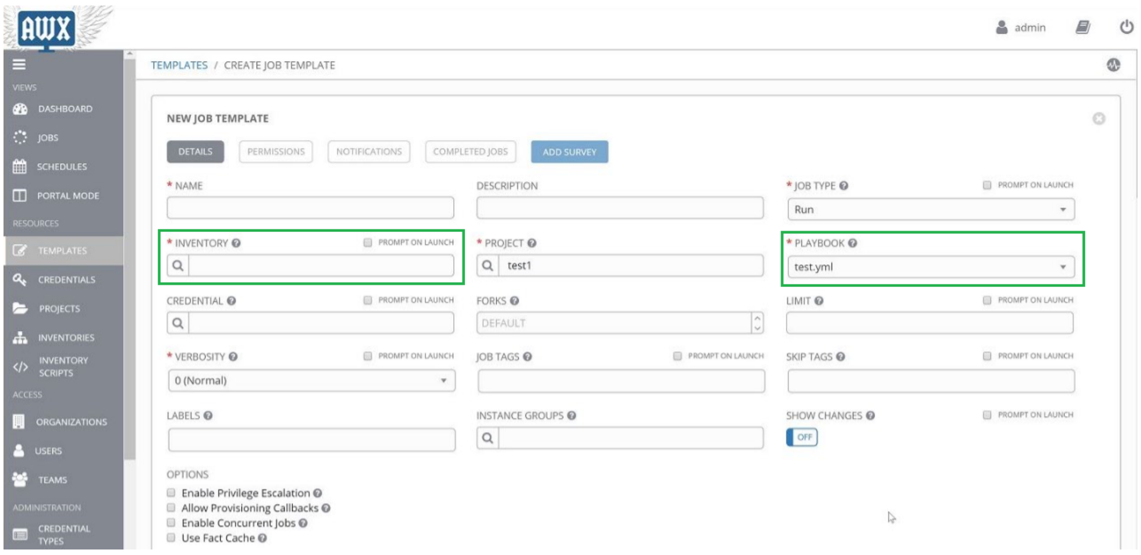
ORGANIZATION	Default
LAST MODIFIED	12/18/2018 9:07:43 AM

Templates

Job Template

A job template is a definition and set of parameters for running an Ansible job. Job templates are useful to execute the same job many times and reuse of Ansible playbook content between teams.

To create a new job template, click the add ⊕ button then select "Job Template" from the menu list.



NEW JOB TEMPLATE

* NAME:	DESCRIPTION:	* JOB TYPE: Run (PROMPT ON LAUNCH)
* INVENTORY: (PROMPT ON LAUNCH)	* PROJECT: test1	* PLAYBOOK: test.yml
CREDENTIAL: (PROMPT ON LAUNCH)	FORKS: DEFAULT	LIMIT: (PROMPT ON LAUNCH)
* VERBOSITY: 0 (Normal) (PROMPT ON LAUNCH)	JOB TAGS: (PROMPT ON LAUNCH)	SKIP TAGS: (PROMPT ON LAUNCH)
LABELS:	INSTANCE GROUPS:	SHOW CHANGES: OFF (PROMPT ON LAUNCH)

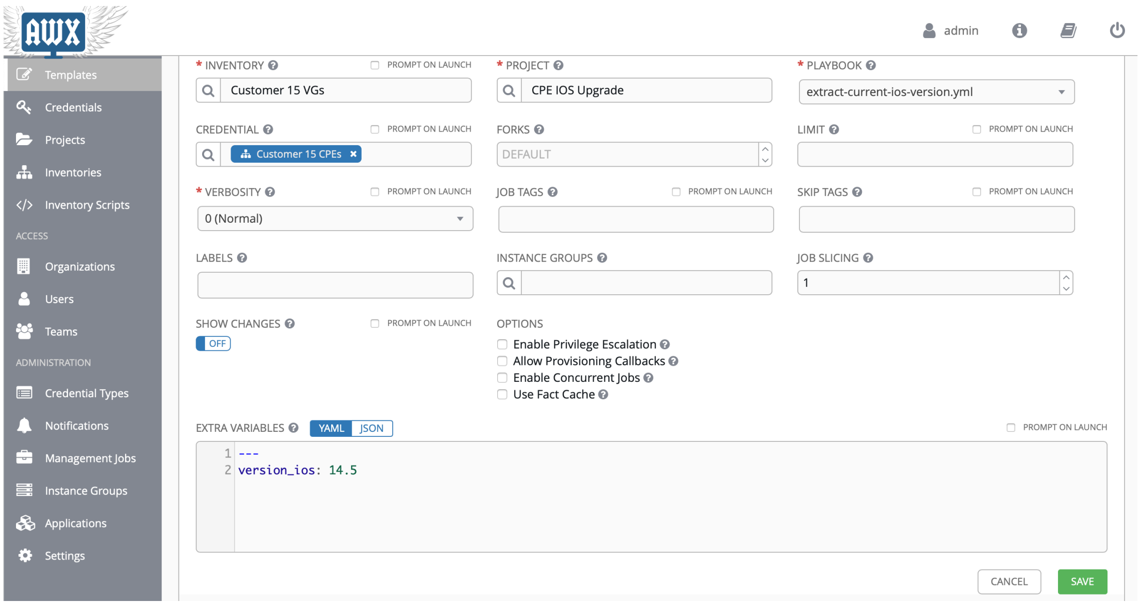
OPTIONS
- ☐ Enable Privilege Escalation
- ☐ Allow Provisioning Callbacks
- ☐ Enable Concurrent Jobs
- ☐ Use Fact Cache

Enter the appropriate details into the following fields:

- Name: Enter a name for the job.
- Description: Enter an arbitrary description as appropriate (optional).
- Job Type:
  - Run: Execute the playbook when launched, running Ansible tasks on the selected hosts.
  - Check: Perform a “dry run” of the playbook and report changes that would be made without actually making them. Tasks that do not support check mode will be skipped and will not report potential changes.
    - Prompt on Launch – If selected, even if a default value is supplied, you will be prompted upon launch to choose a job type of run or check.
- Inventory: Choose the inventory to be used with this job template from the inventories available to the currently logged in AWX user.
  - Prompt on Launch – If selected, even if a default value is supplied, you will be prompted upon launch to choose an inventory to run this job template against.
- Project: Choose the project to be used with this job template from the projects available to the currently logged in AWX user.
- Playbook: Choose the playbook to be launched with this job template from the available playbooks. This menu is automatically populated with the names of the playbooks found in the project base path for the selected project.
- Credential: Click the search button to open a separate window. Choose the credential from the available options to be used with this job template. Use the drop-down menu list to filter by credential type if the list is extensive.
  - Prompt on Launch: If selected, upon launching a job template that has a default machine credential, you will not be able to remove the default machine credential in the Prompt dialog without replacing it with another machine credential before it can launch.
- Forks: The number of parallel or simultaneous processes to use while executing the playbook. A value of zero uses the Ansible default setting, which is 5 parallel processes unless overridden in /etc/ansible/ansible.cfg.
- Limit: A host pattern to further constrain the list of hosts managed or affected by the playbook. Multiple patterns can be separated by colons (“:”). As with Ansible engine, “a:b” means “in group a or b”, “a:b:&c” means “in a or b but must be in c”, and “a:!b” means “in a, and definitely not in b”.

  - Prompt on Launch: If selected, even if a default value is supplied, you will be prompted upon launch to choose a limit.
- Verbosity: Control the level of output Ansible produces as the playbook executes. Set the verbosity to any of Default, Verbose, or Debug. This only appears in the “details” report view. Verbose logging includes the output of all commands. Debug logging is exceedingly verbose and includes information on SSH operations that can be useful in certain support instances.
  - Prompt on Launch: If selected, even if a default value is supplied, you will be prompted upon launch to choose a verbosity.
- Job Tags: Provide a comma-separated list of playbook tags to specify what parts of the playbooks should be executed.
  - Prompt on Launch – If selected, even if a default value is supplied, you will be prompted upon launch to choose a job tag.
- Skip Tags: Provide a comma-separated list of playbook tags to skip certain tasks or parts of the playbooks to be executed.
  - Prompt on Launch – If selected, even if a default value is supplied, you will be prompted upon launch to choose tag(s) to skip.
- Labels: Supply optional labels that describe this job template, such as “dev” or “test”. Labels can be used to group and filter job templates and completed jobs in the AWX display.
  - Labels are created when they are added to the Job Template. Labels are associated to a single Organization using the Project that is provided in the Job Template. Members of the Organization can create labels on a Job Template if they have edit permissions (such as admin role).
  - Once the Job Template is saved, the labels appear in the Job Templates overview.
  - Click on the “x” beside a label to remove it. When a label is removed, and is no longer associated with a Job or Job Template, the label is permanently deleted from the list of Organization labels.
  - Jobs inherit labels from the Job Template at the time of launch. If a label is deleted from a Job Template, it is also deleted from the Job.
- Instance Groups: Click the search button to open a separate window. Choose the instance groups on which you want to run this job template. If the list is extensive, use the search to narrow the options.
- Show Changes: Allows you to see the changes made by Ansible tasks.
  - Prompt on Launch – If selected, even if a default value is supplied, you will be prompted upon launch to choose whether or not to show changes.
- Options: Supply optional labels that describe this job template, such as “dev” or “test”. Labels can be used to group and filter job templates and completed jobs in the AWX display.
  - Enable Privilege Escalation: If enabled, run this playbook as an administrator. This is the equivalent of passing the --become option to the ansible-playbook command.
  - Allow Provisioning Callbacks: Enable a host to call back to AWX via the AWX API and invoke the launch of a job from this job template.
  - Enable Concurrent Jobs: Allow jobs in the queue to run simultaneously if not dependent on one another.
  - Use Fact Cache: When enabled, AWX will activate an Ansible fact cache plugin for all hosts in an inventory related to the job running.
- Extra Variables:
  - Pass extra command line variables to the playbook. This is the “-e” or “–extra-vars” command line parameter for ansible-playbook that is documented in the Ansible documentation at Passing Variables on the Command Line.

 ○ Provide key/value pairs using either YAML or JSON. These variables have a maximum value of precedence and overrides other variables specified elsewhere.



* INVENTORY: Customer 15 VGs	* PROJECT: CPE IOS Upgrade	* PLAYBOOK: extract-current-ios-version.yml
CREDENTIAL: Customer 15 CPEs	FORKS: DEFAULT	LIMIT:
* VERBOSITY: 0 (Normal)	JOB TAGS:	SKIP TAGS:
LABELS:	INSTANCE GROUPS:	JOB SLICING: 1
SHOW CHANGES: OFF	OPTIONS: Enable Privilege Escalation, Allow Provisioning Callbacks, Enable Concurrent Jobs, Use Fact Cache	

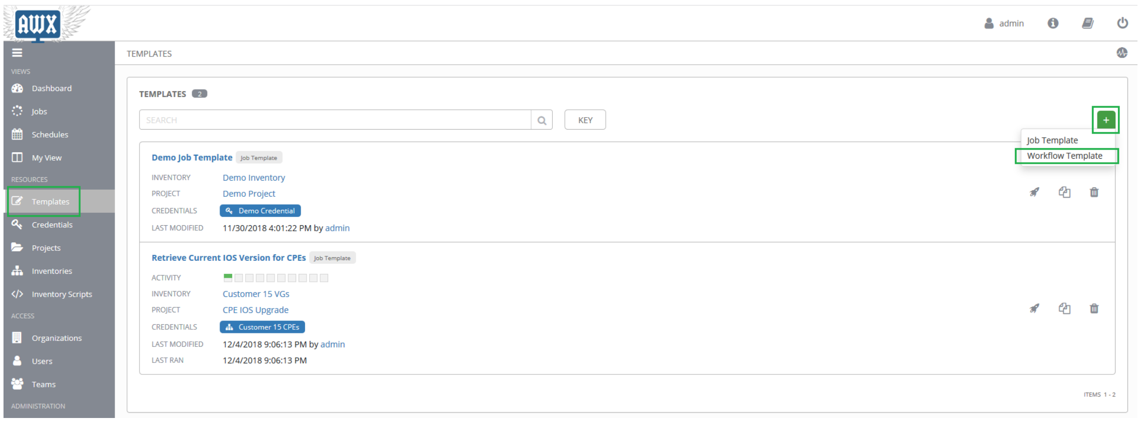
EXTRA VARIABLES (YAML / JSON)

```
---
version_ios: 14.5
```

Workflow Template

This feature in AWX enables users to create sequences consisting of any combination of job templates, project syncs, and inventory syncs that are linked together in order to execute them as a single unit. Another reason workflows are useful is because they allow the user to take any number of playbooks with the ability to make a decision tree depending on a job's success or failure and sending notifications also.

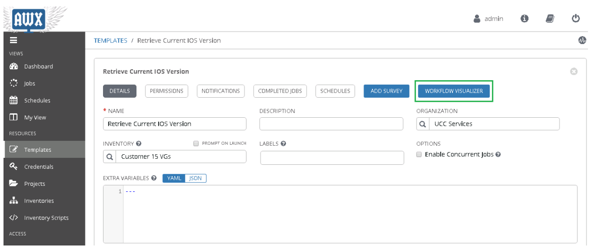
From the Templates meny, click on ➕ and make sure to select "Workflow Template".



Demo Job Template (Job Template)
INVENTORY: Demo Inventory
PROJECT: Demo Project
CREDENTIALS: Demo Credential
LAST MODIFIED: 11/30/2018 4:01:22 PM by admin

Retrieve Current IOS Version for CPEs (Job Template)
ACTIVITY
INVENTORY: Customer 15 VGs
PROJECT: CPE IOS Upgrade
CREDENTIALS: Customer 15 CPEs
LAST MODIFIED: 12/4/2018 9:06:13 PM by admin
LAST RAN: 12/4/2018 9:06:13 PM

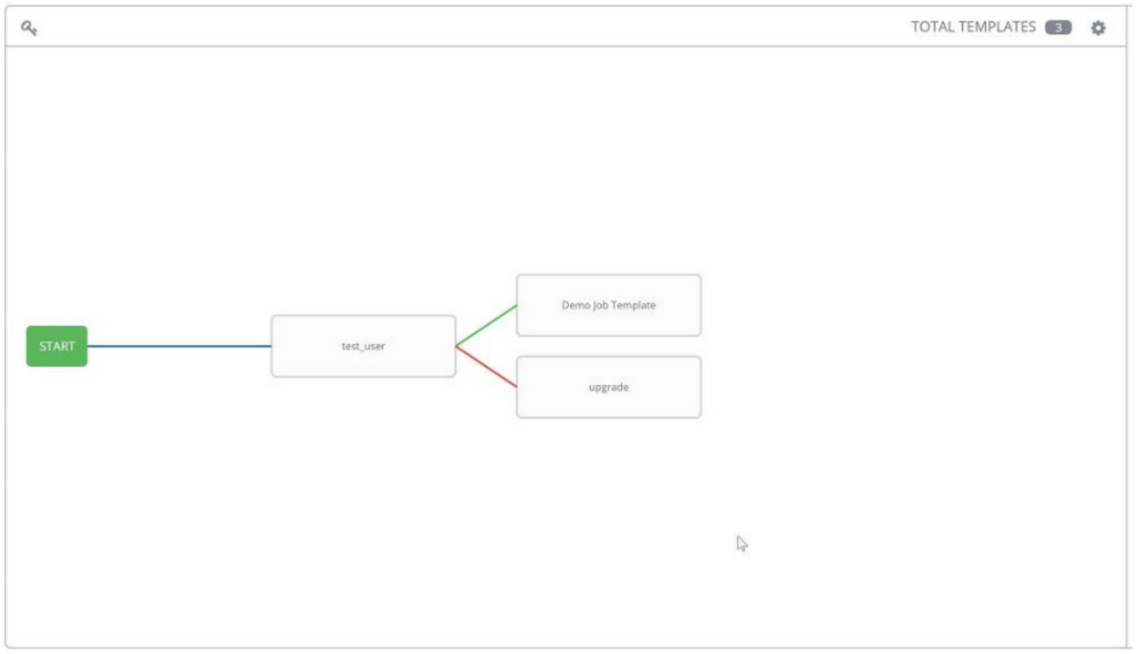
Once the mandatory fields are filled, click on Save and on Workflow Visualizer:



Retrieve Current IOS Version
* NAME: Retrieve Current IOS Version
DESCRIPTION:
ORGANIZATION: UCC Services
INVENTORY: Customer 15 VGs
LABELS:
OPTIONS: Enable Concurrent Jobs
EXTRA VARIABLES (YAML / JSON)

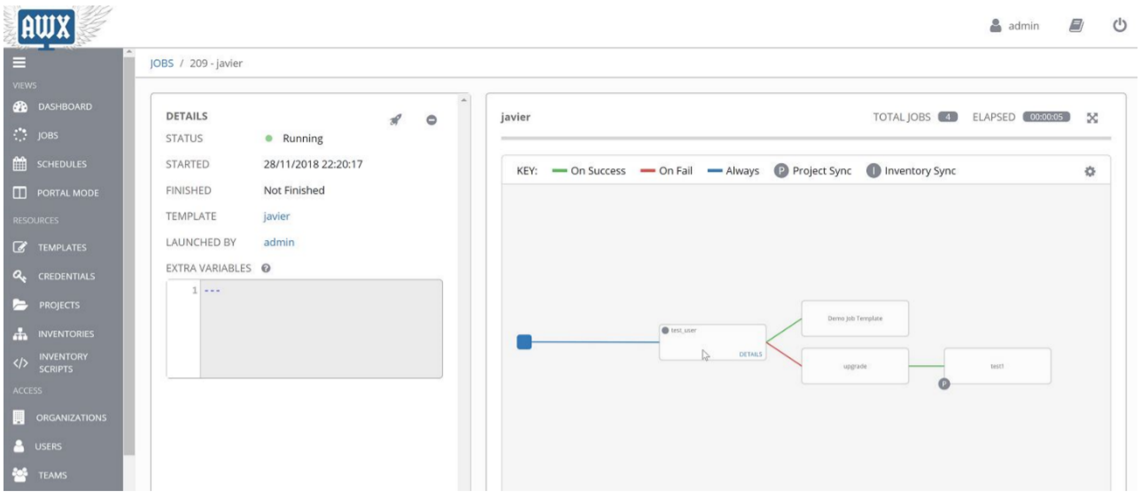
```
---
```

This screen will come up, where you can add different job templates and make sure they run on failure, success, or with either outcome.



Note that you can decide if things run on success, on failure, or always.

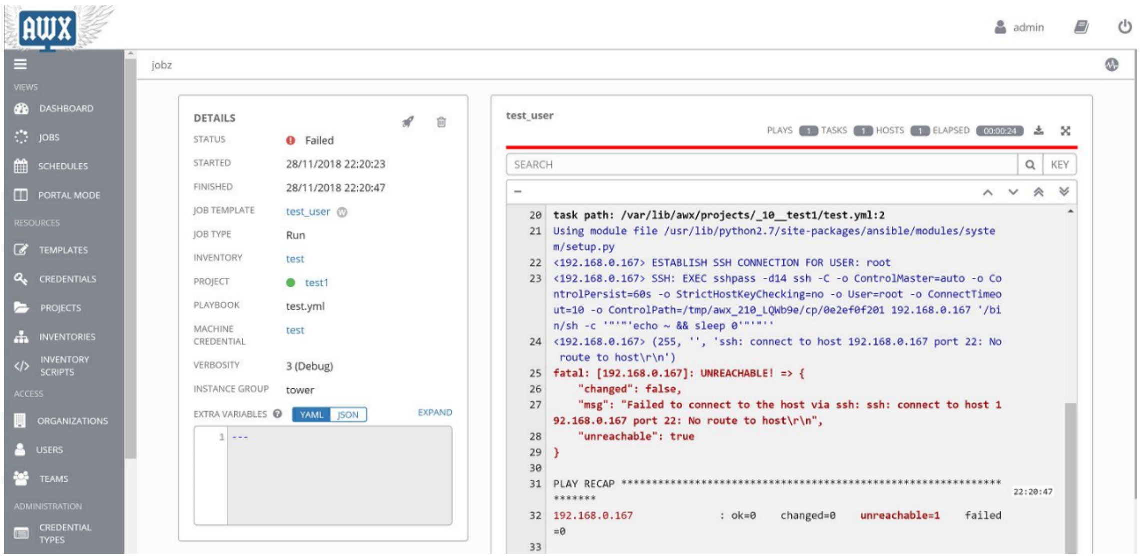
After everything is set and saved, you are ready to launch your template, which you can do by clicking on the rocket icon next to the workflow you would like to run.



DETAILS

STATUS	● Running
STARTED	28/11/2018 22:20:17
FINISHED	Not Finished
TEMPLATE	javier
LAUNCHED BY	admin
EXTRA VARIABLES	

```
---
```



jobz

DETAILS

STATUS	Failed
STARTED	28/11/2018 22:20:23
FINISHED	28/11/2018 22:20:47
JOB TEMPLATE	test_user
JOB TYPE	Run
INVENTORY	test
PROJECT	test1
PLAYBOOK	test.yml
MACHINE CREDENTIAL	test
VERBOSITY	3 (Debug)
INSTANCE GROUP	tower

EXTRA VARIABLES YAML JSON EXPAND

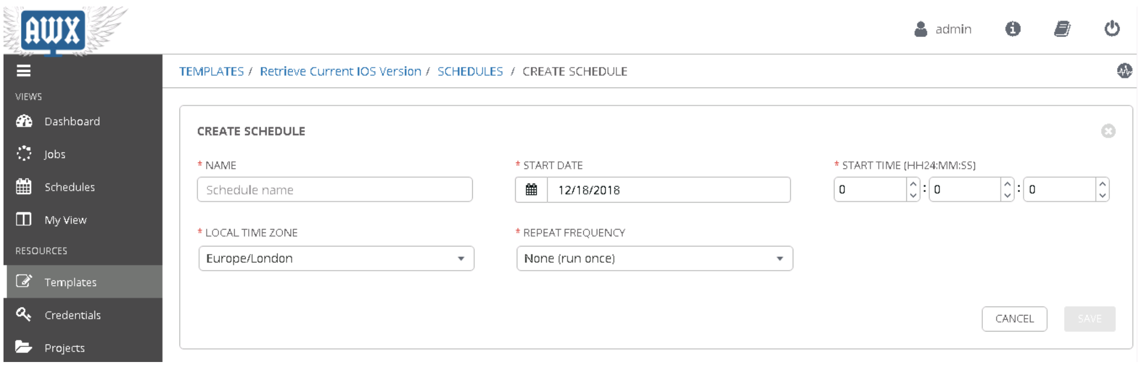
```
---
```

test_user

PLAYS 1 TASKS 1 HOSTS 1 ELAPSED 00:00:24

```
task path: /var/lib/awx/projects/_10__test1/test.yml:2
Using module file /usr/lib/python2.7/site-packages/ansible/modules/syste
m/setup.py
<192.168.0.167> ESTABLISH SSH CONNECTION FOR USER: root
<192.168.0.167> SSH: EXEC sshpass -d14 ssh -C -o ControlMaster=auto -o Co
ntrolPersist=60s -o StrictHostKeyChecking=no -o User=root -o ConnectTimeo
ut=10 -o ControlPath=/tmp/awx_210_LQWb9e/cp/0e2ef0f201 192.168.0.167 '/bi
n/sh -c '"'"'echo ~ && sleep 0'"'"''
<192.168.0.167> (255, '', 'ssh: connect to host 192.168.0.167 port 22: No
 route to host\r\n')
fatal: [192.168.0.167]: UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 1
92.168.0.167 port 22: No route to host\r\n",
    "unreachable": true
}

PLAY RECAP *************************************************************    22:20:47
*******
192.168.0.167              : ok=0    changed=0    unreachable=1    failed
=0

```

You can schedule your workflows to run when you need them to. Click on the calendar icon next to any workflow job template:



TEMPLATES / Retrieve Current IOS Version / SCHEDULES / CREATE SCHEDULE

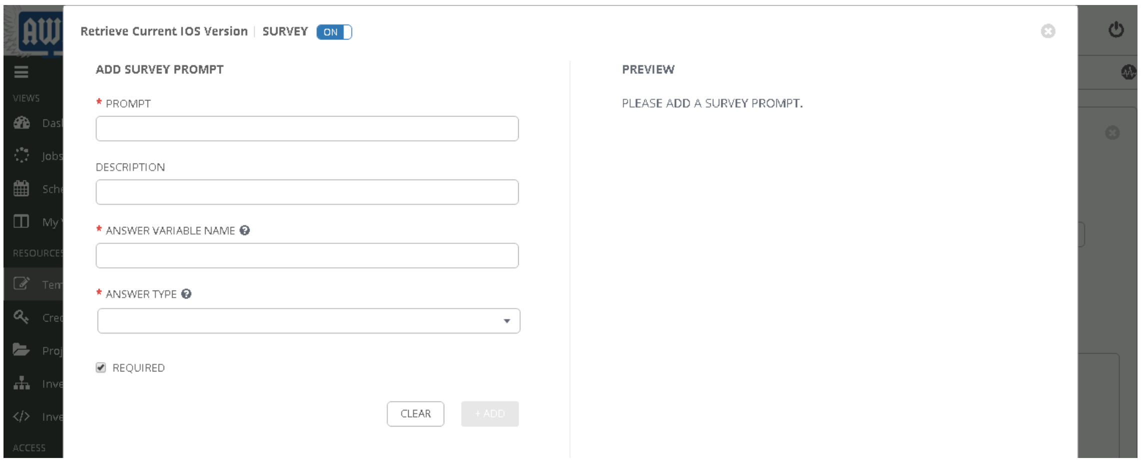
CREATE SCHEDULE

* NAME	* START DATE	* START TIME (HH24:MM:SS)
Schedule name	12/18/2018	0 : 0 : 0
* LOCAL TIME ZONE	* REPEAT FREQUENCY	
Europe/London	None (run once)	

CANCEL SAVE

If you need to set extra variables for the playbooks involved in a workflow template and/or allow for authorization of user input, then setting up surveys is the way to go.

In order to set one up, select a workflow template and click on the “Add Survey” button:



Retrieve Current IOS Version | SURVEY ON

ADD SURVEY PROMPT

* PROMPT

DESCRIPTION

* ANSWER VARIABLE NAME

* ANSWER TYPE

☑ REQUIRED

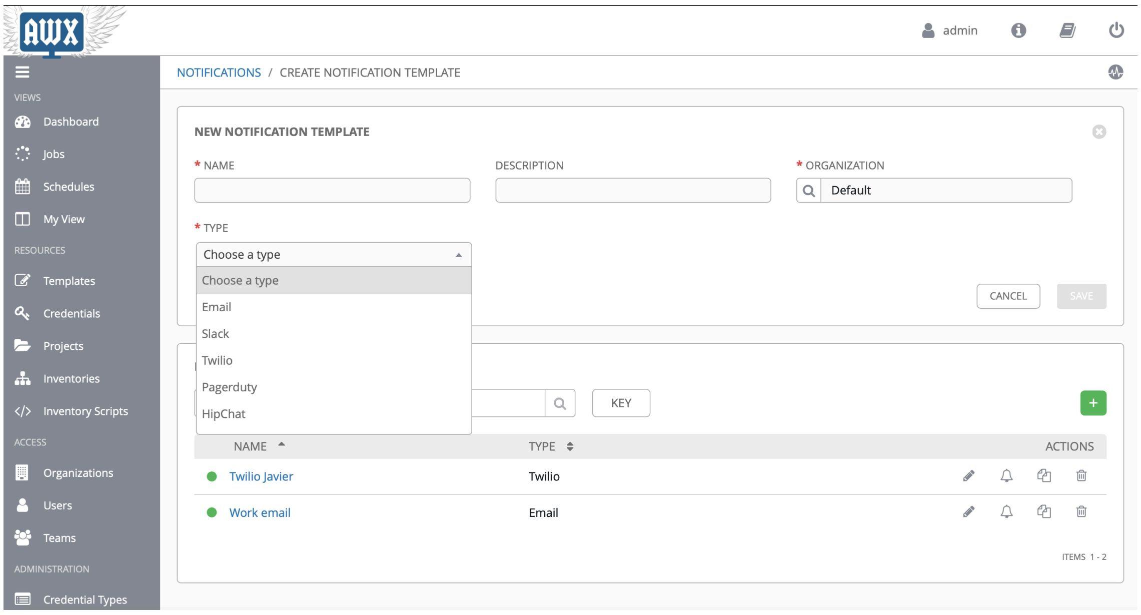
CLEAR + ADD

PREVIEW

PLEASE ADD A SURVEY PROMPT.

Notifications

AWX provides a bunch of notification channels: email, Slack, Twilio, etc.



NEW NOTIFICATION TEMPLATE

* NAME	DESCRIPTION	* ORGANIZATION
		Default

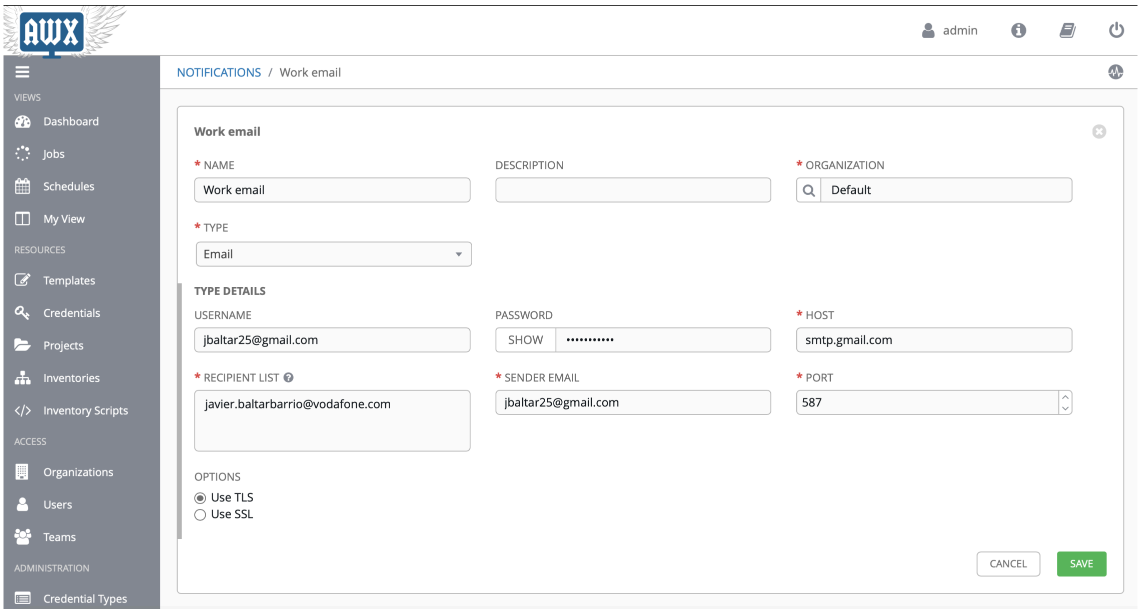
* TYPE: Choose a type
  - Choose a type
  - Email
  - Slack
  - Twilio
  - Pagerduty
  - HipChat

NAME	TYPE	ACTIONS
● Twilio Javier	Twilio	
● Work email	Email	

ITEMS 1 - 2

Email

Click on Notification and add a new one:



Work email

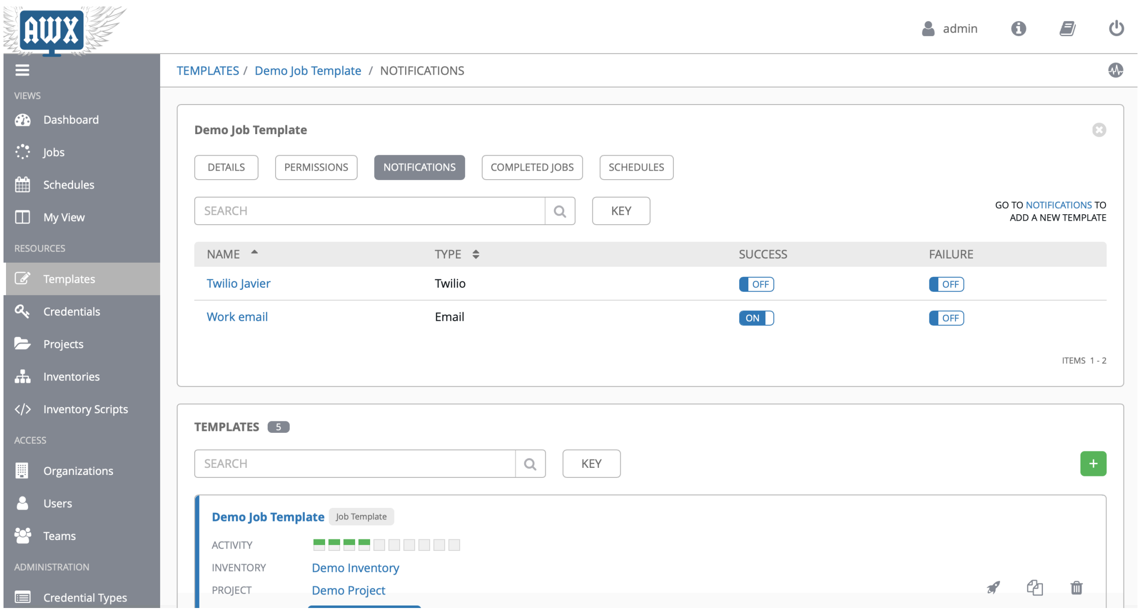
* NAME: Work email	DESCRIPTION:	* ORGANIZATION: Default
* TYPE: Email		

TYPE DETAILS

USERNAME: jbaltar25@gmail.com	PASSWORD: SHOW •••••••••••	* HOST: smtp.gmail.com
* RECIPIENT LIST: javier.baltarbarrio@vodafone.com	* SENDER EMAIL: jbaltar25@gmail.com	* PORT: 587

OPTIONS
- [✓ Use TLS]
- Use SSL

From the Templates tab, you can control the notifications that are triggered by an specific template when it is successfully completed or fails.



Demo Job Template

NAME	TYPE	SUCCESS	FAILURE
Twilio Javier	Twilio	OFF	OFF
Work email	Email	ON	OFF

ITEMS 1 - 2

TEMPLATES 5

Demo Job Template Job Template
ACTIVITY
INVENTORY Demo Inventory
PROJECT Demo Project

This is an example of email notification received to your mailbox.

Twilio

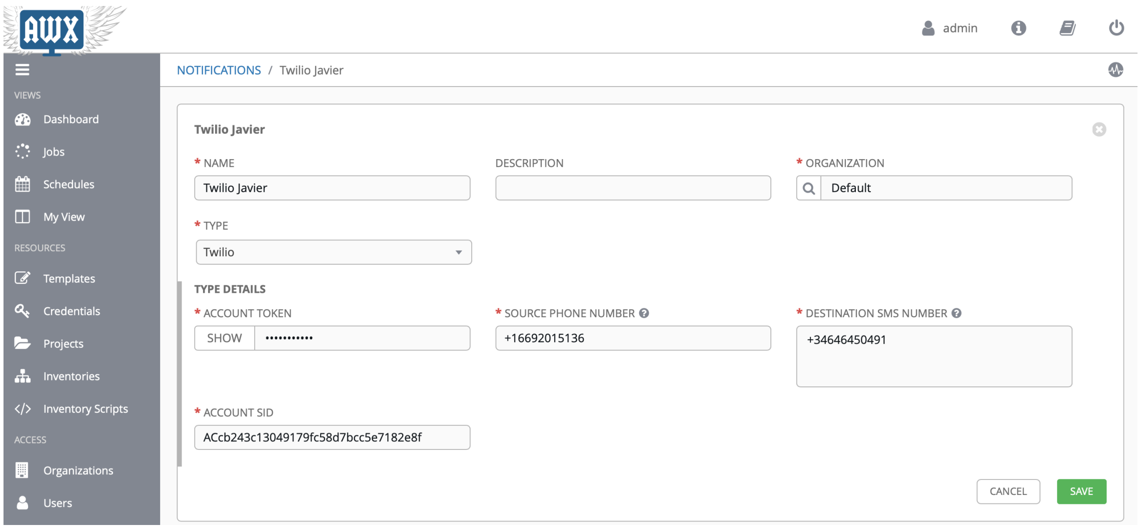
To trigger a Twilio API call, you need to configure the following items in your account:

Click on dashboard > Twilio account details, where Account SID and AUTH TOKEN are listed.

Click on Twilio verified called IDs, which contains the allowed destination numbers.

Click on Geo Restrictions in order to ensure your destination number is not restricted:

You have to create your Twilio notification in AWX:

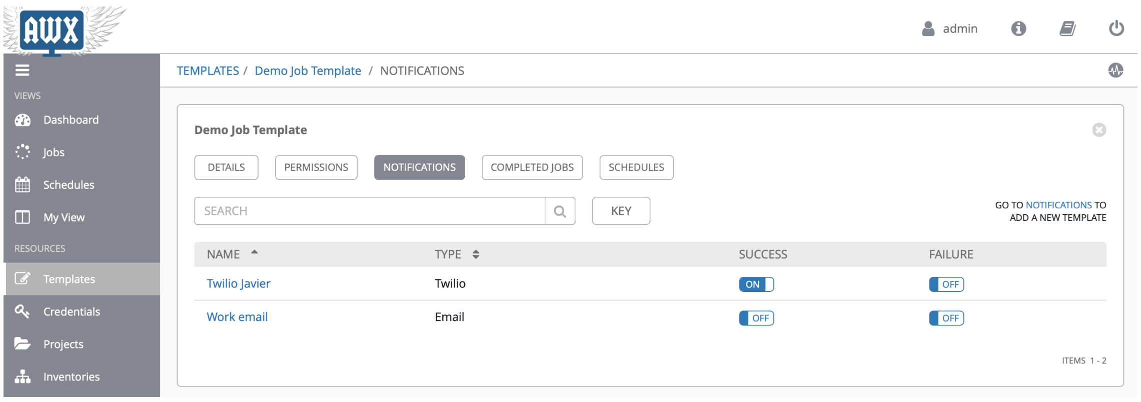


Twilio Javier

* NAME: Twilio Javier
DESCRIPTION:
* ORGANIZATION: Default
* TYPE: Twilio

TYPE DETAILS
* ACCOUNT TOKEN: SHOW ••••••••••
* SOURCE PHONE NUMBER: +16692015136
* DESTINATION SMS NUMBER: +34646450491
* ACCOUNT SID: ACcb243c13049179fc58d7bcc5e7182e8f

Associate your template with the notification:



TEMPLATES / Demo Job Template / NOTIFICATIONS

Demo Job Template

DETAILS | PERMISSIONS | NOTIFICATIONS | COMPLETED JOBS | SCHEDULES

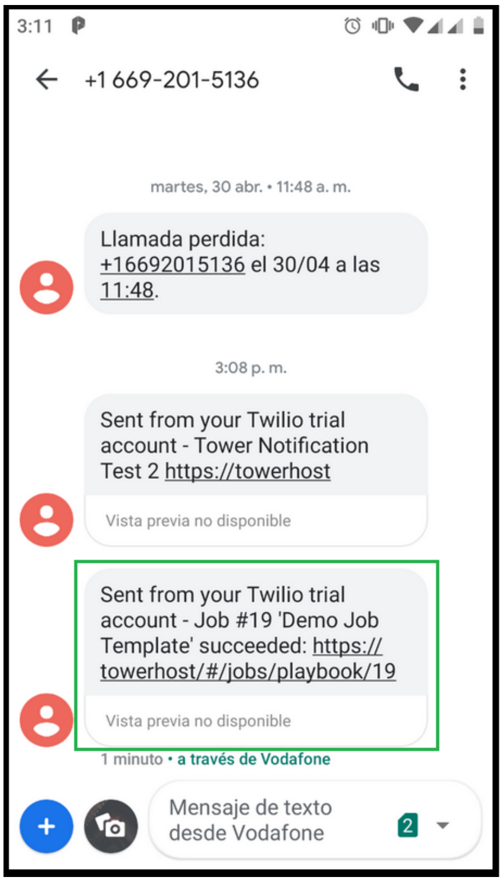
SEARCH | KEY

GO TO NOTIFICATIONS TO ADD A NEW TEMPLATE

NAME	TYPE	SUCCESS	FAILURE
Twilio Javier	Twilio	ON	OFF
Work email	Email	OFF	OFF

ITEMS 1 - 2

After running your job, the SMS notification below is dropped to my mobile:



martes, 30 abr. • 11:48 a. m.

Llamada perdida: +16692015136 el 30/04 a las 11:48.

3:08 p. m.

Sent from your Twilio trial account - Tower Notification Test 2 https://towerhost

Vista previa no disponible

Sent from your Twilio trial account - Job #19 'Demo Job Template' succeeded: https://towerhost/#/jobs/playbook/19

Vista previa no disponible

1 minuto • a través de Vodafone

Mensaje de texto desde Vodafone

Advanced Playbooks Features

Error Handling

Blocks allow you to group related tasks together and apply particular task parameters on the block level. They also allow you to handle errors inside the blocks in a way similar to most programming languages exception handling.

This example playbook that uses blocks to run group of tasks specific to one platform. If you want to perform a series of tasks with one set of task parameters (e.g. with_items, when, or sudo) applied, blocks are very handy.

Blocks are also useful if you want to be able to gracefully handle failures. There might be a non-critical task that is not essential for a deployment to succeed, so it would be better to gracefully handle a failure than to stop the entire deployment rollout.

This is an example of how to use a block to gracefully handle task failures:

```
tasks:
 - block:
     - name: Shell script to deploy a critical service.
       script: deploy_application.sh
   rescue:
     - name: This will only run in case of an error in the block.
       debug: msg="There was an error in the block."
   always:
     - name: This will always run
       debug: msg="This always executes."
```

Tasks inside the block will be run first. If there is a failure in any task in block, tasks inside rescue will be run. The tasks inside always will always be run, whether or not there were failures in either block or rescue.

Please refer to Ansible documentation: https://docs.ansible.com/ansible/latest/user_guide/playbooks_blocks.html#error-handling

Handlers

A Handler is exactly the same as a task, but it will run when called by another task. A Handler will take an action when called by an event it listens for.

This is useful for secondary actions that might be required after running a task, such as starting a new service after installation or reloading a service after a configuration change.

```
---
- hosts: webservers

  tasks:
   - name: Install Nginx
     apt: pkg=nginx state=installed update_cache=true
     notify:
       - start Nginx

  handlers:
   - name: start Nginx
     service: name=nginx state=started
```

Templates and Variables

A template in Ansible is a file which contains all your configuration parameters, but the dynamic values are given as variables. During the playbook execution, depending on the conditions like which cluster you are using, the variables will be replaced with the relevant values.

```
- hosts: webservers
   vars:
     variable_to_be_replaced: 'value1'
     inline_variable: 'value2'
  tasks:
   - name: Ansible using templates
      template:
        src: template_example.j2
        dest: /userA/docs/personal_data.txt
```

Template file

```
template_example.j2

{{ variable_to_be_replaced }}
Variable given as inline - {{ inline_variable }} -
```

This is the output of the playbook above:

```
value1
Variable given as inline - value2 -
```

Loops

With_items

```
- name: Remove users from the system.
   user:
     name: "{{ item }}"
     state: absent
     remove: yes
  with_items:
    - userA
    - userB
```

With_nested
Define variables

```
users_with_items:
  - name: "userA"
     personal_directories:
       - "old_files"
```

Playbook

```
- name: Create common users directories using
  file:
   dest: "/home/{{ item.0.name }}/{{ item.1 }}"
   owner: "{{ item.0.name }}"
   group: "{{ item.0.name }}"
   state: directory
 with_nested:
   - "{{ users_with_items }}"
   - "{{ common_directories }}"
```

File Manipulation

Useful ad-hoc commands for file manipulation:

Delete a File

```
#Delete the file /backups/tmp/nodelist.txt on all servers
ansible all -b -m file -a "state=absent path=/backups/tmp/nodelist.txt"
```

Update a Line in File

```
#Update the line of text "MY_SETTING" to "BLUE" in /opt/configuration.txt
on all servers
ansible all -b -m lineinfile -a "regexp=MY_SETTING line=BLUE path=/opt/
configuration.txt"
```

Extract /tmp/package.tgz to /opt/ on all hosts in webservers

```
ansible webservers -b -m unarchive -a "src=/tmp/package.tgz dest=/opt/
remote_src=yes"
```

Set the group ownership of a directory on each host in webservers

```
ansible webservers -b -m file -a "recurse=yes sate=directory path=/opt/
automators/secrets group=protected"
```

Modules

Please refer to https://docs.ansible.com/ansible/latest/modules/modules_by_category.html for Ansible module index.

Files

Archive

Some examples using the archive modules:

```
tasks:

- name: Backup Directory /var/log/application01/
- archive:
   path: /var/log/application01/
   dest: "/var/backups/application01-{{ ansible_date_time.date }}.tgz"
```

Copy

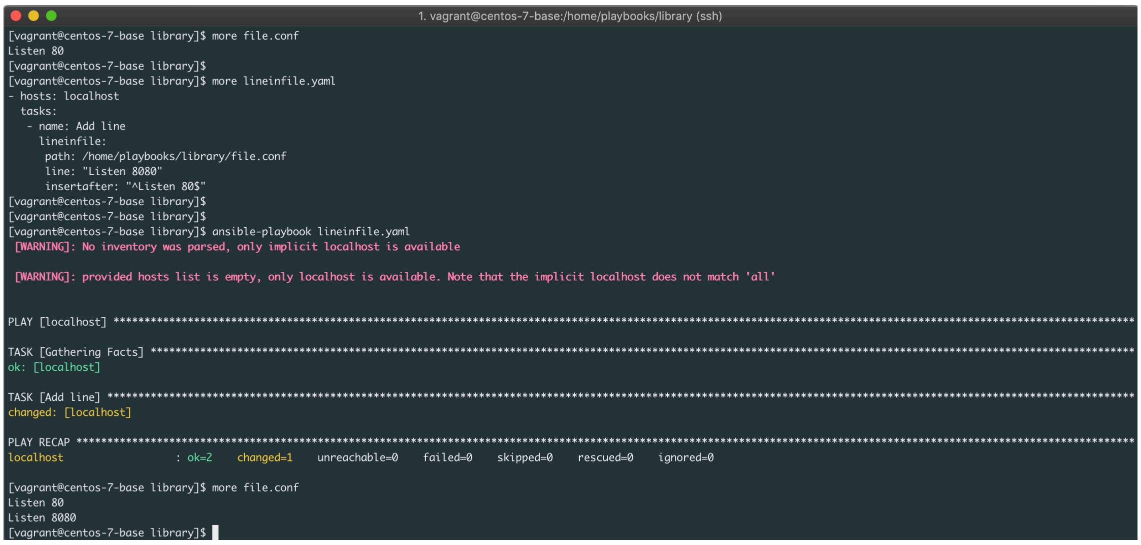
```
tasks:

- name: Copy File
- copy:
    src: /var/log/application01/filename
    dest: "/var/backups/filename
    owner: root
    group: root
    mode: u=r,g=r,o=
```

Fetch

```
tasks:

- name: Copy File from Remote Node

- fetch:
    src: /var/log/application01/filename
    dest: "/var/backups/
```

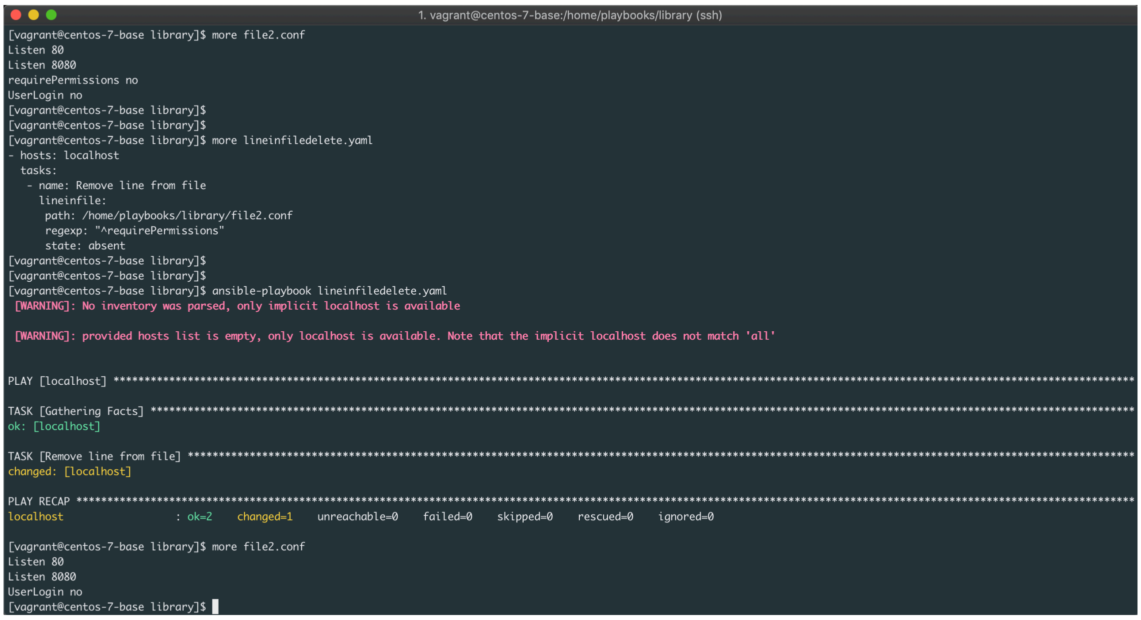
Lineinfile

The following playbook add a line after a string:



```
[vagrant@centos-7-base library]$ more file.conf
Listen 80
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ more lineinfile.yaml
- hosts: localhost
  tasks:
   - name: Add line
     lineinfile:
      path: /home/playbooks/library/file.conf
      line: "Listen 8080"
      insertafter: "^Listen 80$"
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ ansible-playbook lineinfile.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ************************************************************************************************************************

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

TASK [Add line] *************************************************************************************************************************
changed: [localhost]

PLAY RECAP ******************************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[vagrant@centos-7-base library]$ more file.conf
Listen 80
Listen 8080
[vagrant@centos-7-base library]$
```

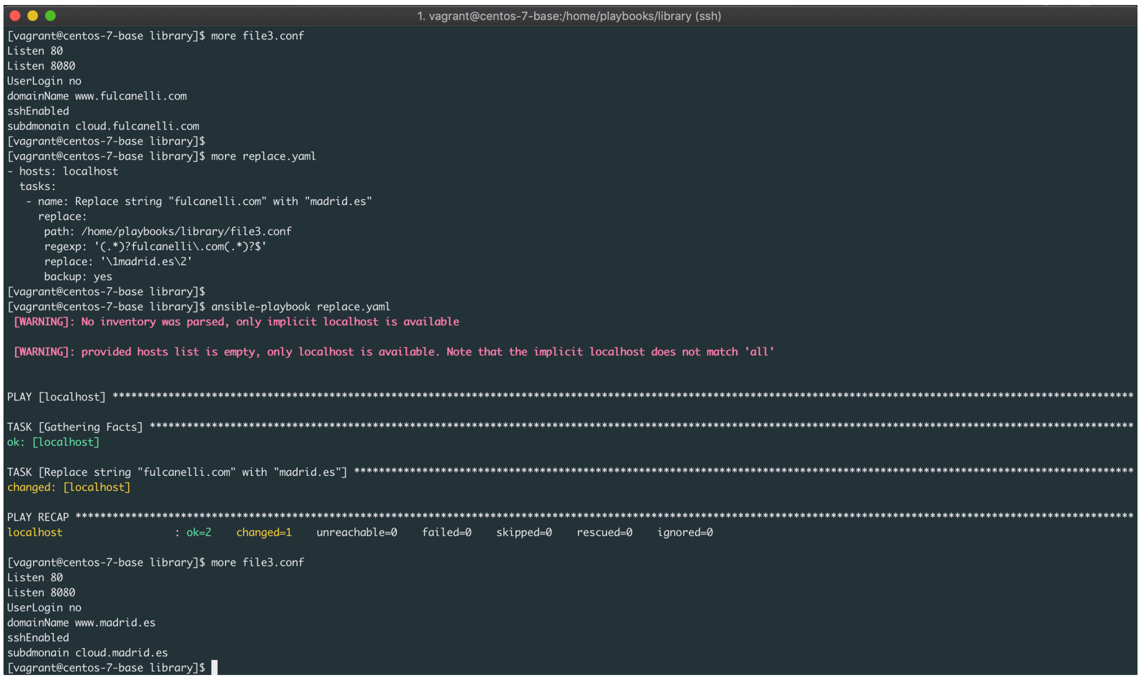
The following playbook deletes a line from a file:



```
[vagrant@centos-7-base library]$ more file2.conf
Listen 80
Listen 8080
requirePermissions no
UserLogin no
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ more lineinfiledelete.yaml
- hosts: localhost
  tasks:
   - name: Remove line from file
     lineinfile:
      path: /home/playbooks/library/file2.conf
      regexp: "^requirePermissions"
      state: absent
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ ansible-playbook lineinfiledelete.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ************************************************************************************************************************

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

TASK [Remove line from file] ************************************************************************************************************
changed: [localhost]

PLAY RECAP ******************************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[vagrant@centos-7-base library]$ more file2.conf
Listen 80
Listen 8080
UserLogin no
[vagrant@centos-7-base library]$
```

Replace

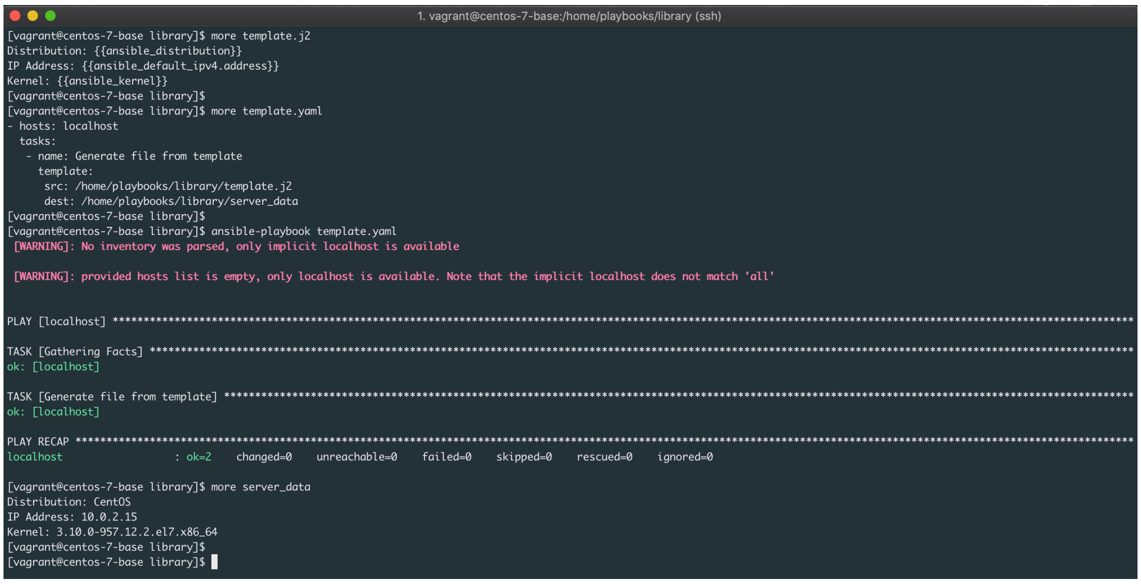
This is an example of playbook using the replace module.



```
1. vagrant@centos-7-base:/home/playbooks/library (ssh)
[vagrant@centos-7-base library]$ more file3.conf
Listen 80
Listen 8080
UserLogin no
domainName www.fulcanelli.com
sshEnabled
subdmonain cloud.fulcanelli.com
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ more replace.yaml
- hosts: localhost
  tasks:
   - name: Replace string "fulcanelli.com" with "madrid.es"
     replace:
      path: /home/playbooks/library/file3.conf
      regexp: '(.*)?fulcanelli\.com(.*)?$'
      replace: '\1madrid.es\2'
      backup: yes
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ ansible-playbook replace.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ********************************************************************************************************************************************************

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

TASK [Replace string "fulcanelli.com" with "madrid.es"] *****************************************************************************************************************
changed: [localhost]

PLAY RECAP **************************************************************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[vagrant@centos-7-base library]$ more file3.conf
Listen 80
Listen 8080
UserLogin no
domainName www.madrid.es
sshEnabled
subdmonain cloud.madrid.es
[vagrant@centos-7-base library]$
```

Template

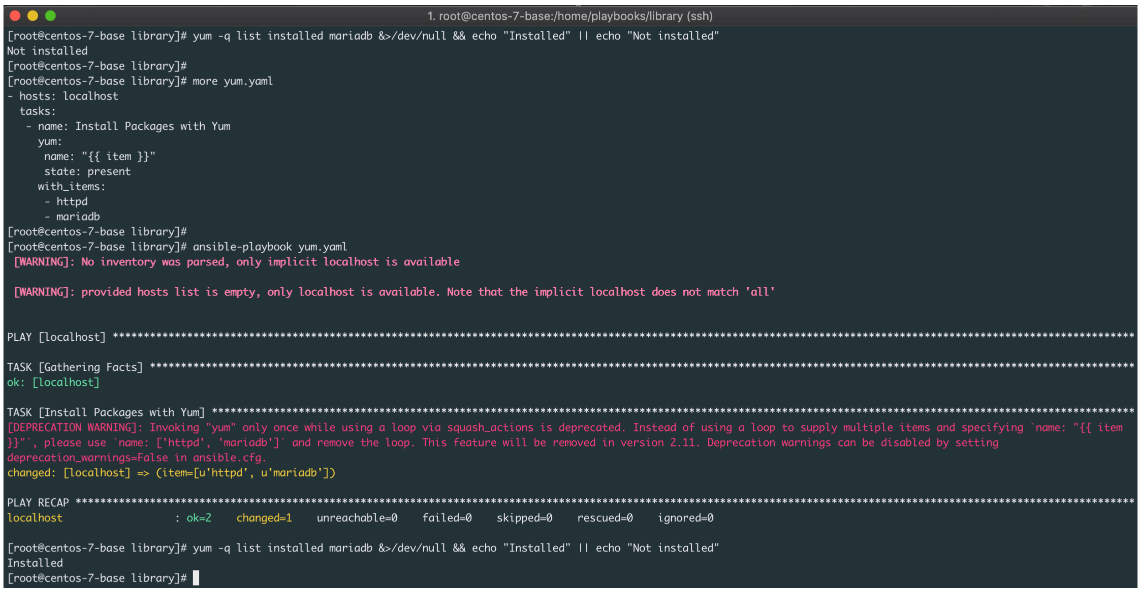
Example playbook using the template module.



```
1. vagrant@centos-7-base:/home/playbooks/library (ssh)
[vagrant@centos-7-base library]$ more template.j2
Distribution: {{ansible_distribution}}
IP Address: {{ansible_default_ipv4.address}}
Kernel: {{ansible_kernel}}
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ more template.yaml
- hosts: localhost
  tasks:
   - name: Generate file from template
     template:
      src: /home/playbooks/library/template.j2
      dest: /home/playbooks/library/server_data
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$ ansible-playbook template.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ********************************************************************************************************************************************************

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

TASK [Generate file from template] **************************************************************************************************************************************
ok: [localhost]

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

[vagrant@centos-7-base library]$ more server_data
Distribution: CentOS
IP Address: 10.0.2.15
Kernel: 3.10.0-957.12.2.el7.x86_64
[vagrant@centos-7-base library]$
[vagrant@centos-7-base library]$
```

Yum

Example playbook using the yum module for installing https and mariadb packages.

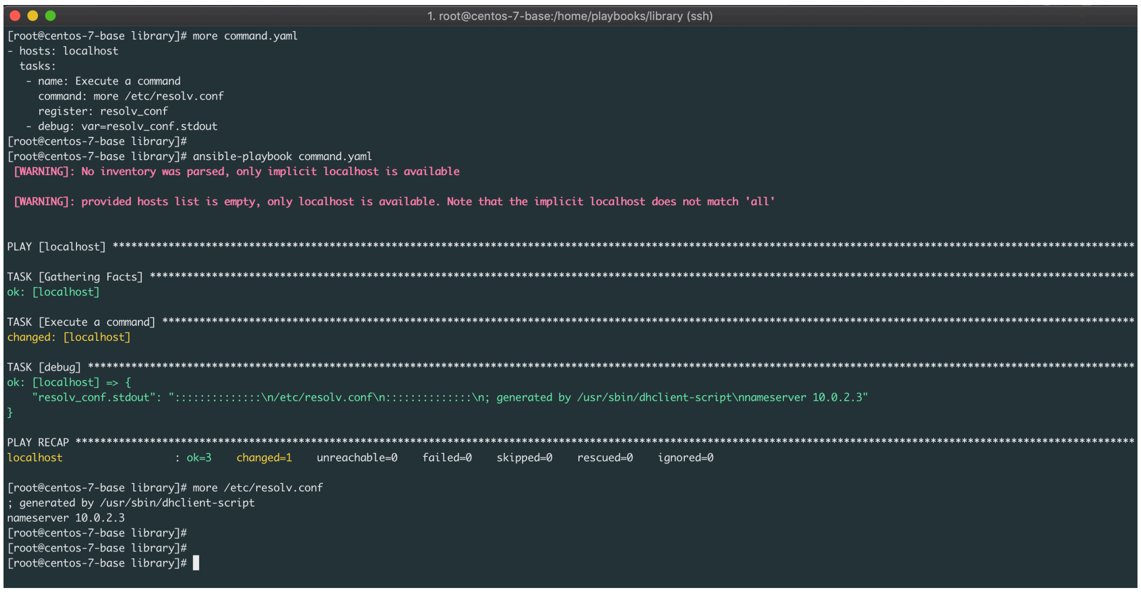


1. root@centos-7-base:/home/playbooks/library (ssh)

```
[root@centos-7-base library]# yum -q list installed mariadb &>/dev/null && echo "Installed" || echo "Not installed"
Not installed
[root@centos-7-base library]#
[root@centos-7-base library]# more yum.yaml
- hosts: localhost
  tasks:
   - name: Install Packages with Yum
     yum:
      name: "{{ item }}"
      state: present
     with_items:
      - httpd
      - mariadb
[root@centos-7-base library]#
[root@centos-7-base library]# ansible-playbook yum.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ********************************************************************************************************************************************************************

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

TASK [Install Packages with Yum] ****************************************************************************************************************************************************
[DEPRECATION WARNING]: Invoking "yum" only once while using a loop via squash_actions is deprecated. Instead of using a loop to supply multiple items and specifying `name: "{{ item
}}"`, please use `name: ['httpd', 'mariadb']` and remove the loop. This feature will be removed in version 2.11. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.
changed: [localhost] => (item=[u'httpd', u'mariadb'])

PLAY RECAP **************************************************************************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@centos-7-base library]# yum -q list installed mariadb &>/dev/null && echo "Installed" || echo "Not installed"
Installed
[root@centos-7-base library]# 
```

Command

Example playbook using the raw command module.



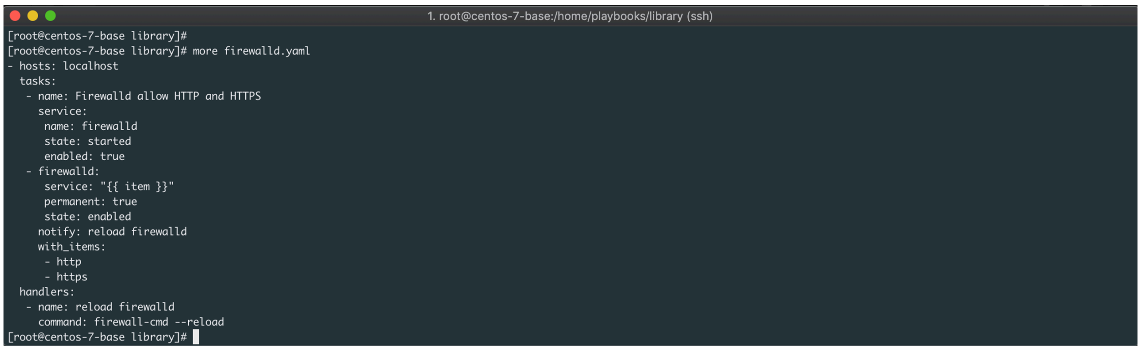
1. root@centos-7-base:/home/playbooks/library (ssh)

```
[root@centos-7-base library]# more command.yaml
- hosts: localhost
  tasks:
   - name: Execute a command
     command: more /etc/resolv.conf
     register: resolv_conf
   - debug: var=resolv_conf.stdout
[root@centos-7-base library]#
[root@centos-7-base library]# ansible-playbook command.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ********************************************************************************************************************************************************************

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

TASK [Execute a command] ************************************************************************************************************************************************************
changed: [localhost]

TASK [debug] ************************************************************************************************************************************************************************
ok: [localhost] => {
    "resolv_conf.stdout": "::::::::::::::\n/etc/resolv.conf\n::::::::::::::\n; generated by /usr/sbin/dhclient-script\nnameserver 10.0.2.3"
}

PLAY RECAP **************************************************************************************************************************************************************************
localhost                  : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@centos-7-base library]# more /etc/resolv.conf
; generated by /usr/sbin/dhclient-script
nameserver 10.0.2.3
[root@centos-7-base library]#
[root@centos-7-base library]#
[root@centos-7-base library]# 
```

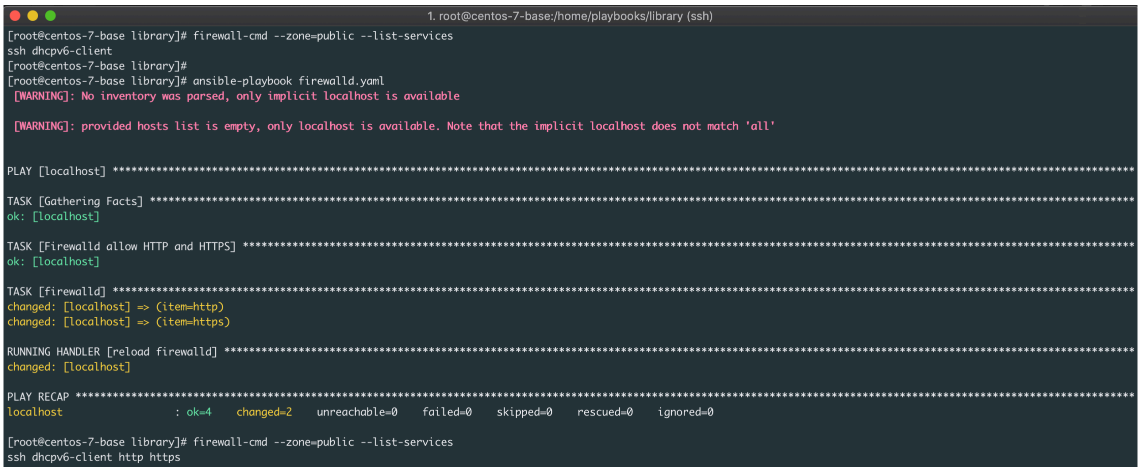
System

Firewalld

Example playbook using the firewalld module allowing ports 80 and 443.



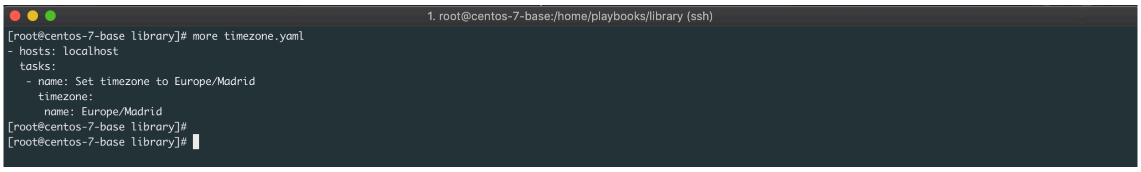
```
[root@centos-7-base library]#
[root@centos-7-base library]# more firewalld.yaml
- hosts: localhost
  tasks:
   - name: Firewalld allow HTTP and HTTPS
     service:
      name: firewalld
      state: started
      enabled: true
   - firewalld:
      service: "{{ item }}"
      permanent: true
      state: enabled
     notify: reload firewalld
     with_items:
      - http
      - https
  handlers:
   - name: reload firewalld
     command: firewall-cmd --reload
[root@centos-7-base library]#
```



```
[root@centos-7-base library]# firewall-cmd --zone=public --list-services
ssh dhcpv6-client
[root@centos-7-base library]#
[root@centos-7-base library]# ansible-playbook firewalld.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ****************************************************************************************************************************************************

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

TASK [Firewalld allow HTTP and HTTPS] *******************************************************************************************************************************
ok: [localhost]

TASK [firewalld] ****************************************************************************************************************************************************
changed: [localhost] => (item=http)
changed: [localhost] => (item=https)

RUNNING HANDLER [reload firewalld] **********************************************************************************************************************************
changed: [localhost]

PLAY RECAP **********************************************************************************************************************************************************
localhost                  : ok=4    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@centos-7-base library]# firewall-cmd --zone=public --list-services
ssh dhcpv6-client http https
```

Timezone

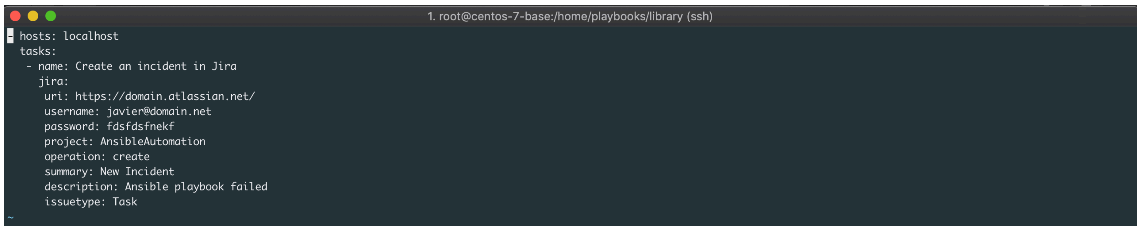
This playbook uses the timezone module for setting the timezone.



```
[root@centos-7-base library]# more timezone.yaml
- hosts: localhost
  tasks:
   - name: Set timezone to Europe/Madrid
     timezone:
      name: Europe/Madrid
[root@centos-7-base library]#
[root@centos-7-base library]#
```

Jira

This playbook creates a task in Jira.



```
- hosts: localhost
  tasks:
   - name: Create an incident in Jira
     jira:
      uri: https://domain.atlassian.net/
      username: javier@domain.net
      password: fdsfdsfnekf
      project: AnsibleAutomation
      operation: create
      summary: New Incident
      description: Ansible playbook failed
      issuetype: Task
~
```

Notifications

Slack

```
- name: Sending message to Slack Channel
   slack:
     token: '{{ slack_token }}'
     channel: "#companynameAnsible"
     domain: "companyname.slack.com"
     parse: "full"
     color: "good"
     msg: 'The changes is completed on {{ inventory_hostname }}.'
```

Twilio

```
- name: Send an SMS to multiple phone numbers when the change is
completed
   twilio:
     msg: The configuration change is completed!
     account_sid: XXXXXXXXXXX
     auth_token: XXXXXXXXXXX
     from_number: +34XXXXXXXXX
     to_number:
       - +34XXXXXXXX1
       - +34XXXXXXXX2
  delegate_to: localhost
```

NOTE: The delegate_to: localhost is often required when interworking with APIs.

connection: local runs your entire playbook play locally.

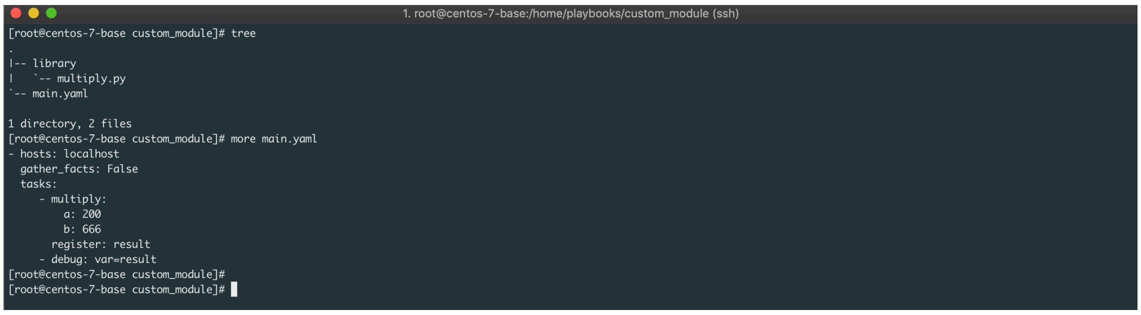
delegate_to: localhost runs a specific task on the localhost.

Custom Modules: Writing your own module

Folder Structure

Ansible requires that the module is stored at /library/custom_module_name.py at the same level where the playbook is located.

In the following example, the playbook is called "main.yaml" and the customer module "multiply.py" is located in the /library/ folder.



```
[root@centos-7-base custom_module]# tree
.
|-- library
|   `-- multiply.py
`-- main.yaml

1 directory, 2 files
[root@centos-7-base custom_module]# more main.yaml
- hosts: localhost
  gather_facts: False
  tasks:
     - multiply:
         a: 200
         b: 666
       register: result
     - debug: var=result
[root@centos-7-base custom_module]#
[root@centos-7-base custom_module]#
```

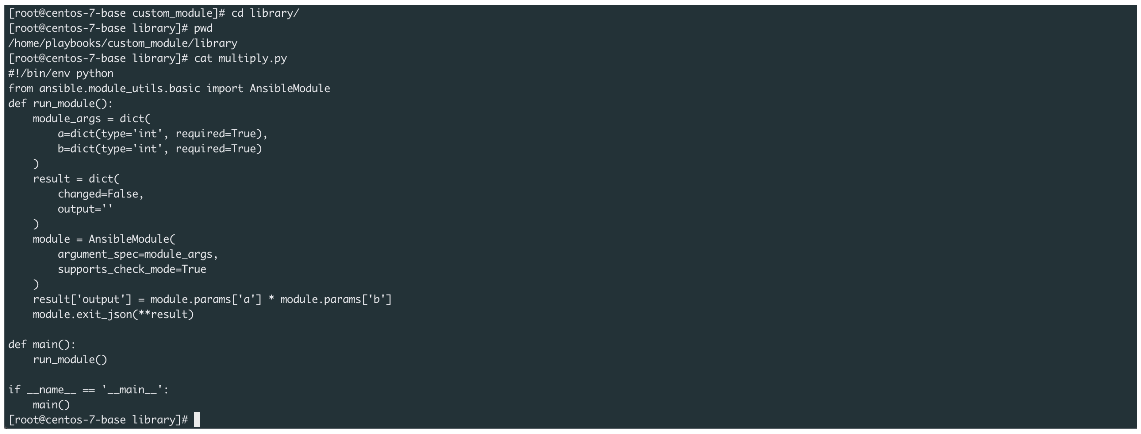
Ansible recommends to include strings for DOCUMENTATION and EXAMPLES at the top of our module code.

```
DOCUMENTATION = '''
---
module: multiply
short_description: Multiply  two given numbers
'''


EXAMPLES = '''
tasks:
- multiply:
    a: 200
    b: 10
  register: result
- debug: var=result
'''
```

The key part is to import the boilerplate code from ansible.module_utils.basic like this:

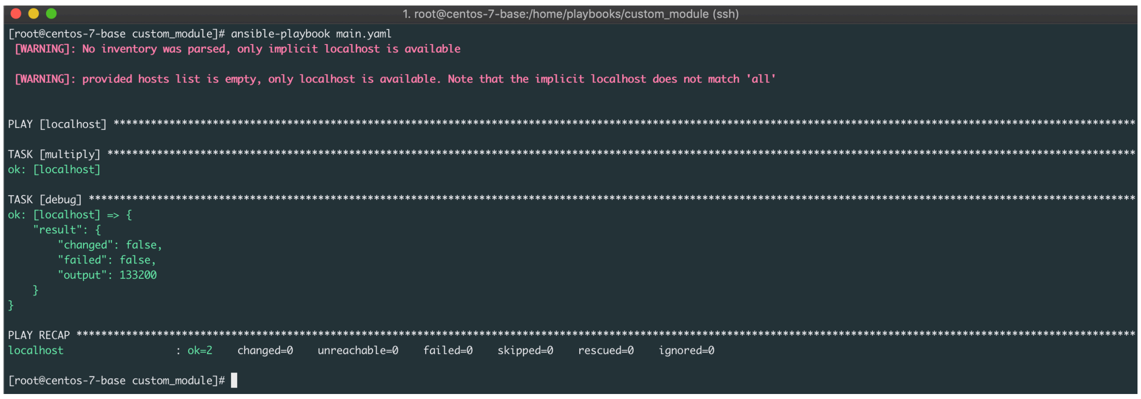
```
from ansible.module_utils.basic import AnsibleModule
if __name__ == '__main__':
    main()
```

The AnsibleModule provides lots of common code for handling returns, parses your arguments for you, and allows you to check inputs.



```
[root@centos-7-base custom_module]# cd library/
[root@centos-7-base library]# pwd
/home/playbooks/custom_module/library
[root@centos-7-base library]# cat multiply.py
#!/bin/env python
from ansible.module_utils.basic import AnsibleModule
def run_module():
    module_args = dict(
        a=dict(type='int', required=True),
        b=dict(type='int', required=True)
    )
    result = dict(
        changed=False,
        output=''
    )
    module = AnsibleModule(
        argument_spec=module_args,
        supports_check_mode=True
    )
    result['output'] = module.params['a'] * module.params['b']
    module.exit_json(**result)

def main():
    run_module()

if __name__ == '__main__':
    main()
[root@centos-7-base library]#
```

Let's run the playbook and check the result:



```
1. root@centos-7-base:/home/playbooks/custom_module (ssh)
[root@centos-7-base custom_module]# ansible-playbook main.yaml
 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'


PLAY [localhost] ***********************************************************************************************************************************

TASK [multiply] ************************************************************************************************************************************
ok: [localhost]

TASK [debug] ***************************************************************************************************************************************
ok: [localhost] => {
    "result": {
        "changed": false,
        "failed": false,
        "output": 133200
    }
}

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

[root@centos-7-base custom_module]#
```

Roles

Roles are nothing but directories laid out in a specific manner. Roles follow predefined directory layout conventions and expect each component to be in the path meant for it. A role directory structure contains the directories below. Each directory must contain a main.yml which is the default file.

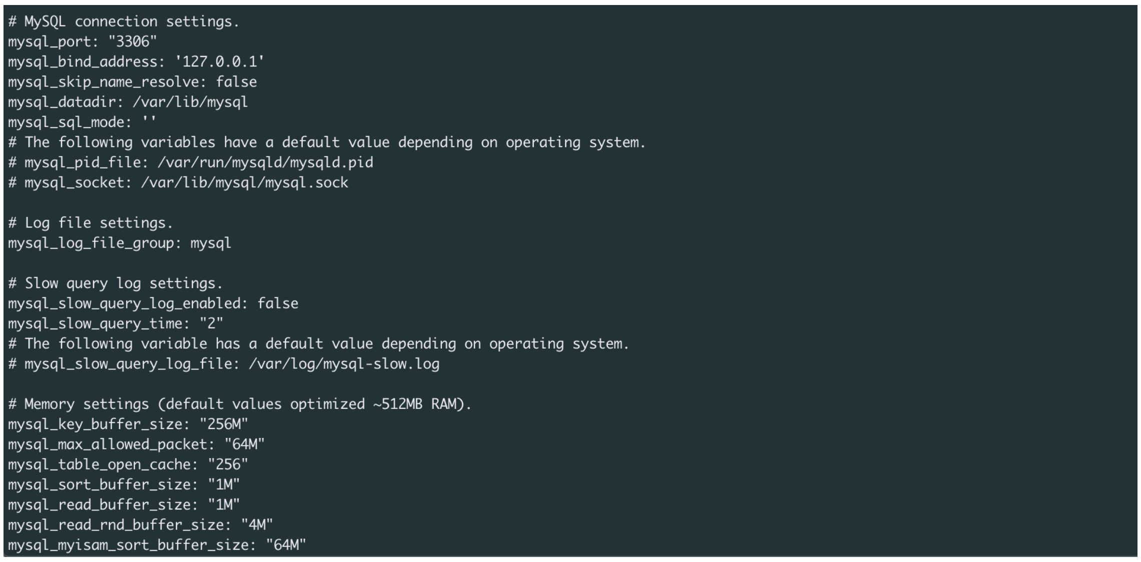
[root@centos-7-base unxnn.mysql]# ls -lrt
total 12
-rw-rw-r--. 1 root root 7657 May 8 11:03 README.md
-rw-rw-r--. 1 root root 1061 May 8 11:03 LICENSE
drwxr-xr-x. 2 root root 22 Jun 14 12:04 handlers
drwxr-xr-x. 2 root root 22 Jun 14 12:04 defaults
drwxr-xr-x. 3 root root 21 Jun 14 12:04 molecule
drwxr-xr-x. 2 root root 67 Jun 14 12:04 templates
drwxr-xr-x. 2 root root 143 Jun 14 12:04 tasks
drwxr-xr-x. 2 root root 64 Jun 14 12:04 vars
drwxr-xr-x. 2 root root 39 Jun 14 12:04 tests
drwxr-xr-x. 2 root root 50 Jun 14 12:04 meta

Defaults: default variables for the role
Vars: variables for the role
Tasks: the main list of steps to be executed by the role
Files: contains files which we want to be transferred to the host
Templates: file template which supports modifications from the role
Meta: contains metadata of role such as dependencies
Handlers: handlers which can be invoked by notify directives

```
-- unxnn.mysql
   |-- defaults
   |   `-- main.yml
   |-- handlers
   |   `-- main.yml
   |-- LICENSE
   |-- meta
   |   `-- main.yml
   |-- molecule
   |   `-- default
   |       |-- molecule.yml
   |       |-- playbook.yml
   |       |-- tests
   |       |   `-- test_default.py
   |       `-- yaml-lint.yml
   |-- README.md
   |-- tasks
   |   |-- configure.yml
   |   |-- main.yml
   |   |-- secure-installation.yml
   |   |-- setup-Debian.yml
   |   |-- setup-RedHat.yml
   |   `-- variables.yml
   |-- templates
   |   |-- my.cnf.j2
   |   |-- root-my.cnf.j2
   |   `-- user-my.cnf.j2
   |-- tests
   |   |-- inventory
   |   `-- test.yml
```

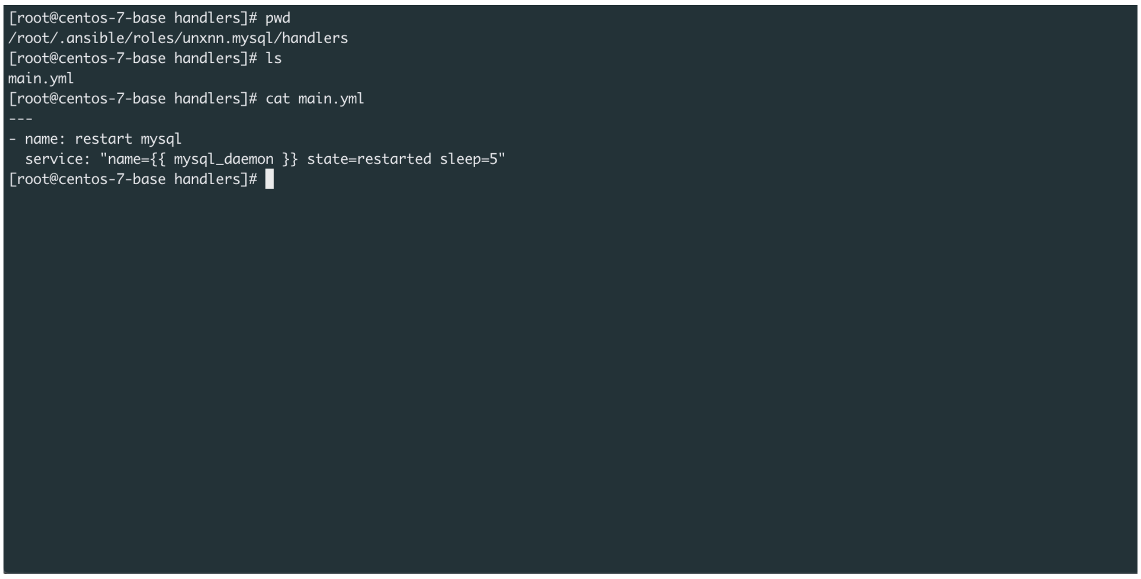
Let's take a glance of these files:

/defaults/main.yml



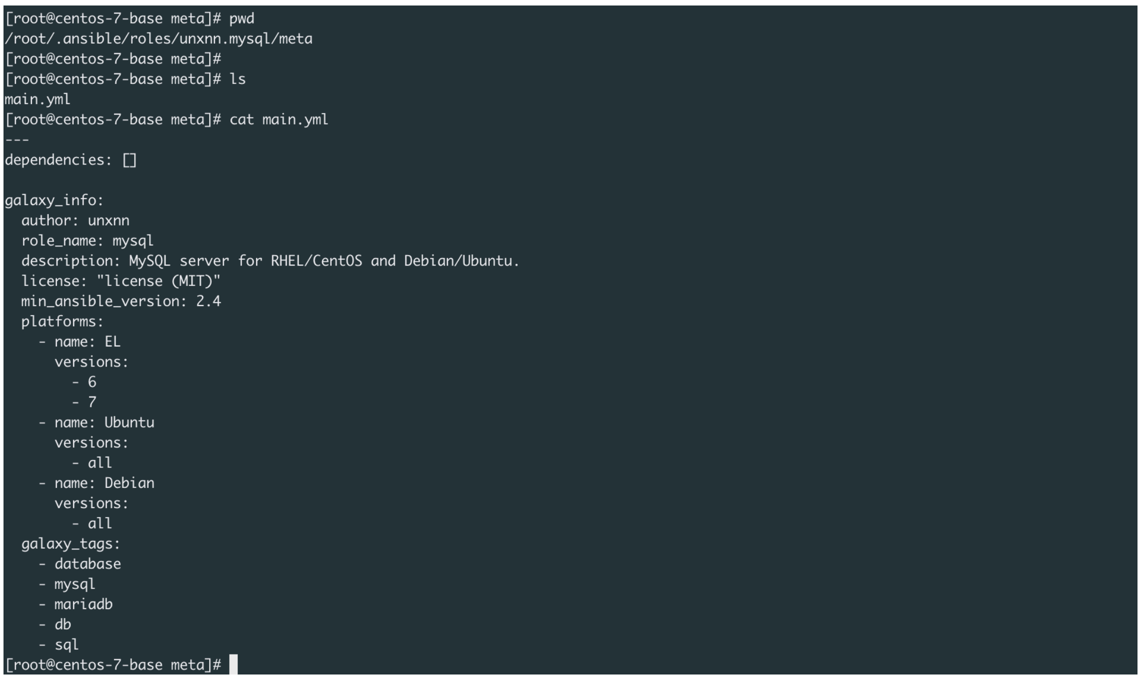
```
# MySQL connection settings.
mysql_port: "3306"
mysql_bind_address: '127.0.0.1'
mysql_skip_name_resolve: false
mysql_datadir: /var/lib/mysql
mysql_sql_mode: ''
# The following variables have a default value depending on operating system.
# mysql_pid_file: /var/run/mysqld/mysqld.pid
# mysql_socket: /var/lib/mysql/mysql.sock

# Log file settings.
mysql_log_file_group: mysql

# Slow query log settings.
mysql_slow_query_log_enabled: false
mysql_slow_query_time: "2"
# The following variable has a default value depending on operating system.
# mysql_slow_query_log_file: /var/log/mysql-slow.log

# Memory settings (default values optimized ~512MB RAM).
mysql_key_buffer_size: "256M"
mysql_max_allowed_packet: "64M"
mysql_table_open_cache: "256"
mysql_sort_buffer_size: "1M"
mysql_read_buffer_size: "1M"
mysql_read_rnd_buffer_size: "4M"
mysql_myisam_sort_buffer_size: "64M"
```

/handlers/mail.yml



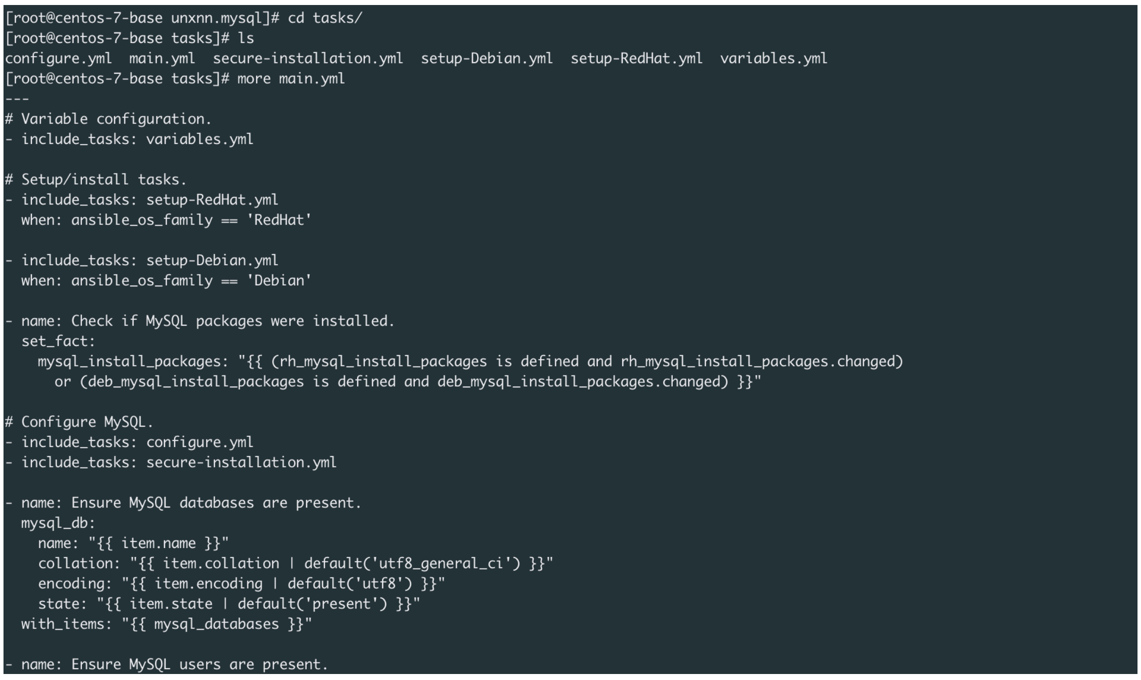
```
[root@centos-7-base handlers]# pwd
/root/.ansible/roles/unxnn.mysql/handlers
[root@centos-7-base handlers]# ls
main.yml
[root@centos-7-base handlers]# cat main.yml
---
- name: restart mysql
  service: "name={{ mysql_daemon }} state=restarted sleep=5"
[root@centos-7-base handlers]#
```

/meta/main.yml



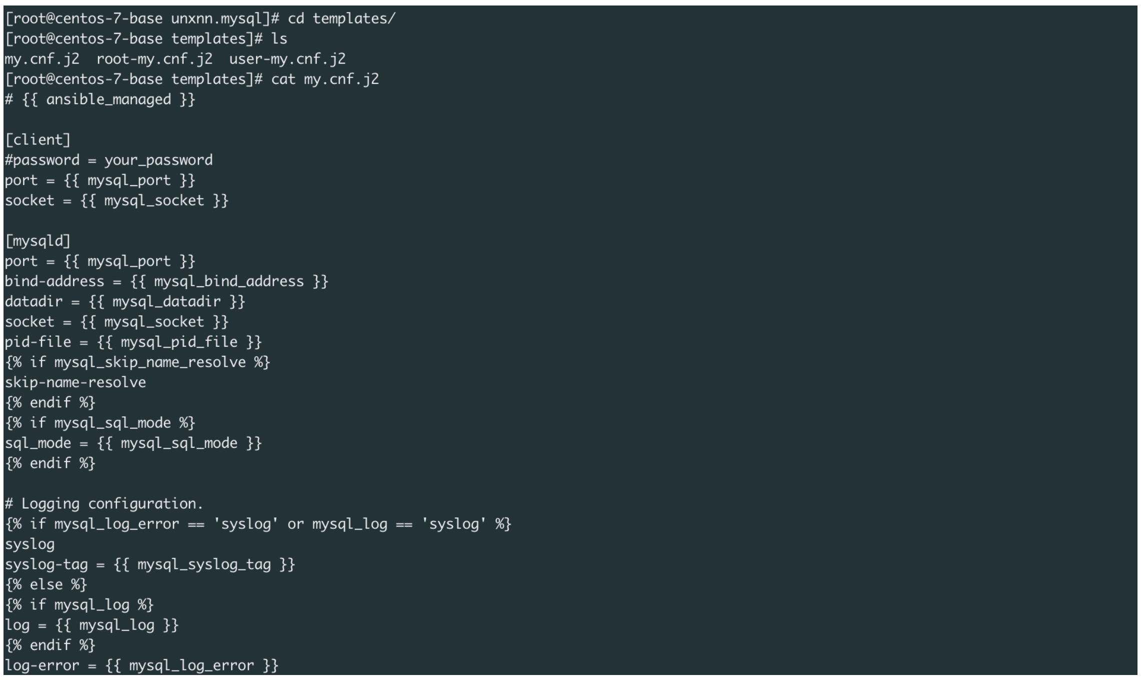
```
[root@centos-7-base meta]# pwd
/root/.ansible/roles/unxnn.mysql/meta
[root@centos-7-base meta]#
[root@centos-7-base meta]# ls
main.yml
[root@centos-7-base meta]# cat main.yml
---
dependencies: []

galaxy_info:
  author: unxnn
  role_name: mysql
  description: MySQL server for RHEL/CentOS and Debian/Ubuntu.
  license: "license (MIT)"
  min_ansible_version: 2.4
  platforms:
    - name: EL
      versions:
        - 6
        - 7
    - name: Ubuntu
      versions:
        - all
    - name: Debian
      versions:
        - all
  galaxy_tags:
    - database
    - mysql
    - mariadb
    - db
    - sql
[root@centos-7-base meta]#
```

/tasks/main.yml



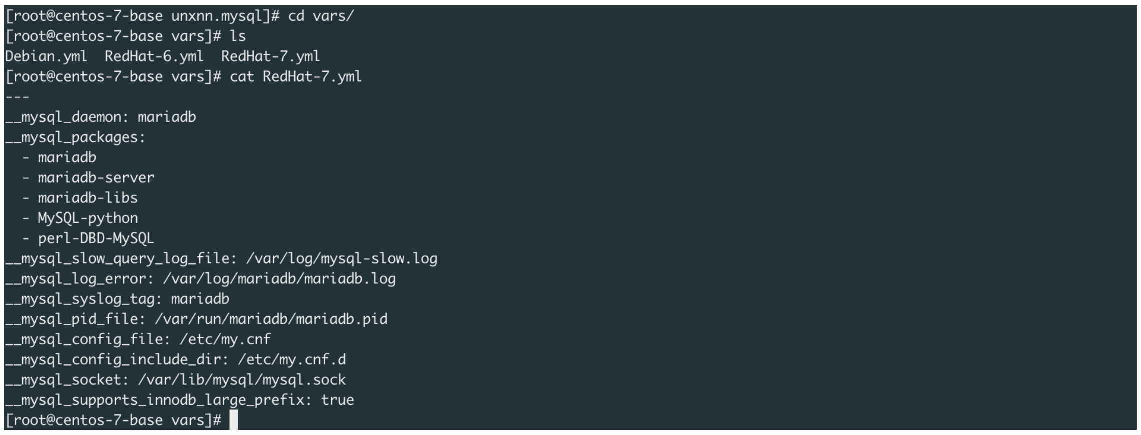
```
[root@centos-7-base unxnn.mysql]# cd tasks/
[root@centos-7-base tasks]# ls
configure.yml  main.yml  secure-installation.yml  setup-Debian.yml  setup-RedHat.yml  variables.yml
[root@centos-7-base tasks]# more main.yml
---
# Variable configuration.
- include_tasks: variables.yml

# Setup/install tasks.
- include_tasks: setup-RedHat.yml
  when: ansible_os_family == 'RedHat'

- include_tasks: setup-Debian.yml
  when: ansible_os_family == 'Debian'

- name: Check if MySQL packages were installed.
  set_fact:
    mysql_install_packages: "{{ (rh_mysql_install_packages is defined and rh_mysql_install_packages.changed)
      or (deb_mysql_install_packages is defined and deb_mysql_install_packages.changed) }}"

# Configure MySQL.
- include_tasks: configure.yml
- include_tasks: secure-installation.yml

- name: Ensure MySQL databases are present.
  mysql_db:
    name: "{{ item.name }}"
    collation: "{{ item.collation | default('utf8_general_ci') }}"
    encoding: "{{ item.encoding | default('utf8') }}"
    state: "{{ item.state | default('present') }}"
  with_items: "{{ mysql_databases }}"

- name: Ensure MySQL users are present.
```

/templates/



```
[root@centos-7-base unxnn.mysql]# cd templates/
[root@centos-7-base templates]# ls
my.cnf.j2  root-my.cnf.j2  user-my.cnf.j2
[root@centos-7-base templates]# cat my.cnf.j2
# {{ ansible_managed }}

[client]
#password = your_password
port = {{ mysql_port }}
socket = {{ mysql_socket }}

[mysqld]
port = {{ mysql_port }}
bind-address = {{ mysql_bind_address }}
datadir = {{ mysql_datadir }}
socket = {{ mysql_socket }}
pid-file = {{ mysql_pid_file }}
{% if mysql_skip_name_resolve %}
skip-name-resolve
{% endif %}
{% if mysql_sql_mode %}
sql_mode = {{ mysql_sql_mode }}
{% endif %}

# Logging configuration.
{% if mysql_log_error == 'syslog' or mysql_log == 'syslog' %}
syslog
syslog-tag = {{ mysql_syslog_tag }}
{% else %}
{% if mysql_log %}
log = {{ mysql_log }}
{% endif %}
log-error = {{ mysql_log_error }}
```

/vars/



```
[root@centos-7-base unxnn.mysql]# cd vars/
[root@centos-7-base vars]# ls
Debian.yml  RedHat-6.yml  RedHat-7.yml
[root@centos-7-base vars]# cat RedHat-7.yml
---
__mysql_daemon: mariadb
__mysql_packages:
  - mariadb
  - mariadb-server
  - mariadb-libs
  - MySQL-python
  - perl-DBD-MySQL
__mysql_slow_query_log_file: /var/log/mysql-slow.log
__mysql_log_error: /var/log/mariadb/mariadb.log
__mysql_syslog_tag: mariadb
__mysql_pid_file: /var/run/mariadb/mariadb.pid
__mysql_config_file: /etc/my.cnf
__mysql_config_include_dir: /etc/my.cnf.d
__mysql_socket: /var/lib/mysql/mysql.sock
__mysql_supports_innodb_large_prefix: true
[root@centos-7-base vars]#
```

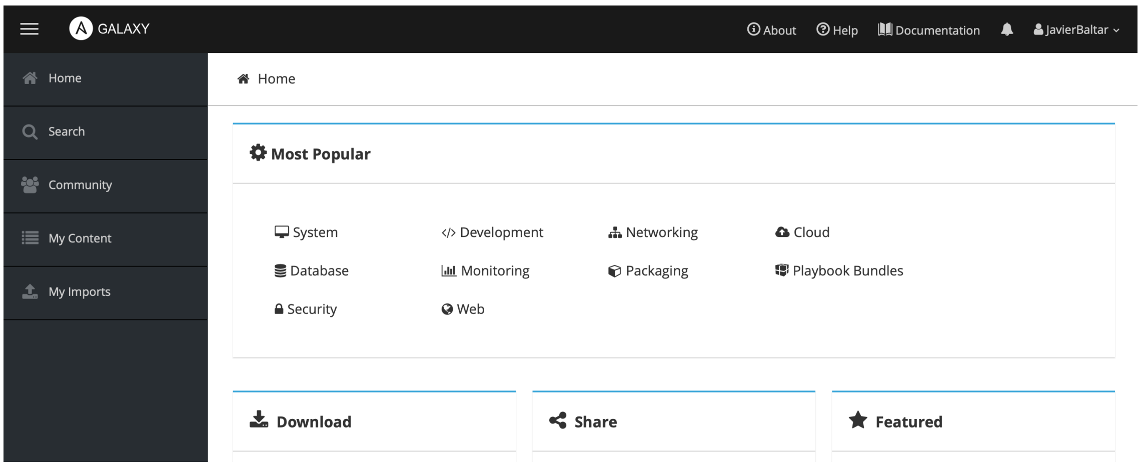
Molecule

Please refer to the following link: https://molecule.readthedocs.io/en/stable/

Ansible Galaxy

Ansible Galaxy refers to the Ansible website where users can share roles, and to a command line tool for installing, creating and managing roles.

https://galaxy.ansible.com/home



The ansible-galaxy command comes bundled with Ansible, and you can use it to install roles from Galaxy or directly from a git based SCM. You can also use it to create a new role, remove roles, or perform tasks on the Galaxy website.

The command line tool by default communicates with the Galaxy website API using the server address https://galaxy.ansible.com. Since the Galaxy project is an open source project, you may be running your own internal Galaxy server and wish to override the default server address. You can do this using the –server option or by setting the Galaxy server value in your ansible.cfg file. For information on setting the value in ansible.cfg visit Galaxy Settings.

Search

Search the Galaxy database by tags, platforms, author and multiple keywords. For example:

```
$ ansible-galaxy search elasticsearch
```

List installed roles

Use list to show the name and version of each role installed in the *roles_path*.

```
$ ansible-galaxy list

- chouseknecht.role-install_mongod, master
- chouseknecht.test-role-1, v1.0.2
- chrismeyersfsu.role-iptables, master
- chrismeyersfsu.role-required_vars, master
```

Remove installed roles

Use remove to delete a role from *roles_path*:

Import a role

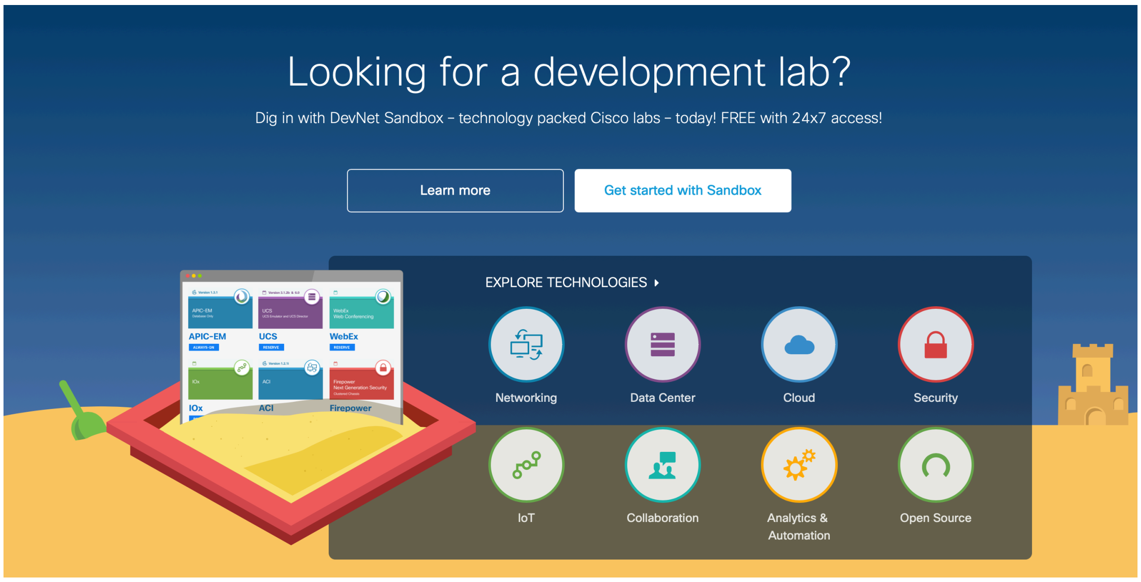
The import command requires that you first authenticate using the login command. Once authenticated you can import any GitHub repository that you own or have been granted access.

Use the following to import to role:

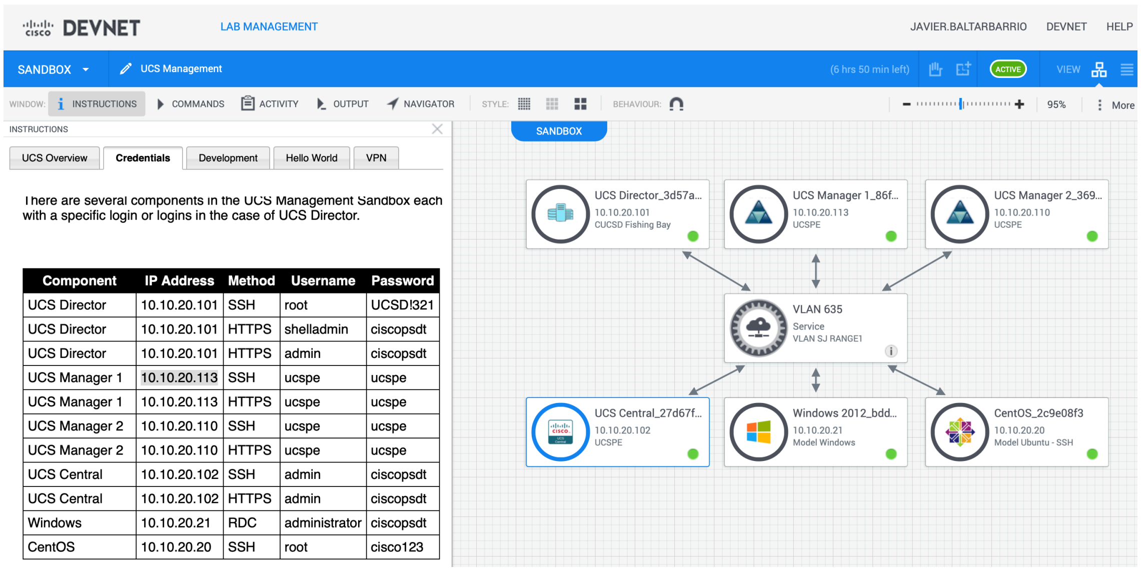
$ ansible-galaxy import github_user github_repo

UCS Manager Management

Some of the examples below are executed upon Cisco Devnet sandboxes: https://developer.cisco.com/site/sandbox/



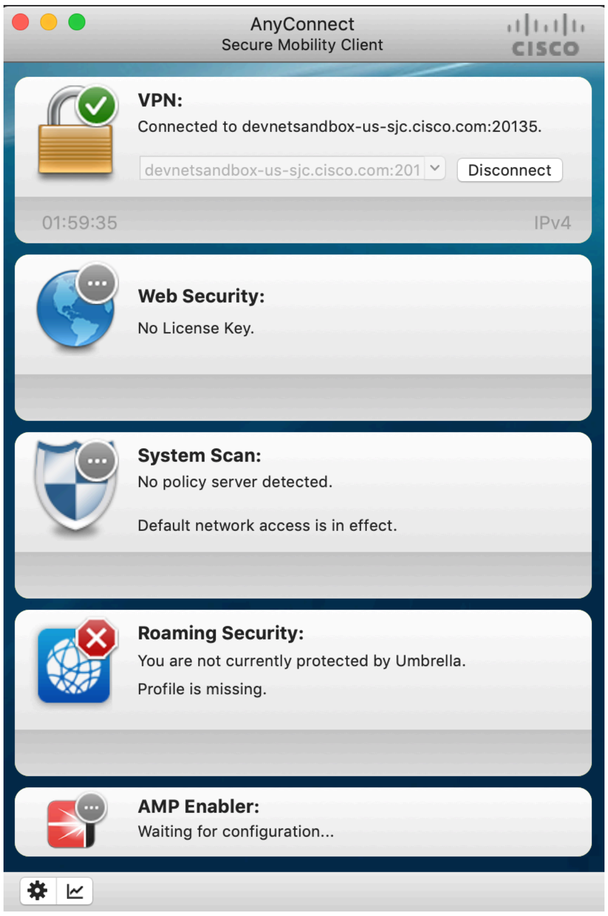
Once your lab is reserved, the VPN credentials are sent by email as shown below:



There are several components in the UCS Management Sandbox each with a specific login or logins in the case of UCS Director.

Component	IP Address	Method	Username	Password
UCS Director	10.10.20.101	SSH	root	UCSD!321
UCS Director	10.10.20.101	HTTPS	shelladmin	ciscopsdt
UCS Director	10.10.20.101	HTTPS	admin	ciscopsdt
UCS Manager 1	10.10.20.113	SSH	ucspe	ucspe
UCS Manager 1	10.10.20.113	HTTPS	ucspe	ucspe
UCS Manager 2	10.10.20.110	SSH	ucspe	ucspe
UCS Manager 2	10.10.20.110	HTTPS	ucspe	ucspe
UCS Central	10.10.20.102	SSH	admin	ciscopsdt
UCS Central	10.10.20.102	HTTPS	admin	ciscopsdt
Windows	10.10.20.21	RDC	administrator	ciscopsdt
CentOS	10.10.20.20	SSH	root	cisco123

You have to set up the VPN connection:



VPN:
Connected to devnetsandbox-us-sjc.cisco.com:20135.
devnetsandbox-us-sjc.cisco.com:201
01:59:35 IPv4

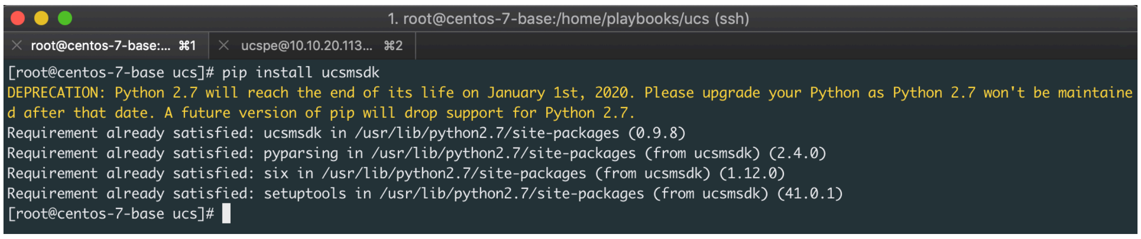
Web Security:
No License Key.

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

Roaming Security:
You are not currently protected by Umbrella.
Profile is missing.

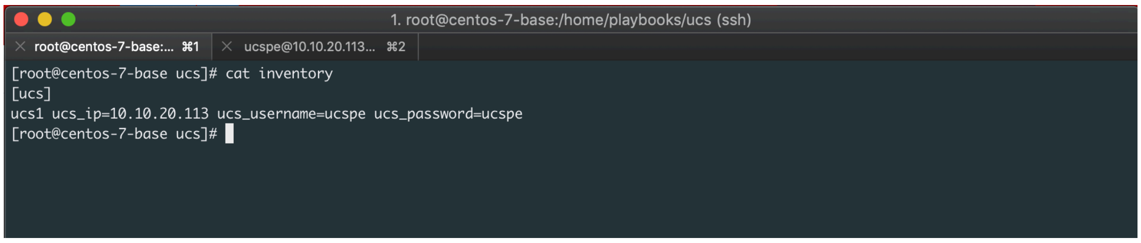
AMP Enabler:
Waiting for configuration...

In order to interwork with UCS Manager, the "ucsmsdk" must be installed:


1. root@centos-7-base:/home/playbooks/ucs (ssh)
root@centos-7-base:... ⌘1
ucspe@10.10.20.113... ⌘2

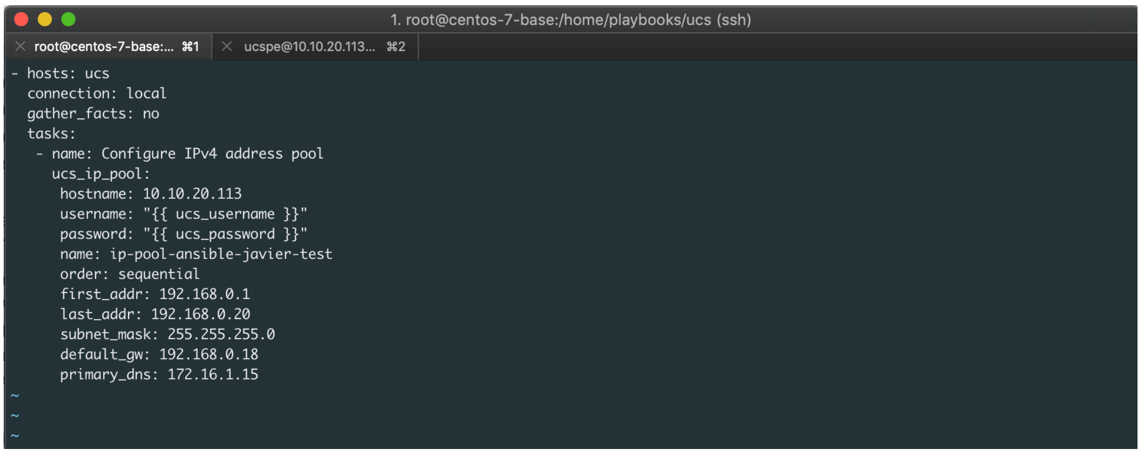
```
[root@centos-7-base ucs]# pip install ucsmsdk
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Please upgrade your Python as Python 2.7 won't be maintaine
d after that date. A future version of pip will drop support for Python 2.7.
Requirement already satisfied: ucsmsdk in /usr/lib/python2.7/site-packages (0.9.8)
Requirement already satisfied: pyparsing in /usr/lib/python2.7/site-packages (from ucsmsdk) (2.4.0)
Requirement already satisfied: six in /usr/lib/python2.7/site-packages (from ucsmsdk) (1.12.0)
Requirement already satisfied: setuptools in /usr/lib/python2.7/site-packages (from ucsmsdk) (41.0.1)
[root@centos-7-base ucs]#
```

For testing purposes, I have created the inventory file below:


1. root@centos-7-base:/home/playbooks/ucs (ssh)
root@centos-7-base:... ⌘1
ucspe@10.10.20.113... ⌘2

```
[root@centos-7-base ucs]# cat inventory
[ucs]
ucs1 ucs_ip=10.10.20.113 ucs_username=ucspe ucs_password=ucspe
[root@centos-7-base ucs]#
```

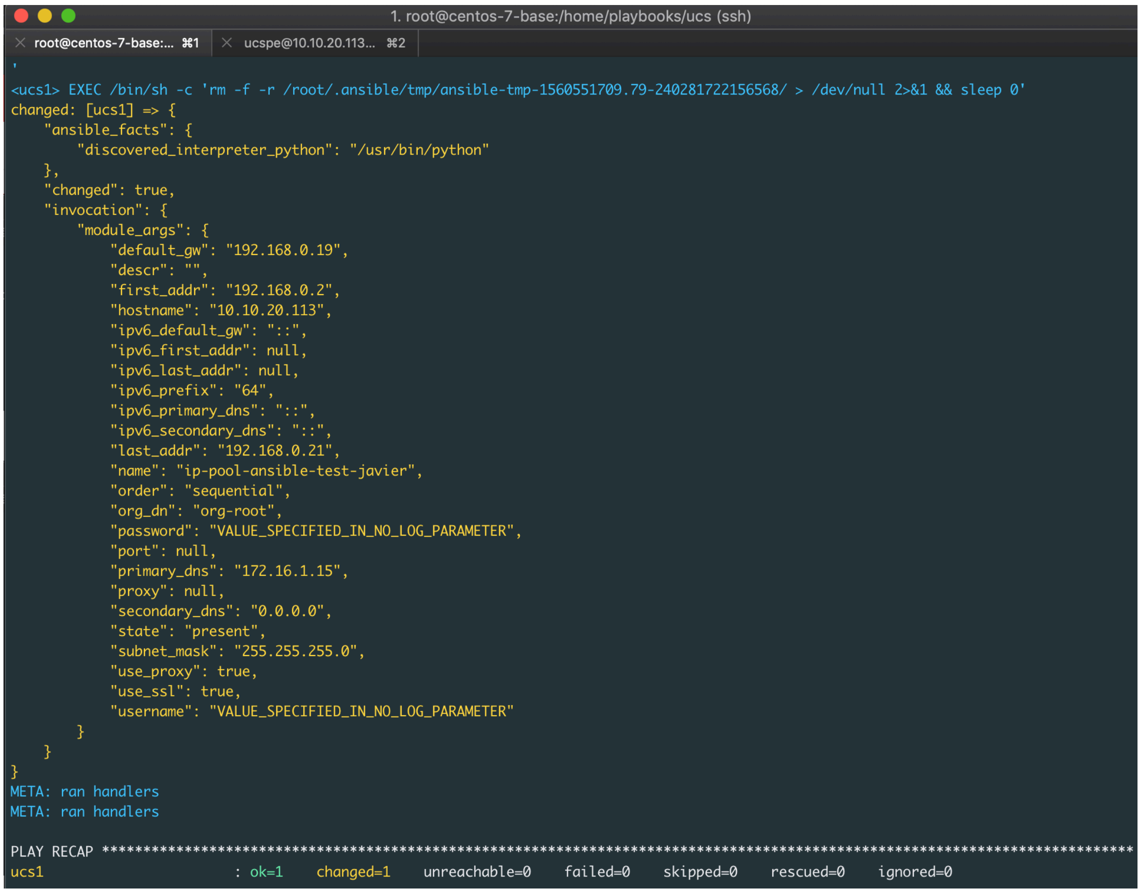
The following playbook configures an address pool in UCS.


1. root@centos-7-base:/home/playbooks/ucs (ssh)
root@centos-7-base:... ⌘1
ucspe@10.10.20.113... ⌘2

```
- hosts: ucs
  connection: local
  gather_facts: no
  tasks:
   - name: Configure IPv4 address pool
     ucs_ip_pool:
       hostname: 10.10.20.113
       username: "{{ ucs_username }}"
       password: "{{ ucs_password }}"
       name: ip-pool-ansible-javier-test
       order: sequential
       first_addr: 192.168.0.1
       last_addr: 192.168.0.20
       subnet_mask: 255.255.255.0
       default_gw: 192.168.0.18
       primary_dns: 172.16.1.15
~
~
~
```

Running the playbook:

# ansible-playbook create-pool-yaml -i inventory -vvv

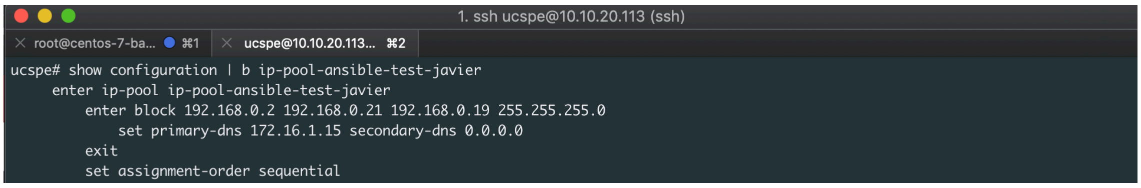


1. root@centos-7-base:/home/playbooks/ucs (ssh)

root@centos-7-base:... ⌘1 | ucspe@10.10.20.113... ⌘2

```
'
<ucs1> EXEC /bin/sh -c 'rm -f -r /root/.ansible/tmp/ansible-tmp-1560551709.79-240281722156568/ > /dev/null 2>&1 && sleep 0'
changed: [ucs1] => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "invocation": {
        "module_args": {
            "default_gw": "192.168.0.19",
            "descr": "",
            "first_addr": "192.168.0.2",
            "hostname": "10.10.20.113",
            "ipv6_default_gw": "::",
            "ipv6_first_addr": null,
            "ipv6_last_addr": null,
            "ipv6_prefix": "64",
            "ipv6_primary_dns": "::",
            "ipv6_secondary_dns": "::",
            "last_addr": "192.168.0.21",
            "name": "ip-pool-ansible-test-javier",
            "order": "sequential",
            "org_dn": "org-root",
            "password": "VALUE_SPECIFIED_IN_NO_LOG_PARAMETER",
            "port": null,
            "primary_dns": "172.16.1.15",
            "proxy": null,
            "secondary_dns": "0.0.0.0",
            "state": "present",
            "subnet_mask": "255.255.255.0",
            "use_proxy": true,
            "use_ssl": true,
            "username": "VALUE_SPECIFIED_IN_NO_LOG_PARAMETER"
        }
    }
}
META: ran handlers
META: ran handlers

PLAY RECAP *****************************************************************************************************************************
ucs1                       : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

You can establish an SSH connection towards UCS Manager and check that the pool is created:

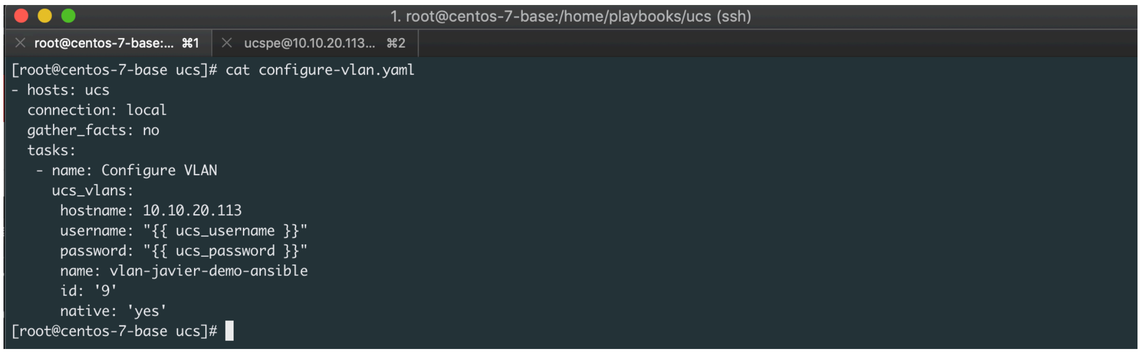


1. ssh ucspe@10.10.20.113 (ssh)

root@centos-7-ba... ⌘1 | ucspe@10.10.20.113... ⌘2

```
ucspe# show configuration | b ip-pool-ansible-test-javier
     enter ip-pool ip-pool-ansible-test-javier
         enter block 192.168.0.2 192.168.0.21 192.168.0.19 255.255.255.0
             set primary-dns 172.16.1.15 secondary-dns 0.0.0.0
         exit
         set assignment-order sequential
```

Similarly, the following playbook creates a VLAN.

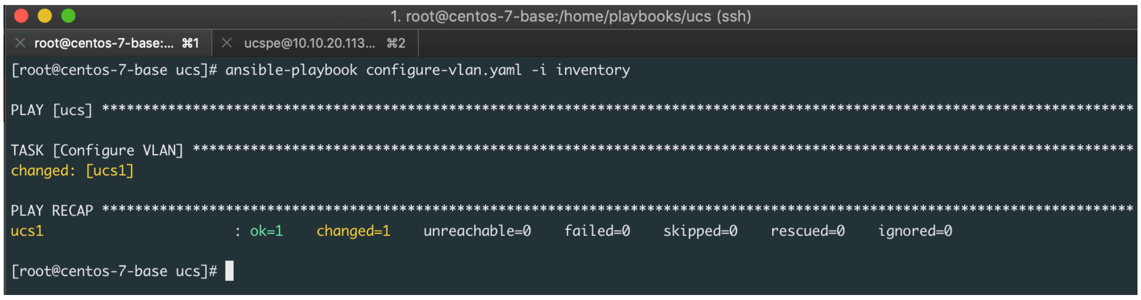


1. root@centos-7-base:/home/playbooks/ucs (ssh)

root@centos-7-base:... ⌘1 | ucspe@10.10.20.113... ⌘2

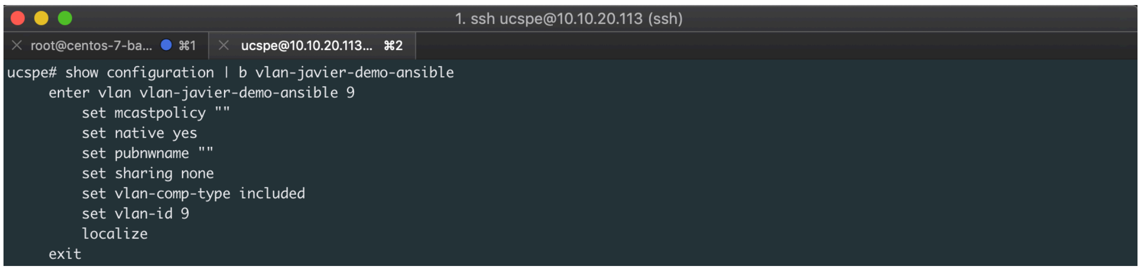
```
[root@centos-7-base ucs]# cat configure-vlan.yaml
- hosts: ucs
  connection: local
  gather_facts: no
  tasks:
    - name: Configure VLAN
      ucs_vlans:
        hostname: 10.10.20.113
        username: "{{ ucs_username }}"
        password: "{{ ucs_password }}"
        name: vlan-javier-demo-ansible
        id: '9'
        native: 'yes'
[root@centos-7-base ucs]# █
```

Running the playbook:


1. root@centos-7-base:/home/playbooks/ucs (ssh)
root@centos-7-base:... ⌘1
ucspe@10.10.20.113... ⌘2

```
[root@centos-7-base ucs]# ansible-playbook configure-vlan.yaml -i inventory

PLAY [ucs] ****************************************************************************************************

TASK [Configure VLAN] *****************************************************************************************
changed: [ucs1]

PLAY RECAP ****************************************************************************************************
ucs1                       : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@centos-7-base ucs]#
```

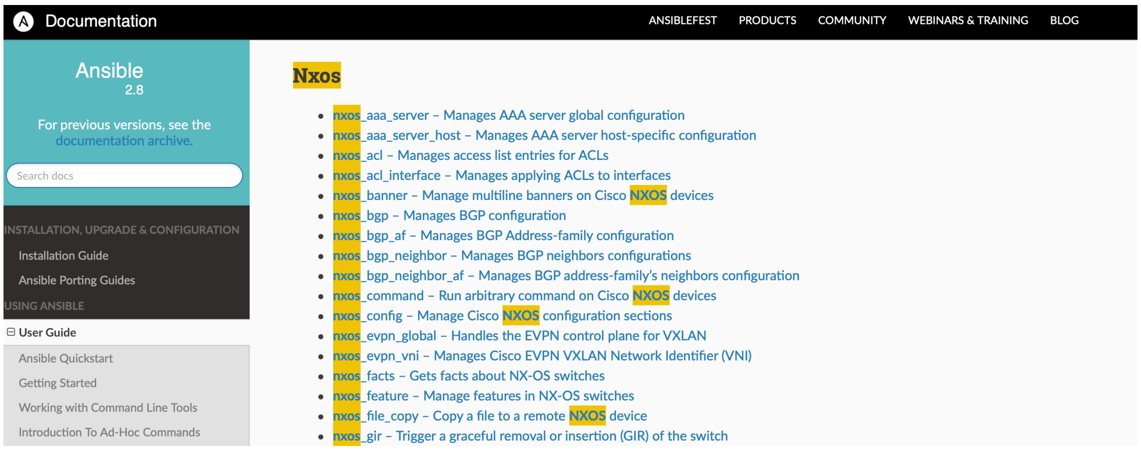
Validating that the VLAN is created:



```
ucspe# show configuration | b vlan-javier-demo-ansible
     enter vlan vlan-javier-demo-ansible 9
         set mcastpolicy ""
         set native yes
         set pubnwname ""
         set sharing none
         set vlan-comp-type included
         set vlan-id 9
         localize
     exit
```

Cisco NXOS Management

Please refer to: https://docs.ansible.com/ansible/latest/modules/list_of_network_modules.html?highlight=nxos



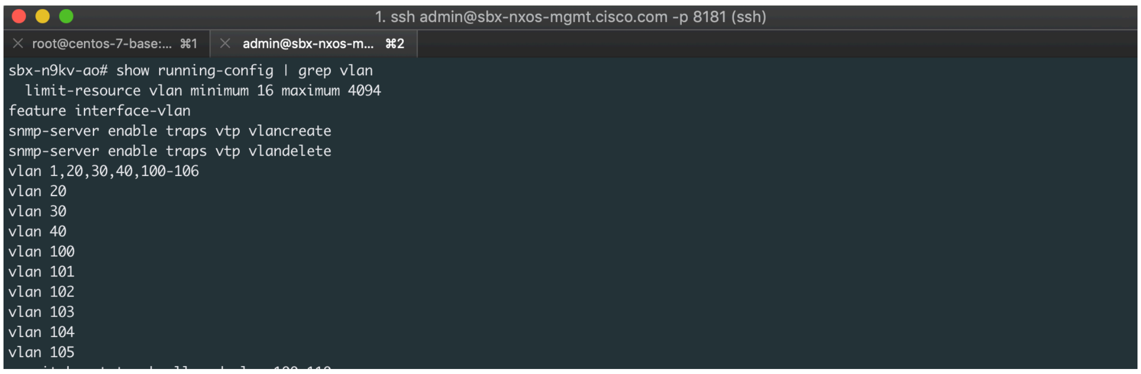
Nxos

- nxos_aaa_server – Manages AAA server global configuration
- nxos_aaa_server_host – Manages AAA server host-specific configuration
- nxos_acl – Manages access list entries for ACLs
- nxos_acl_interface – Manages applying ACLs to interfaces
- nxos_banner – Manage multiline banners on Cisco NXOS devices
- nxos_bgp – Manages BGP configuration
- nxos_bgp_af – Manages BGP Address-family configuration
- nxos_bgp_neighbor – Manages BGP neighbors configurations
- nxos_bgp_neighbor_af – Manages BGP address-family's neighbors configuration
- nxos_command – Run arbitrary command on Cisco NXOS devices
- nxos_config – Manage Cisco NXOS configuration sections
- nxos_evpn_global – Handles the EVPN control plane for VXLAN
- nxos_evpn_vni – Manages Cisco EVPN VXLAN Network Identifier (VNI)
- nxos_facts – Gets facts about NX-OS switches
- nxos_feature – Manage features in NX-OS switches
- nxos_file_copy – Copy a file to a remote NXOS device
- nxos_gir – Trigger a graceful removal or insertion (GIR) of the switch

Playbooks

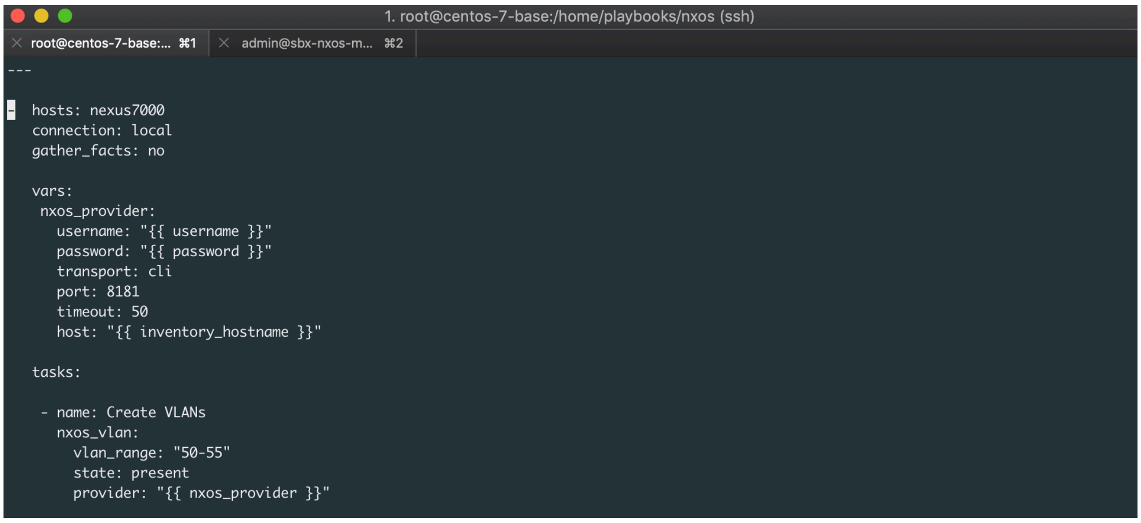
Create VLANs

This is the initial NXOS device configuration:


1. ssh admin@sbx-nxos-mgmt.cisco.com -p 8181 (ssh)
root@centos-7-base:... ⌘1
admin@sbx-nxos-m... ⌘2

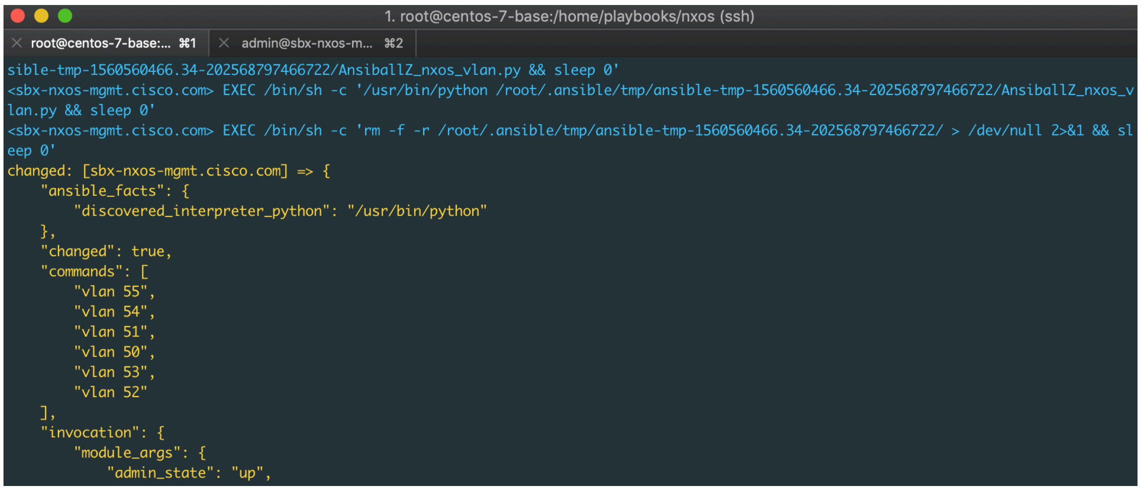
```
sbx-n9kv-ao# show running-config | grep vlan
  limit-resource vlan minimum 16 maximum 4094
feature interface-vlan
snmp-server enable traps vtp vlancreate
snmp-server enable traps vtp vlandelete
vlan 1,20,30,40,100-106
vlan 20
vlan 30
vlan 40
vlan 100
vlan 101
vlan 102
vlan 103
vlan 104
vlan 105
```

The playbook below creates a range of VLANs:


1. root@centos-7-base:/home/playbooks/nxos (ssh)
root@centos-7-base:... ⌘1
admin@sbx-nxos-m... ⌘2

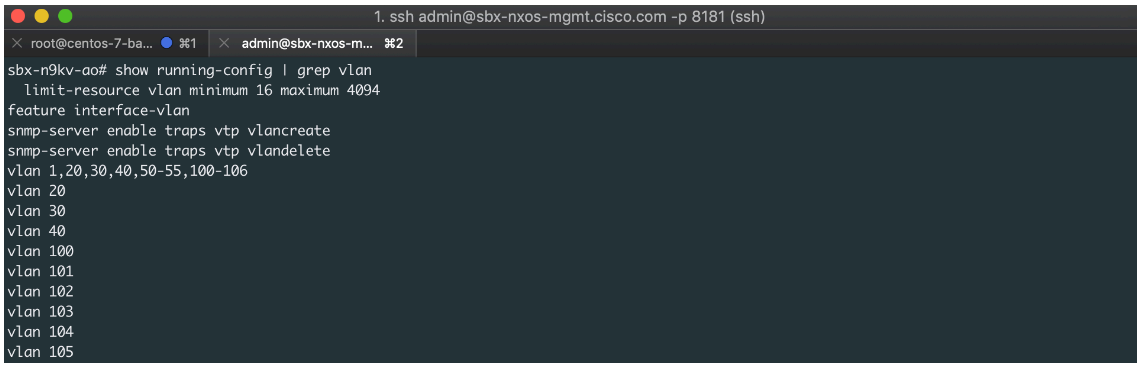
```
---

-  hosts: nexus7000
   connection: local
   gather_facts: no

   vars:
    nxos_provider:
      username: "{{ username }}"
      password: "{{ password }}"
      transport: cli
      port: 8181
      timeout: 50
      host: "{{ inventory_hostname }}"

   tasks:

    - name: Create VLANs
      nxos_vlan:
        vlan_range: "50-55"
        state: present
        provider: "{{ nxos_provider }}"
```

Running the playbook:


1. root@centos-7-base:/home/playbooks/nxos (ssh)
root@centos-7-base:... ⌘1
admin@sbx-nxos-m... ⌘2

```
sible-tmp-1560560466.34-202568797466722/AnsiballZ_nxos_vlan.py && sleep 0'
<sbx-nxos-mgmt.cisco.com> EXEC /bin/sh -c '/usr/bin/python /root/.ansible/tmp/ansible-tmp-1560560466.34-202568797466722/AnsiballZ_nxos_v
lan.py && sleep 0'
<sbx-nxos-mgmt.cisco.com> EXEC /bin/sh -c 'rm -f -r /root/.ansible/tmp/ansible-tmp-1560560466.34-202568797466722/ > /dev/null 2>&1 && sl
eep 0'
changed: [sbx-nxos-mgmt.cisco.com] => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "commands": [
        "vlan 55",
        "vlan 54",
        "vlan 51",
        "vlan 50",
        "vlan 53",
        "vlan 52"
    ],
    "invocation": {
        "module_args": {
            "admin_state": "up",
```

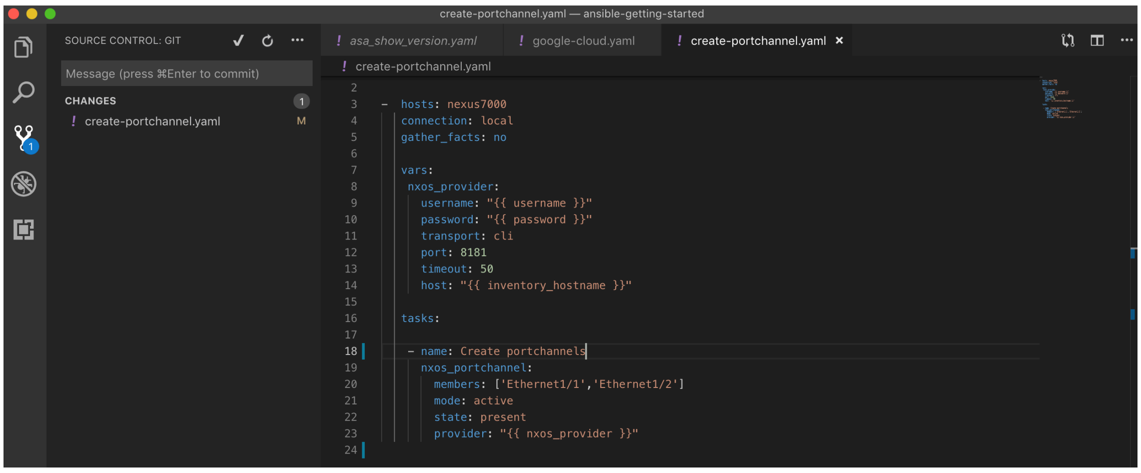
This is the list of VLANs once the playbook is executed:



```
sbx-n9kv-ao# show running-config | grep vlan
  limit-resource vlan minimum 16 maximum 4094
feature interface-vlan
snmp-server enable traps vtp vlancreate
snmp-server enable traps vtp vlandelete
vlan 1,20,30,40,50-55,100-106
vlan 20
vlan 30
vlan 40
vlan 100
vlan 101
vlan 102
vlan 103
vlan 104
vlan 105
```

Configure Portchannels

```
tasks:
- name: Create portchannel 200
   nxos_portchannel:
    group: 200
    members: ['Ethernet1/1','Ethernet1/2']
    mode: 'active'
    host: "{{ inventory_hostname }}"
    state: present
```

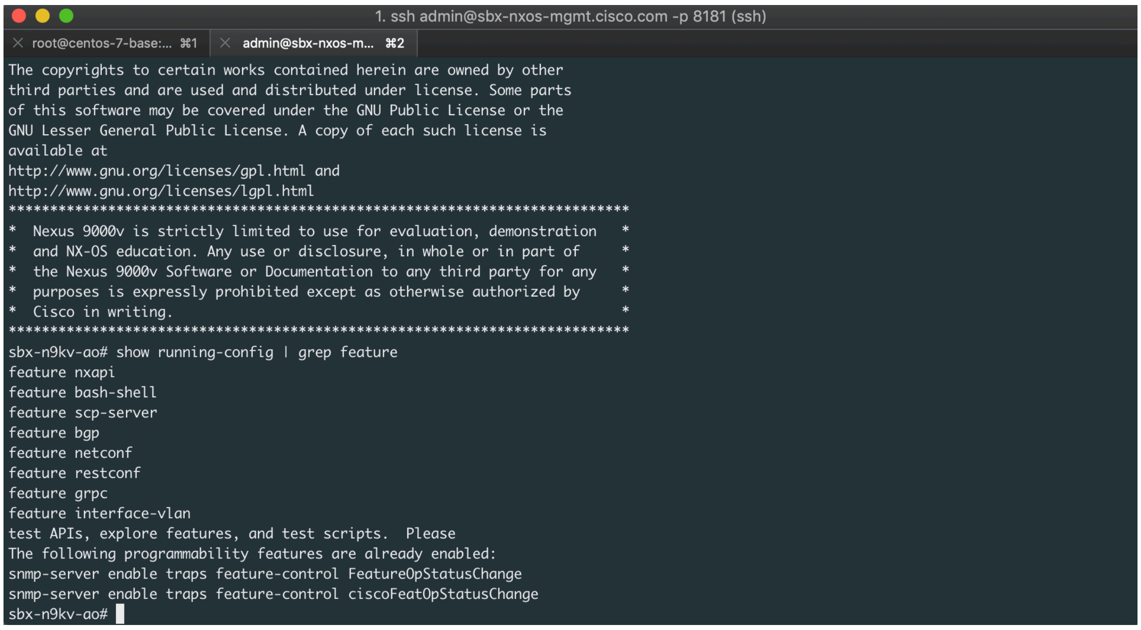

create-portchannel.yaml — ansible-getting-started
SOURCE CONTROL: GIT
Message (press ⌘Enter to commit)
CHANGES
create-portchannel.yaml M
asa_show_version.yaml
google-cloud.yaml
create-portchannel.yaml

```

- hosts: nexus7000
  connection: local
  gather_facts: no

  vars:
   nxos_provider:
     username: "{{ username }}"
     password: "{{ password }}"
     transport: cli
     port: 8181
     timeout: 50
     host: "{{ inventory_hostname }}"

  tasks:

   - name: Create portchannels
     nxos_portchannel:
       members: ['Ethernet1/1','Ethernet1/2']
       mode: active
       state: present
       provider: "{{ nxos_provider }}"

```

Feature Enablement

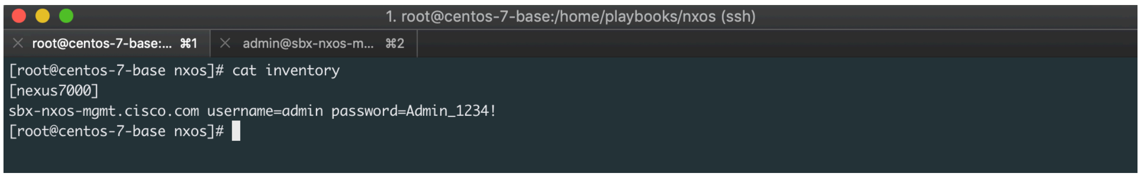
```
tasks:
- name: Ensure DHCP is enabled
  nxos_feature:
   feature: dhcp
   state: enabled
   host: "{{ inventory_hostname }}"
```

This is the list of features enabled:



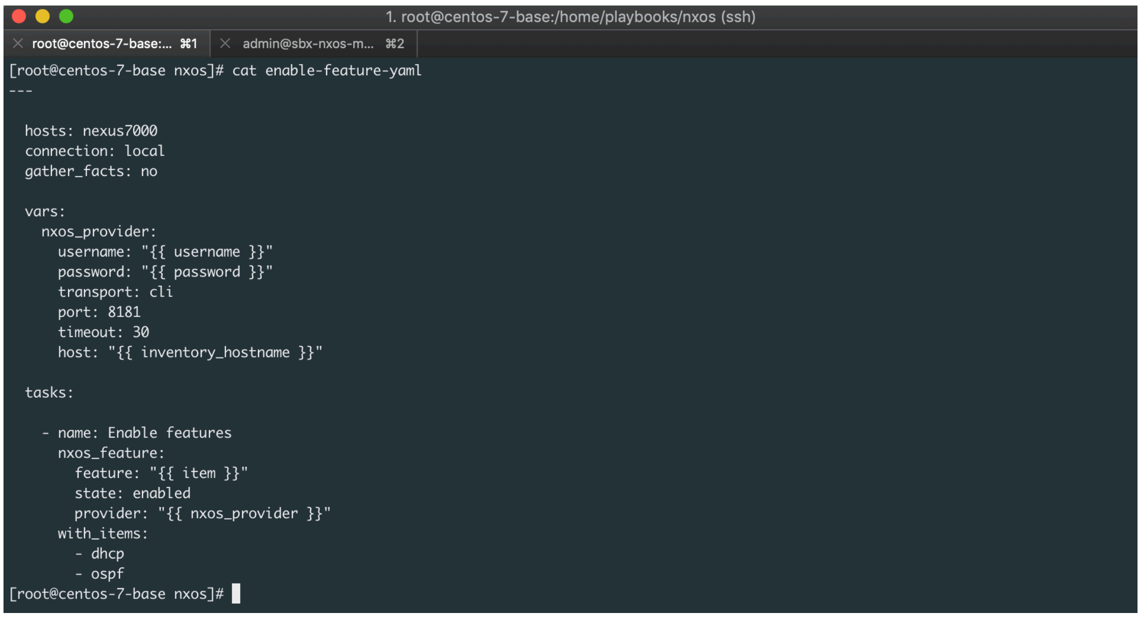
```
The copyrights to certain works contained herein are owned by other
third parties and are used and distributed under license. Some parts
of this software may be covered under the GNU Public License or the
GNU Lesser General Public License. A copy of each such license is
available at
http://www.gnu.org/licenses/gpl.html and
http://www.gnu.org/licenses/lgpl.html
***************************************************************************
*  Nexus 9000v is strictly limited to use for evaluation, demonstration   *
*  and NX-OS education. Any use or disclosure, in whole or in part of     *
*  the Nexus 9000v Software or Documentation to any third party for any   *
*  purposes is expressly prohibited except as otherwise authorized by     *
*  Cisco in writing.                                                      *
***************************************************************************
sbx-n9kv-ao# show running-config | grep feature
feature nxapi
feature bash-shell
feature scp-server
feature bgp
feature netconf
feature restconf
feature grpc
feature interface-vlan
test APIs, explore features, and test scripts.  Please
The following programmability features are already enabled:
snmp-server enable traps feature-control FeatureOpStatusChange
snmp-server enable traps feature-control ciscoFeatOpStatusChange
sbx-n9kv-ao#
```

The inventory file includes the Nexus 9K hostname, username and password:



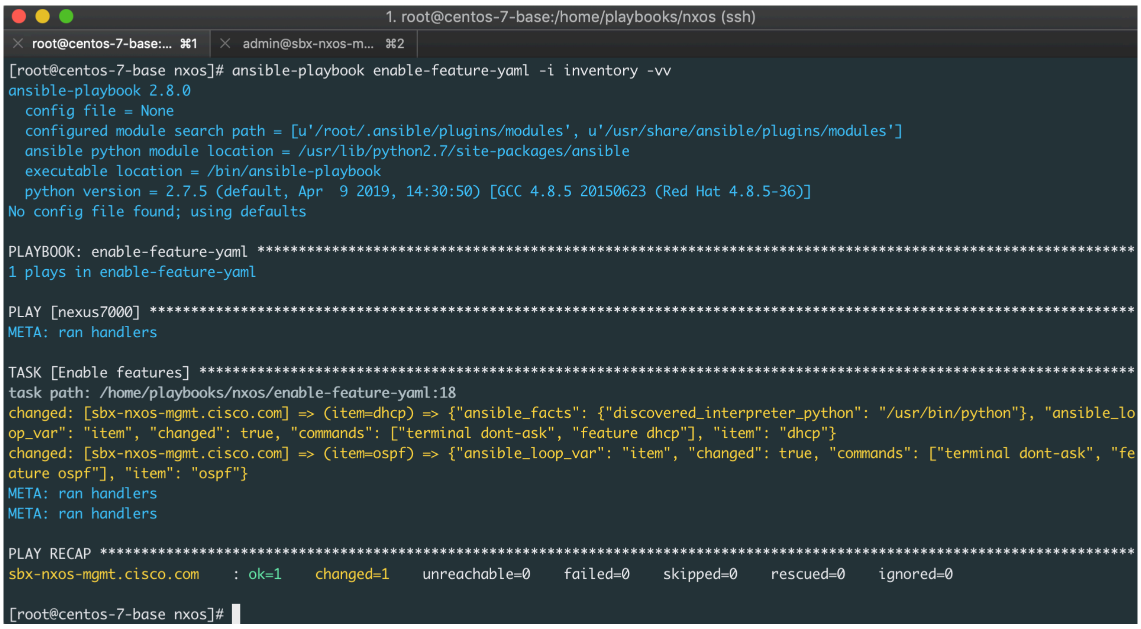
```
[root@centos-7-base nxos]# cat inventory
[nexus7000]
sbx-nxos-mgmt.cisco.com username=admin password=Admin_1234!
[root@centos-7-base nxos]#
```

The playbook enables two new features: DHCP and OSPF:


1. root@centos-7-base:/home/playbooks/nxos (ssh)
root@centos-7-base:... ⌘1
admin@sbx-nxos-m... ⌘2

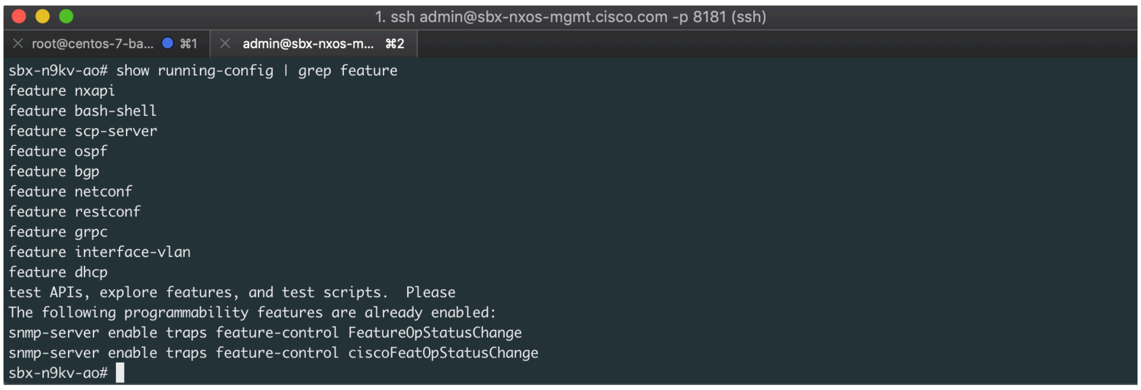
```
[root@centos-7-base nxos]# cat enable-feature-yaml
---

  hosts: nexus7000
  connection: local
  gather_facts: no

  vars:
    nxos_provider:
      username: "{{ username }}"
      password: "{{ password }}"
      transport: cli
      port: 8181
      timeout: 30
      host: "{{ inventory_hostname }}"

  tasks:

    - name: Enable features
      nxos_feature:
        feature: "{{ item }}"
        state: enabled
        provider: "{{ nxos_provider }}"
      with_items:
        - dhcp
        - ospf
[root@centos-7-base nxos]#
```

Running the playbook:


1. root@centos-7-base:/home/playbooks/nxos (ssh)
root@centos-7-base:... ⌘1
admin@sbx-nxos-m... ⌘2

```
[root@centos-7-base nxos]# ansible-playbook enable-feature-yaml -i inventory -vv
ansible-playbook 2.8.0
  config file = None
  configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/site-packages/ansible
  executable location = /bin/ansible-playbook
  python version = 2.7.5 (default, Apr  9 2019, 14:30:50) [GCC 4.8.5 20150623 (Red Hat 4.8.5-36)]
No config file found; using defaults

PLAYBOOK: enable-feature-yaml ****************************************************************************************************************
1 plays in enable-feature-yaml

PLAY [nexus7000] *****************************************************************************************************************************
META: ran handlers

TASK [Enable features] ***********************************************************************************************************************
task path: /home/playbooks/nxos/enable-feature-yaml:18
changed: [sbx-nxos-mgmt.cisco.com] => (item=dhcp) => {"ansible_facts": {"discovered_interpreter_python": "/usr/bin/python"}, "ansible_lo
op_var": "item", "changed": true, "commands": ["terminal dont-ask", "feature dhcp"], "item": "dhcp"}
changed: [sbx-nxos-mgmt.cisco.com] => (item=ospf) => {"ansible_loop_var": "item", "changed": true, "commands": ["terminal dont-ask", "fe
ature ospf"], "item": "ospf"}
META: ran handlers
META: ran handlers

PLAY RECAP ***********************************************************************************************************************************
sbx-nxos-mgmt.cisco.com    : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@centos-7-base nxos]#
```

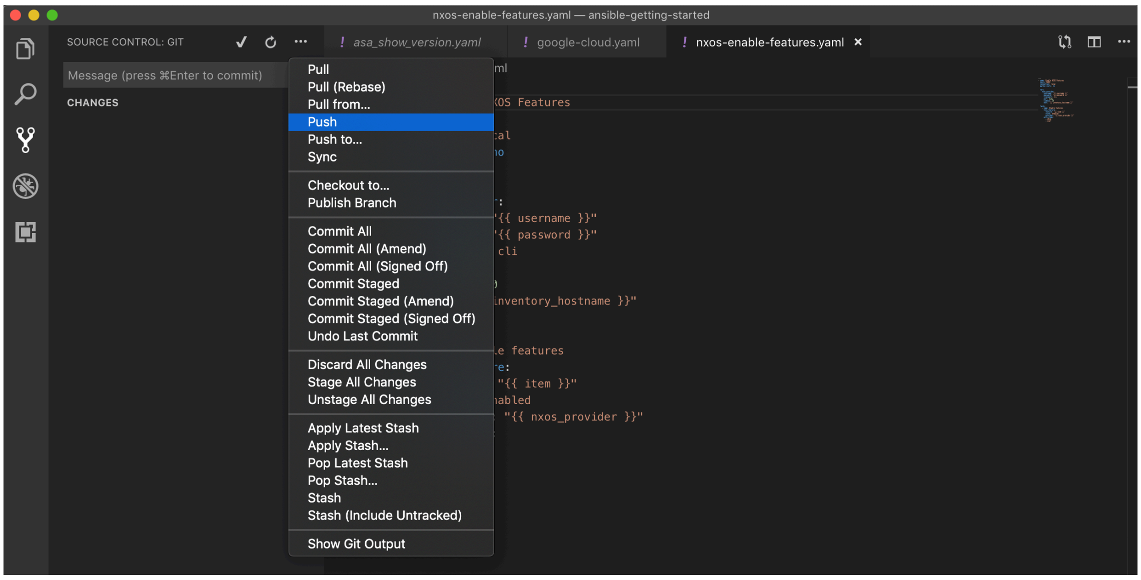
The new features are enabled:

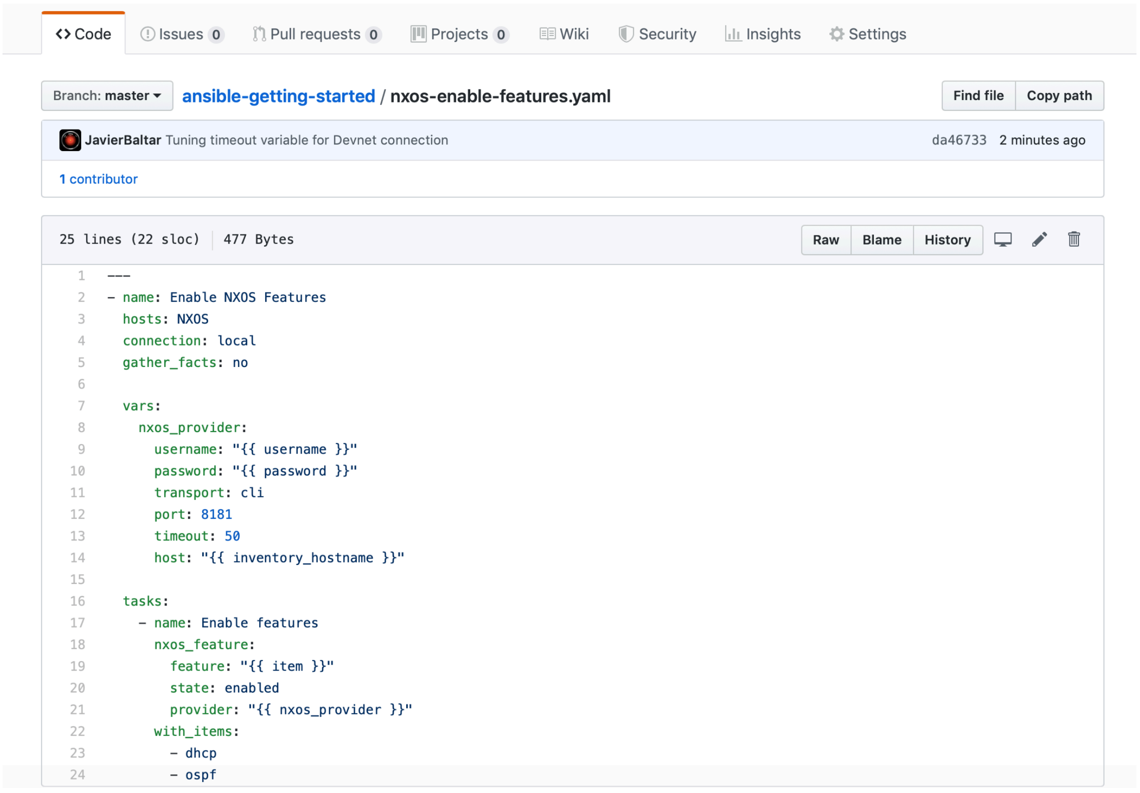


```
sbx-n9kv-ao# show running-config | grep feature
feature nxapi
feature bash-shell
feature scp-server
feature ospf
feature bgp
feature netconf
feature restconf
feature grpc
feature interface-vlan
feature dhcp
test APIs, explore features, and test scripts.  Please
The following programmability features are already enabled:
snmp-server enable traps feature-control FeatureOpStatusChange
snmp-server enable traps feature-control ciscoFeatOpStatusChange
sbx-n9kv-ao#
```

It is time to get into AWX and play around with all these possibilities.

You have to push your code to the Git repository:





ansible-getting-started / nxos-enable-features.yaml

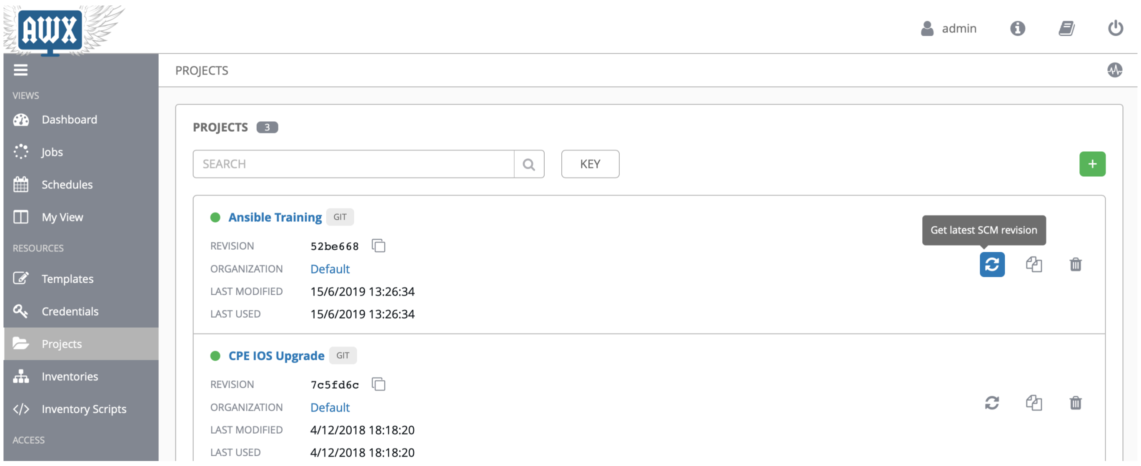
JavierBaltar Tuning timeout variable for Devnet connection da46733 2 minutes ago

1 contributor

25 lines (22 sloc) | 477 Bytes

```yaml
---
- name: Enable NXOS Features
  hosts: NXOS
  connection: local
  gather_facts: no

  vars:
    nxos_provider:
      username: "{{ username }}"
      password: "{{ password }}"
      transport: cli
      port: 8181
      timeout: 50
      host: "{{ inventory_hostname }}"

  tasks:
    - name: Enable features
      nxos_feature:
        feature: "{{ item }}"
        state: enabled
        provider: "{{ nxos_provider }}"
      with_items:
        - dhcp
        - ospf
```

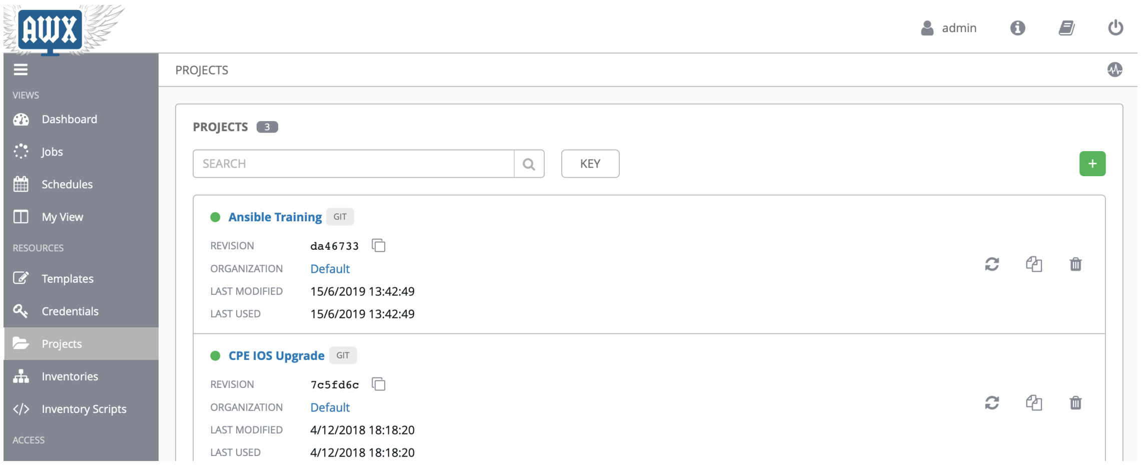
Ensure that AWX project is updated by clicking on Sync:



Ansible Training GIT
REVISION 52be668
ORGANIZATION Default
LAST MODIFIED 15/6/2019 13:26:34
LAST USED 15/6/2019 13:26:34

CPE IOS Upgrade GIT
REVISION 7c5fd6c
ORGANIZATION Default
LAST MODIFIED 4/12/2018 18:18:20
LAST USED 4/12/2018 18:18:20

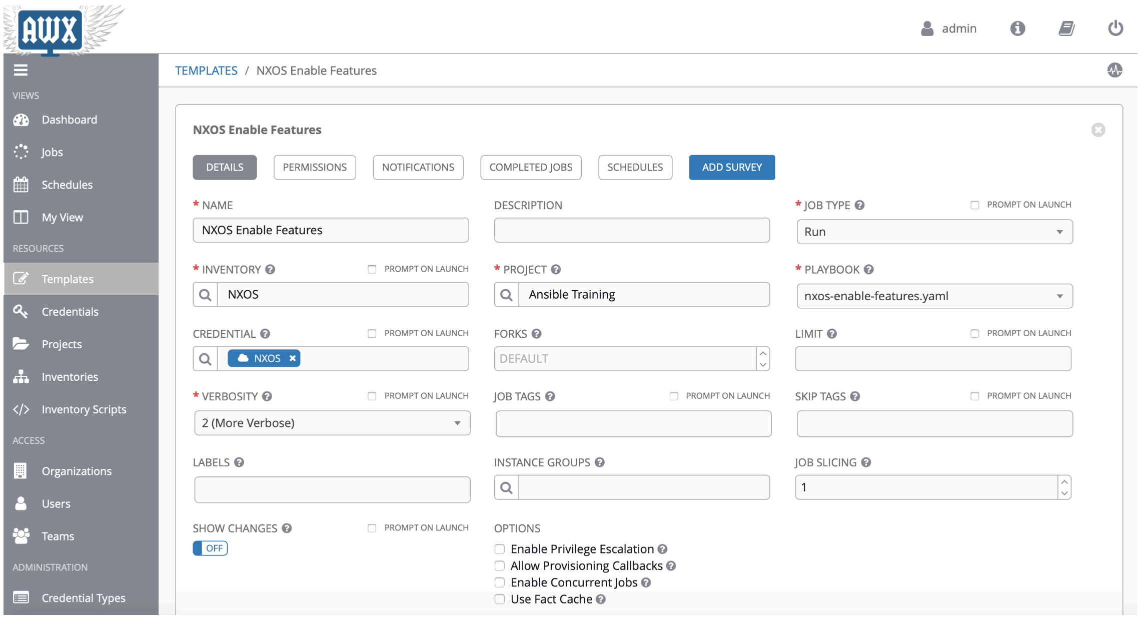
The revision matches with Github:



Ansible Training GIT
REVISION da46733
ORGANIZATION Default
LAST MODIFIED 15/6/2019 13:42:49
LAST USED 15/6/2019 13:42:49

CPE IOS Upgrade GIT
REVISION 7c5fd6c
ORGANIZATION Default
LAST MODIFIED 4/12/2018 18:18:20
LAST USED 4/12/2018 18:18:20

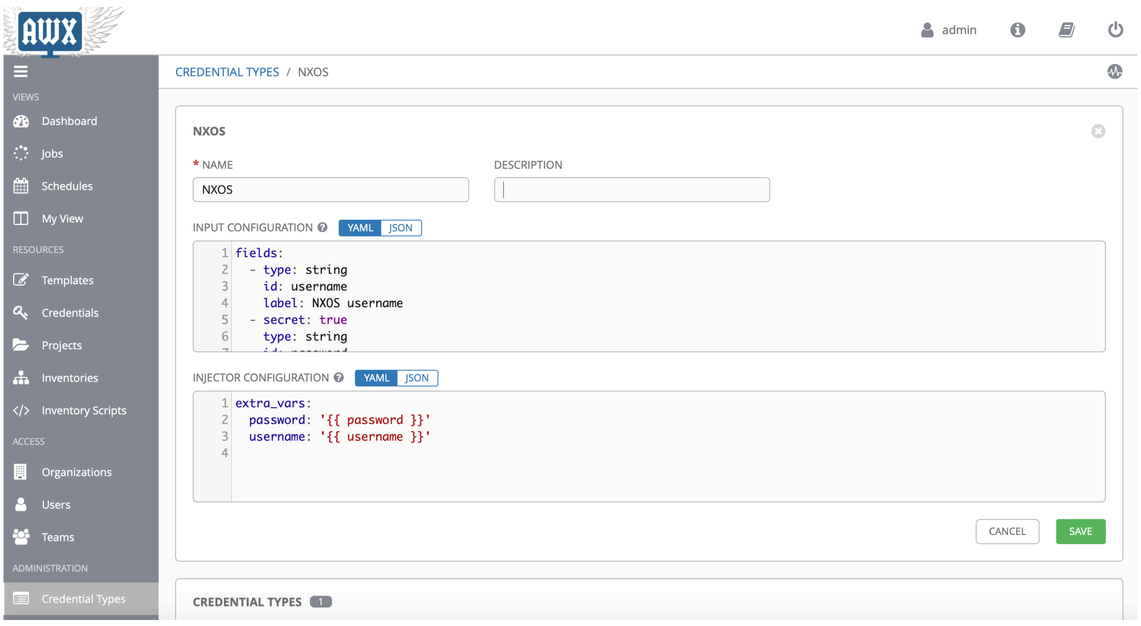
Before creating the template, you have to provision the credentials.



NXOS Enable Features

* NAME: NXOS Enable Features
DESCRIPTION:
* JOB TYPE: Run
* INVENTORY: NXOS
* PROJECT: Ansible Training
* PLAYBOOK: nxos-enable-features.yaml
CREDENTIAL: NXOS
FORKS: DEFAULT
LIMIT:
* VERBOSITY: 2 (More Verbose)
JOB TAGS:
SKIP TAGS:
LABELS:
INSTANCE GROUPS:
JOB SLICING: 1
SHOW CHANGES: OFF
OPTIONS
- Enable Privilege Escalation
- Allow Provisioning Callbacks
- Enable Concurrent Jobs
- Use Fact Cache

Click on Credentials Types



NXOS

* NAME: NXOS

DESCRIPTION:

INPUT CONFIGURATION: YAML

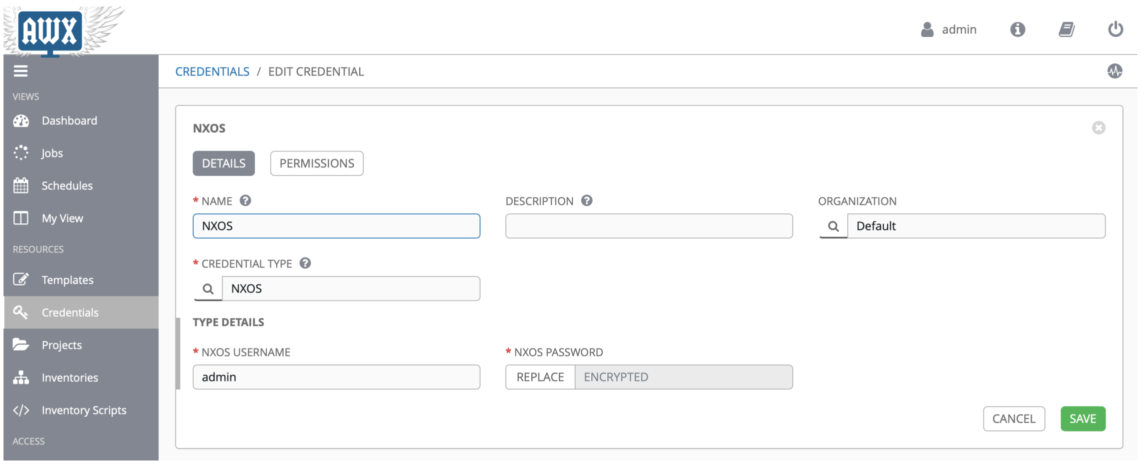
```
fields:
  - type: string
    id: username
    label: NXOS username
  - secret: true
    type: string
```

INJECTOR CONFIGURATION: YAML

```
extra_vars:
  password: '{{ password }}'
  username: '{{ username }}'
```

Input Configuration:

fields:
- type: string
id: username
label: NXOS username
- secret: true
type: string
id: password
label: NXOS password
required:
- username
- password



NXOS

* NAME: NXOS

DESCRIPTION:

ORGANIZATION: Default

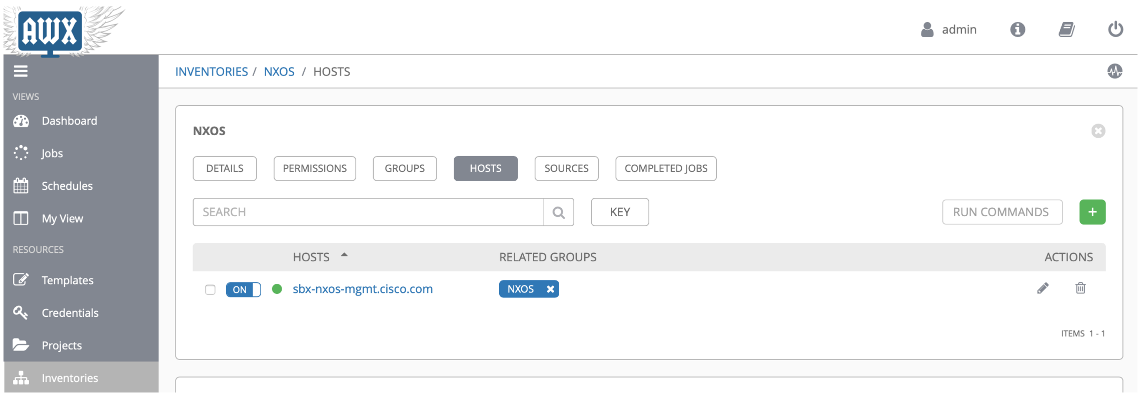
* CREDENTIAL TYPE: NXOS

TYPE DETAILS

* NXOS USERNAME: admin

* NXOS PASSWORD: REPLACE ENCRYPTED

Click on Inventory and add the NXOS host:



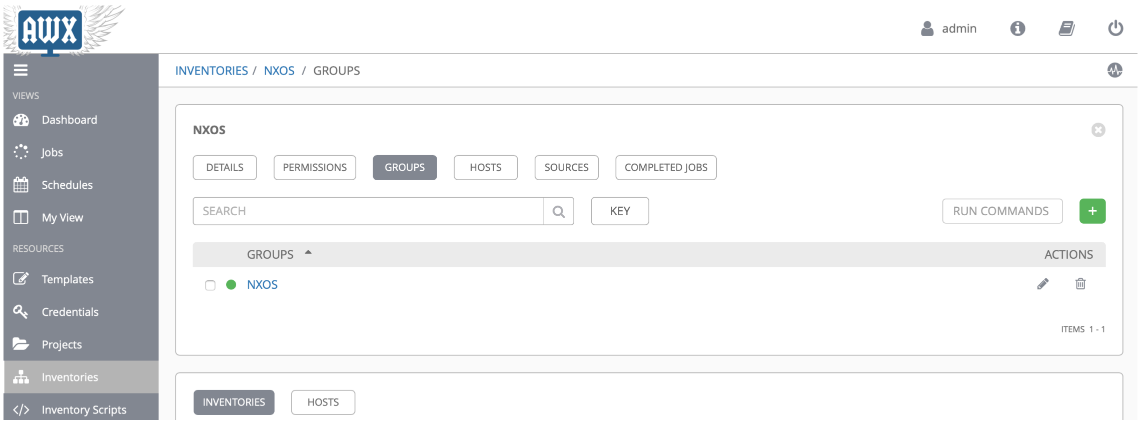
INVENTORIES / NXOS / HOSTS

NXOS

HOSTS	RELATED GROUPS	ACTIONS
ON sbx-nxos-mgmt.cisco.com	NXOS	

ITEMS 1 - 1

Create the corresponding group:



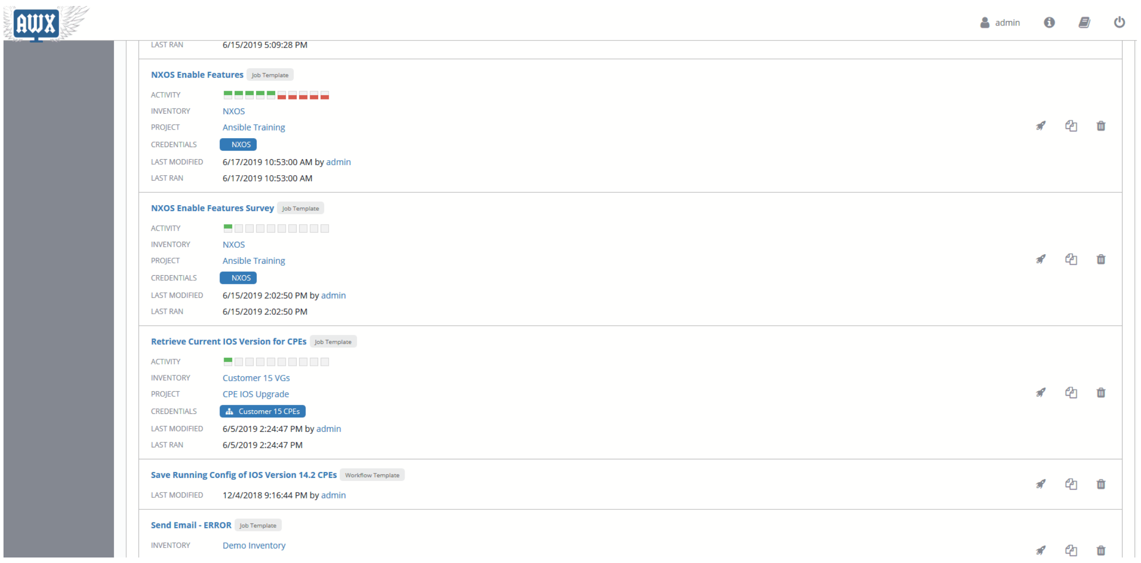
INVENTORIES / NXOS / GROUPS

NXOS

GROUPS	ACTIONS
NXOS	

ITEMS 1 - 1

Let´s run the plyabook:



LAST RAN 6/15/2019 5:09:28 PM

NXOS Enable Features Job Template
ACTIVITY 🟩🟩🟩🟩🟩🟥🟥🟥🟥🟥
INVENTORY NXOS
PROJECT Ansible Training
CREDENTIALS NXOS
LAST MODIFIED 6/17/2019 10:53:00 AM by admin
LAST RAN 6/17/2019 10:53:00 AM

NXOS Enable Features Survey Job Template
ACTIVITY 🟩
INVENTORY NXOS
PROJECT Ansible Training
CREDENTIALS NXOS
LAST MODIFIED 6/15/2019 2:02:50 PM by admin
LAST RAN 6/15/2019 2:02:50 PM

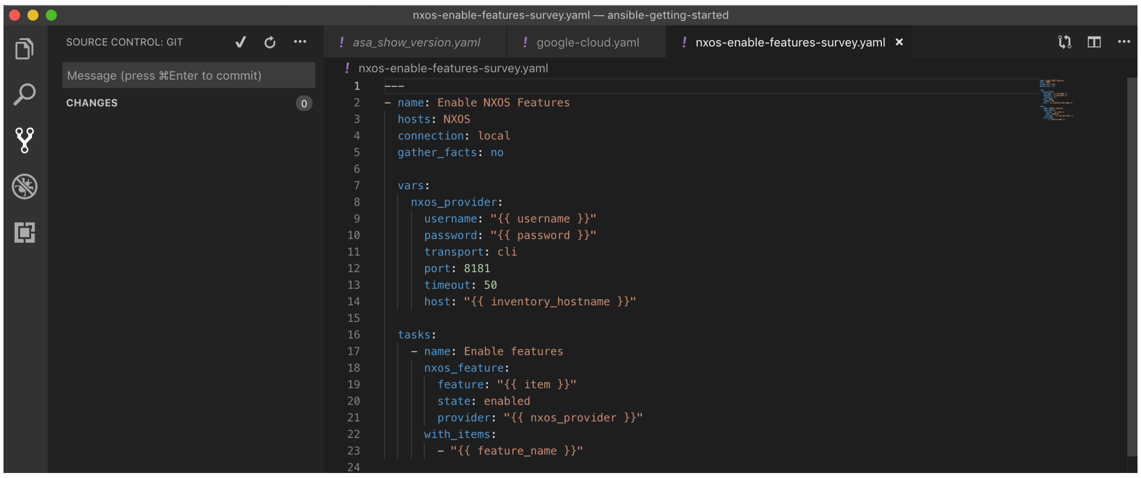
Retrieve Current IOS Version for CPEs Job Template
ACTIVITY 🟩
INVENTORY Customer 15 VGs
PROJECT CPE IOS Upgrade
CREDENTIALS Customer 15 CPEs
LAST MODIFIED 6/5/2019 2:24:47 PM by admin
LAST RAN 6/5/2019 2:24:47 PM

Save Running Config of IOS Version 14.2 CPEs Workflow Template
LAST MODIFIED 12/4/2018 9:16:44 PM by admin

Send Email - ERROR Job Template
INVENTORY Demo Inventory

You can also add a survey, which passes a variable value to the playbook execution.

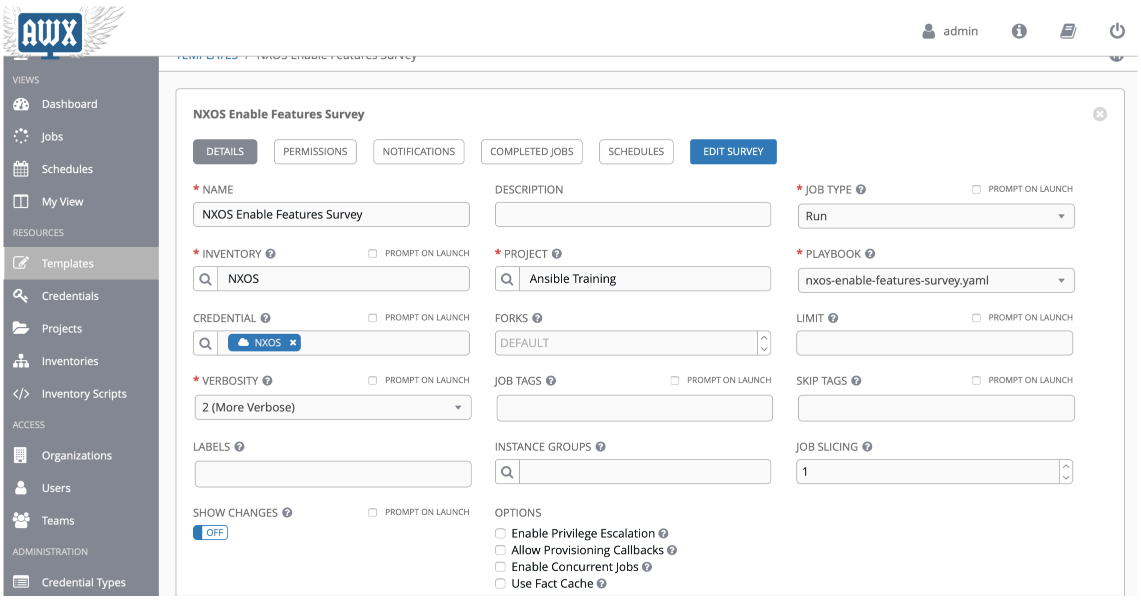
I am adding a new variable called "feature_name".



nxos-enable-features-survey.yaml — ansible-getting-started

```
---
- name: Enable NXOS Features
  hosts: NXOS
  connection: local
  gather_facts: no

  vars:
    nxos_provider:
      username: "{{ username }}"
      password: "{{ password }}"
      transport: cli
      port: 8181
      timeout: 50
      host: "{{ inventory_hostname }}"

  tasks:
    - name: Enable features
      nxos_feature:
        feature: "{{ item }}"
        state: enabled
        provider: "{{ nxos_provider }}"
      with_items:
        - "{{ feature_name }}"
```

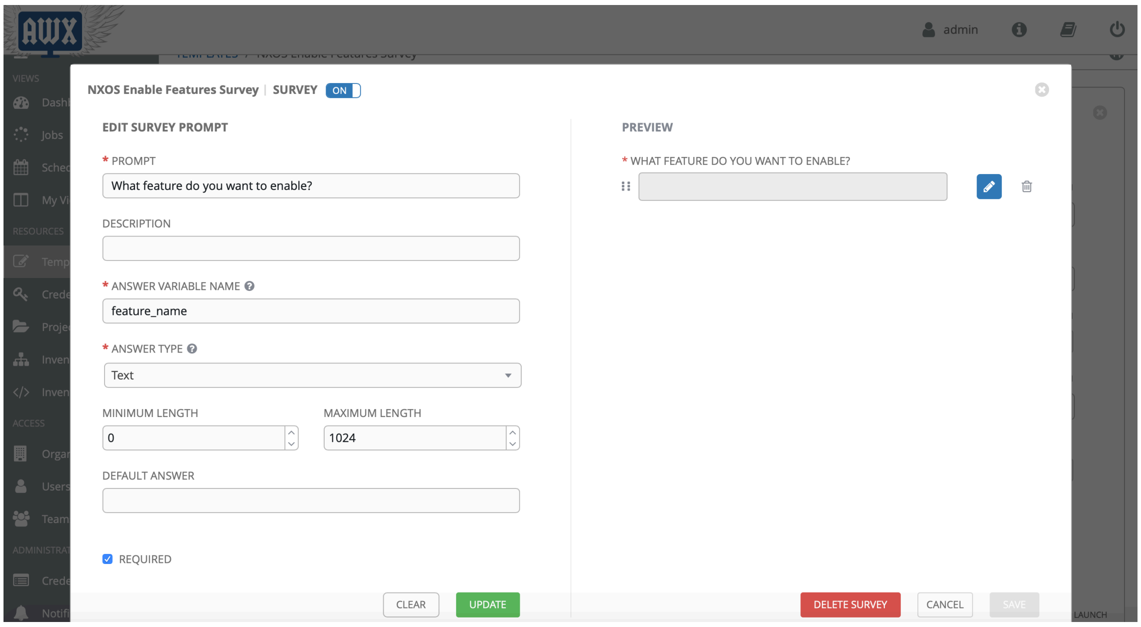
Click on your template and Edit Survey:



NXOS Enable Features Survey

* NAME: NXOS Enable Features Survey	DESCRIPTION:	* JOB TYPE: Run
* INVENTORY: NXOS	* PROJECT: Ansible Training	* PLAYBOOK: nxos-enable-features-survey.yaml
CREDENTIAL: NXOS	FORKS: DEFAULT	LIMIT:
* VERBOSITY: 2 (More Verbose)	JOB TAGS:	SKIP TAGS:
LABELS:	INSTANCE GROUPS:	JOB SLICING: 1
SHOW CHANGES: OFF	OPTIONS: Enable Privilege Escalation; Allow Provisioning Callbacks; Enable Concurrent Jobs; Use Fact Cache	

Add a prompt, variable name and answer type:



* PROMPT: What feature do you want to enable?

DESCRIPTION:

* ANSWER VARIABLE NAME: feature_name

* ANSWER TYPE: Text

MINIMUM LENGTH: 0

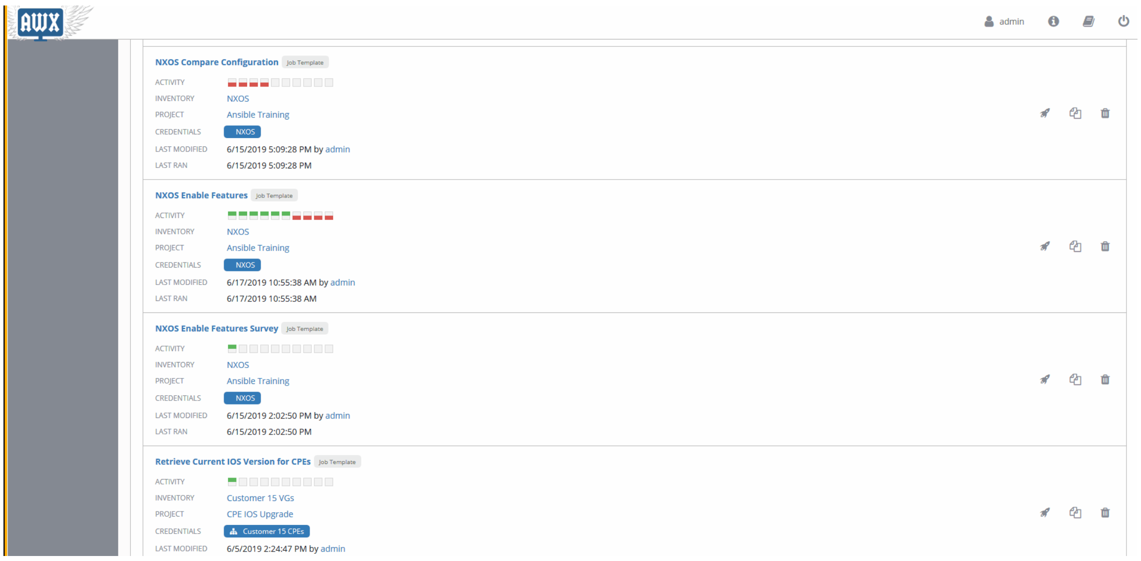
MAXIMUM LENGTH: 1024

DEFAULT ANSWER:

REQUIRED ✓

* WHAT FEATURE DO YOU WANT TO ENABLE?

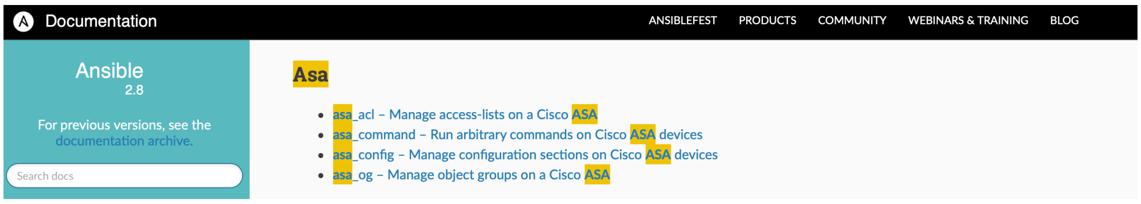
In this case, you have to specify the feature before running the playbook as shown below:



NXOS Compare Configuration	Job Template
ACTIVITY	
INVENTORY	NXOS
PROJECT	Ansible Training
CREDENTIALS	NXOS
LAST MODIFIED	6/15/2019 5:09:28 PM by admin
LAST RAN	6/15/2019 5:09:28 PM
NXOS Enable Features	Job Template
ACTIVITY	
INVENTORY	NXOS
PROJECT	Ansible Training
CREDENTIALS	NXOS
LAST MODIFIED	6/17/2019 10:55:38 AM by admin
LAST RAN	6/17/2019 10:55:38 AM
NXOS Enable Features Survey	Job Template
ACTIVITY	
INVENTORY	NXOS
PROJECT	Ansible Training
CREDENTIALS	NXOS
LAST MODIFIED	6/15/2019 2:02:50 PM by admin
LAST RAN	6/15/2019 2:02:50 PM
Retrieve Current IOS Version for CPEs	Job Template
ACTIVITY	
INVENTORY	Customer 15 VGs
PROJECT	CPE IOS Upgrade
CREDENTIALS	Customer 15 CPEs
LAST MODIFIED	6/5/2019 2:24:47 PM by admin

Cisco ASA Management

Please refer to: https://docs.ansible.com/ansible/latest/modules/list_of_network_modules.html?highlight=asa



Asa

- asa_acl – Manage access-lists on a Cisco ASA
- asa_command – Run arbitrary commands on Cisco ASA devices
- asa_config – Manage configuration sections on Cisco ASA devices
- asa_og – Manage object groups on a Cisco ASA

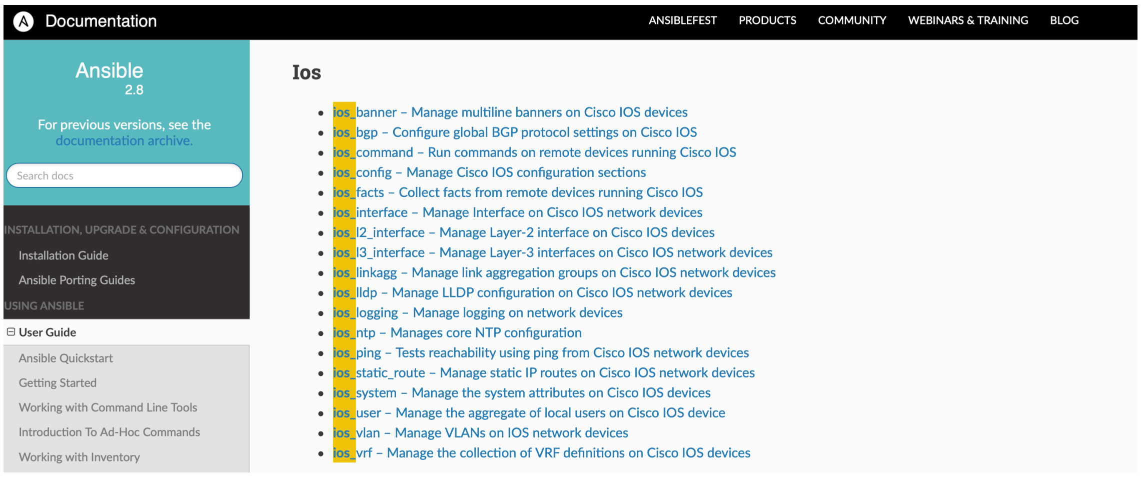
Playbooks

Create Network object

```
---
- hosts: asa
  gather_facts: no
  connection: network_cli

vars:
  object_hosts:
   - 10.45.16.79
   - 10.45.16.72

  object_group: TEST

  tasks:
   - name: "CREATE HOST AND GROUPS ON {{ ansible_ssh_host }}"
      asa_config:
       lines:
        - network-object host {{ item }}
        parents: object-group network {{ object_group }}
       with_items: "{{ object_hosts }}"
```

Cisco IOS Management

Please refer to: https://docs.ansible.com/ansible/latest/modules/list_of_network_modules.html?highlight=ios_



Ios

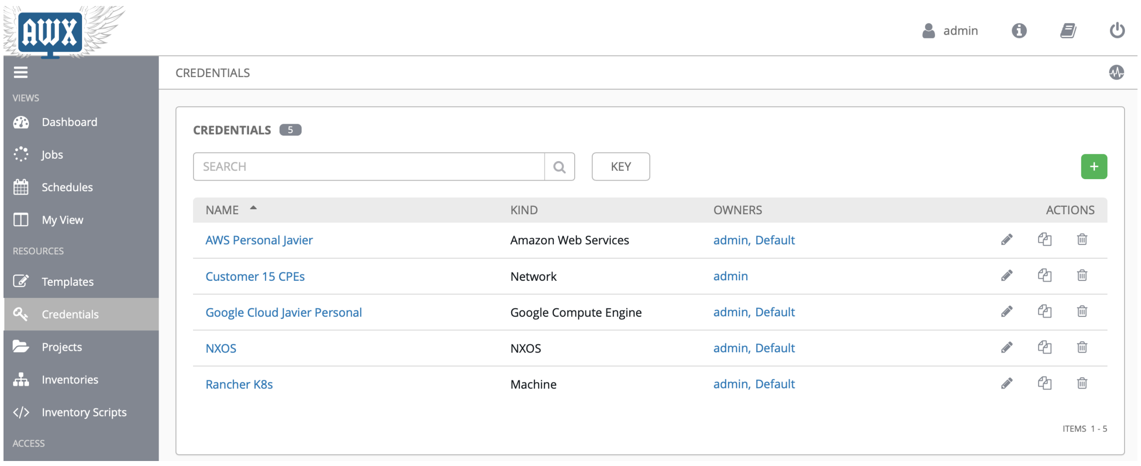
- ios_banner – Manage multiline banners on Cisco IOS devices
- ios_bgp – Configure global BGP protocol settings on Cisco IOS
- ios_command – Run commands on remote devices running Cisco IOS
- ios_config – Manage Cisco IOS configuration sections
- ios_facts – Collect facts from remote devices running Cisco IOS
- ios_interface – Manage Interface on Cisco IOS network devices
- ios_l2_interface – Manage Layer-2 interface on Cisco IOS devices
- ios_l3_interface – Manage Layer-3 interfaces on Cisco IOS network devices
- ios_linkagg – Manage link aggregation groups on Cisco IOS network devices
- ios_lldp – Manage LLDP configuration on Cisco IOS network devices
- ios_logging – Manage logging on network devices
- ios_ntp – Manages core NTP configuration
- ios_ping – Tests reachability using ping from Cisco IOS network devices
- ios_static_route – Manage static IP routes on Cisco IOS network devices
- ios_system – Manage the system attributes on Cisco IOS devices
- ios_user – Manage the aggregate of local users on Cisco IOS device
- ios_vlan – Manage VLANs on IOS network devices
- ios_vrf – Manage the collection of VRF definitions on Cisco IOS devices

Deploying to Amazon Web Services

Inventory

You can configure a dynamic inventory syncing with your AWS resources.

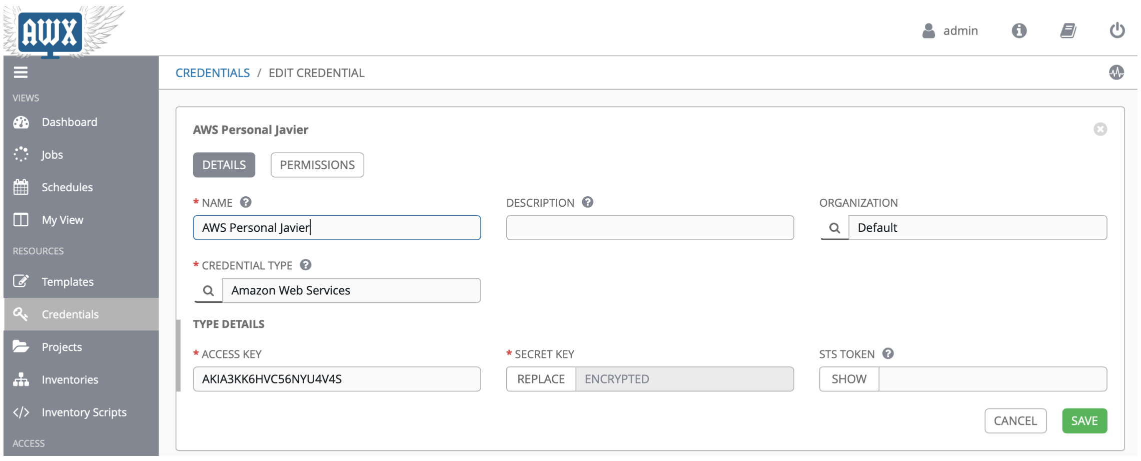
First, configure your AWS account credentials:



NAME	KIND	OWNERS	ACTIONS
AWS Personal Javier	Amazon Web Services	admin, Default	
Customer 15 CPEs	Network	admin	
Google Cloud Javier Personal	Google Compute Engine	admin, Default	
NXOS	NXOS	admin, Default	
Rancher K8s	Machine	admin, Default	

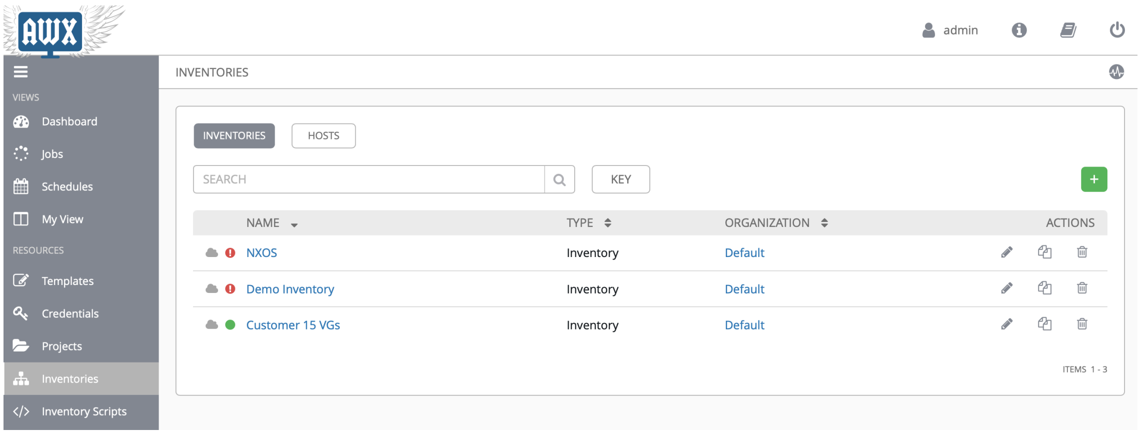
ITEMS 1 - 5

Add your access and secrets keys.



* NAME: AWS Personal Javier	DESCRIPTION:	ORGANIZATION: Default
* CREDENTIAL TYPE: Amazon Web Services		
* ACCESS KEY: AKIA3KK6HVC56NYU4V4S	* SECRET KEY: REPLACE ENCRYPTED	STS TOKEN: SHOW

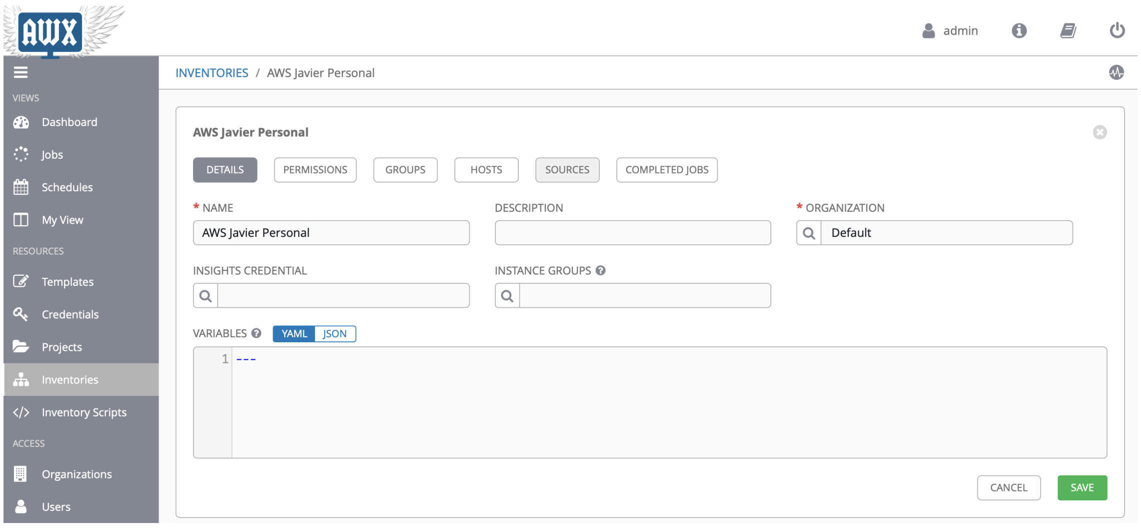
Click on Inventories:



NAME	TYPE	ORGANIZATION	ACTIONS
NXOS	Inventory	Default	
Demo Inventory	Inventory	Default	
Customer 15 VGs	Inventory	Default	

ITEMS 1 - 3

Click on Sources.



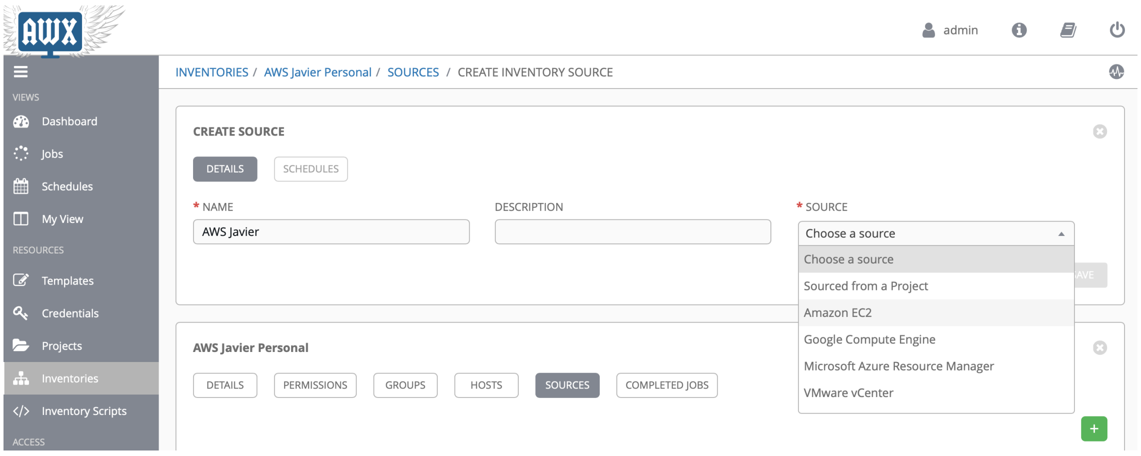
AWS Javier Personal

* NAME	DESCRIPTION	* ORGANIZATION
AWS Javier Personal		Default
INSIGHTS CREDENTIAL	INSTANCE GROUPS	

```
VARIABLES
1 ---
```

Choose the Amazon EC2 source.

NOTE: VMware vCenter is also supported.



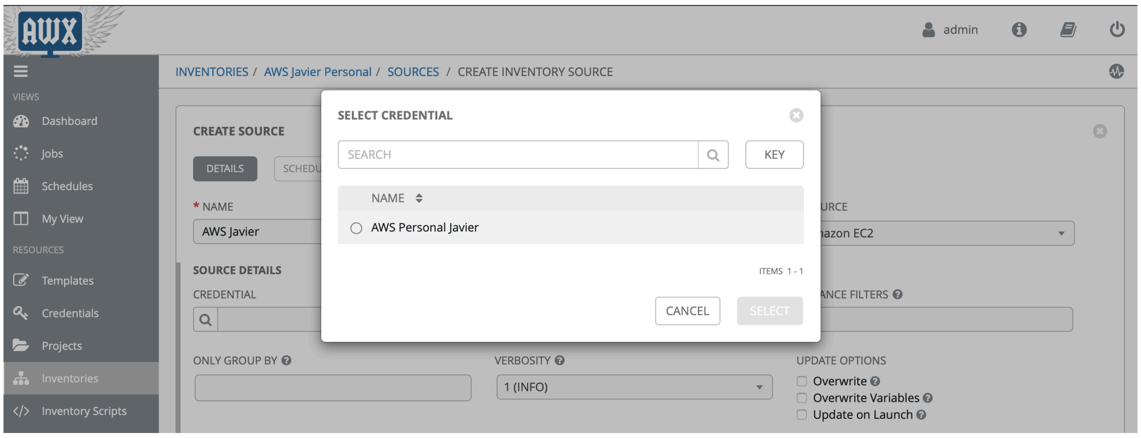
CREATE SOURCE

* NAME	DESCRIPTION	* SOURCE
AWS Javier		Choose a source

Choose a source
Sourced from a Project
Amazon EC2
Google Compute Engine
Microsoft Azure Resource Manager
VMware vCenter

AWS Javier Personal

Choose your AWS credentials:



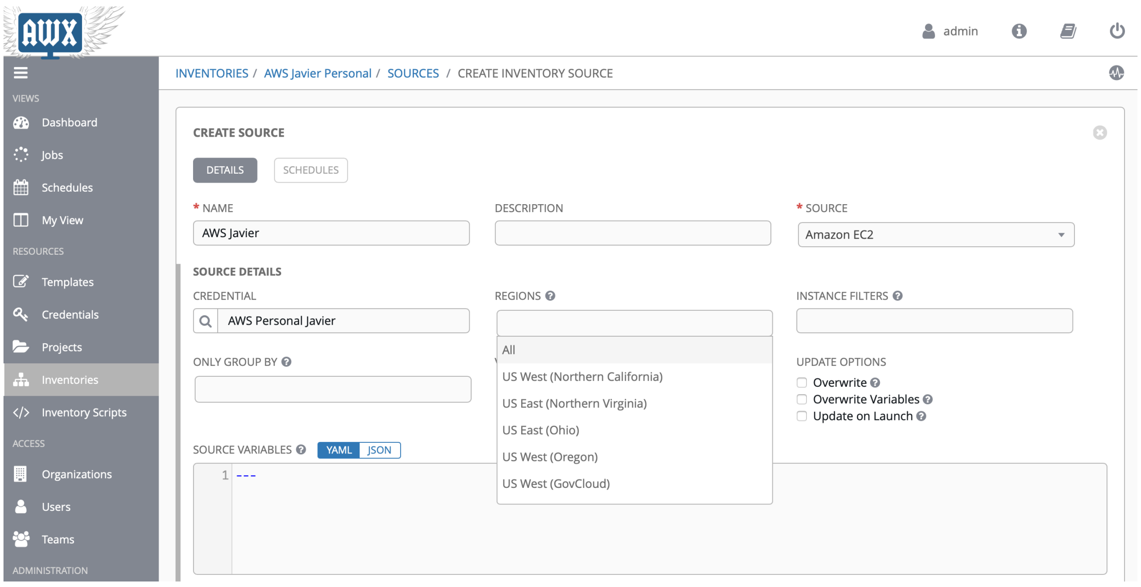
SELECT CREDENTIAL

NAME
○ AWS Personal Javier

ITEMS 1 - 1

* NAME: AWS Javier
SOURCE: Amazon EC2
SOURCE DETAILS
CREDENTIAL
ONLY GROUP BY
VERBOSITY: 1 (INFO)
UPDATE OPTIONS
☐ Overwrite
☐ Overwrite Variables
☐ Update on Launch

Select your AWS regions:



INVENTORIES / AWS Javier Personal / SOURCES / CREATE INVENTORY SOURCE

CREATE SOURCE

* NAME: AWS Javier
DESCRIPTION:
* SOURCE: Amazon EC2

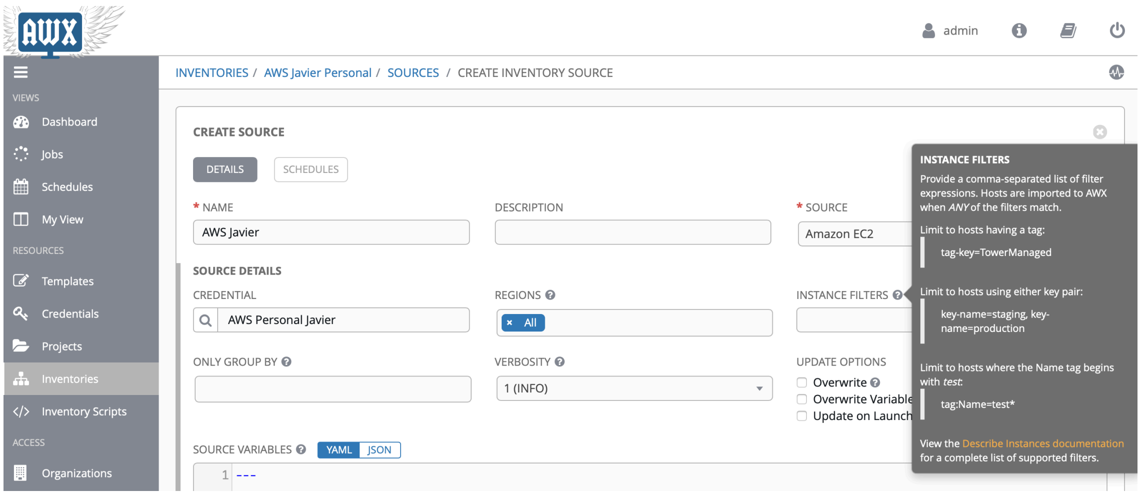
SOURCE DETAILS
CREDENTIAL: AWS Personal Javier
REGIONS:
- All
- US West (Northern California)
- US East (Northern Virginia)
- US East (Ohio)
- US West (Oregon)
- US West (GovCloud)

INSTANCE FILTERS:
ONLY GROUP BY:
UPDATE OPTIONS
- Overwrite
- Overwrite Variables
- Update on Launch

SOURCE VARIABLES

```
---
```

You can also apply filters:



INVENTORIES / AWS Javier Personal / SOURCES / CREATE INVENTORY SOURCE

CREATE SOURCE

* NAME: AWS Javier
DESCRIPTION:
* SOURCE: Amazon EC2

SOURCE DETAILS
CREDENTIAL: AWS Personal Javier
REGIONS: All
INSTANCE FILTERS:
ONLY GROUP BY:
VERBOSITY: 1 (INFO)
UPDATE OPTIONS
- Overwrite
- Overwrite Variables
- Update on Launch

SOURCE VARIABLES

```
---
```

INSTANCE FILTERS

Provide a comma-separated list of filter expressions. Hosts are imported to AWX when *ANY* of the filters match.

Limit to hosts having a tag:

tag-key=TowerManaged

Limit to hosts using either key pair:

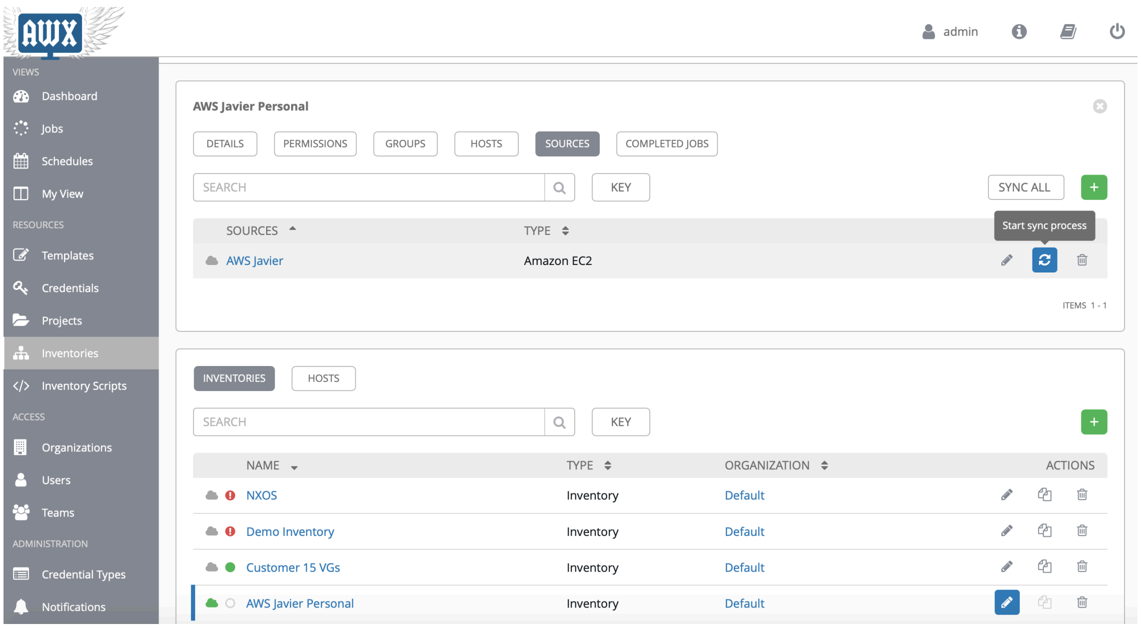
key-name=staging, key-name=production

Limit to hosts where the Name tag begins with *test*:

tag:Name=test*

View the Describe Instances documentation for a complete list of supported filters.

Start the sync process:



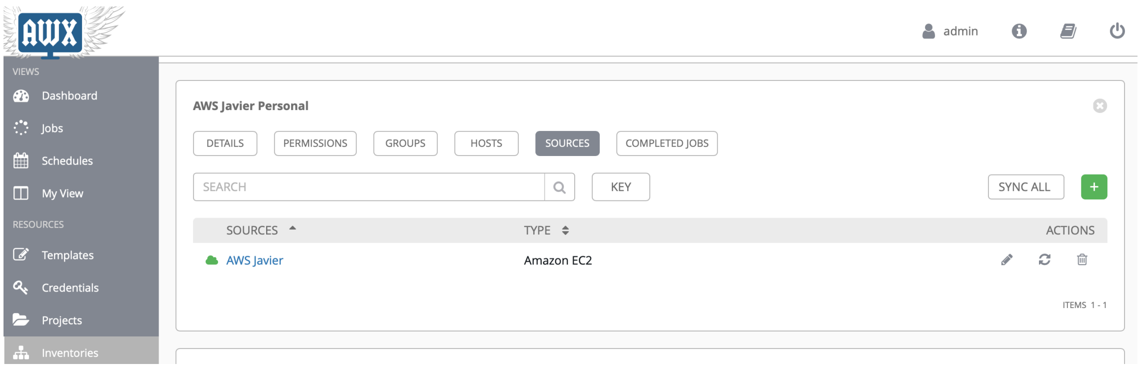
AWS Javier Personal

SOURCES	TYPE
AWS Javier	Amazon EC2

ITEMS 1 - 1

NAME	TYPE	ORGANIZATION	ACTIONS
NXOS	Inventory	Default	
Demo Inventory	Inventory	Default	
Customer 15 VGs	Inventory	Default	
AWS Javier Personal	Inventory	Default	

Your inventory is synced:

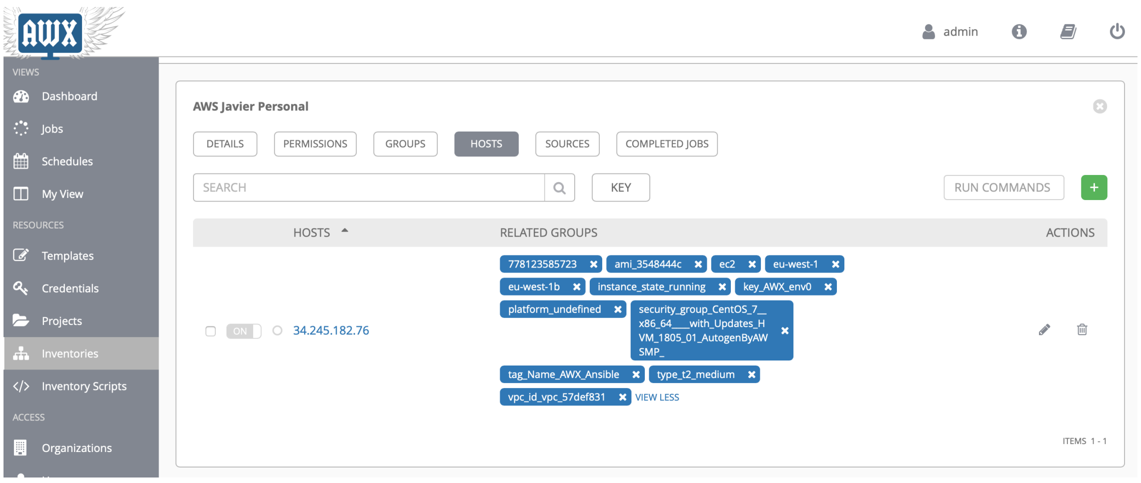


AWS Javier Personal

SOURCES	TYPE	ACTIONS
AWS Javier	Amazon EC2	

ITEMS 1 - 1

Click on hosts and check that the hosts are imported:

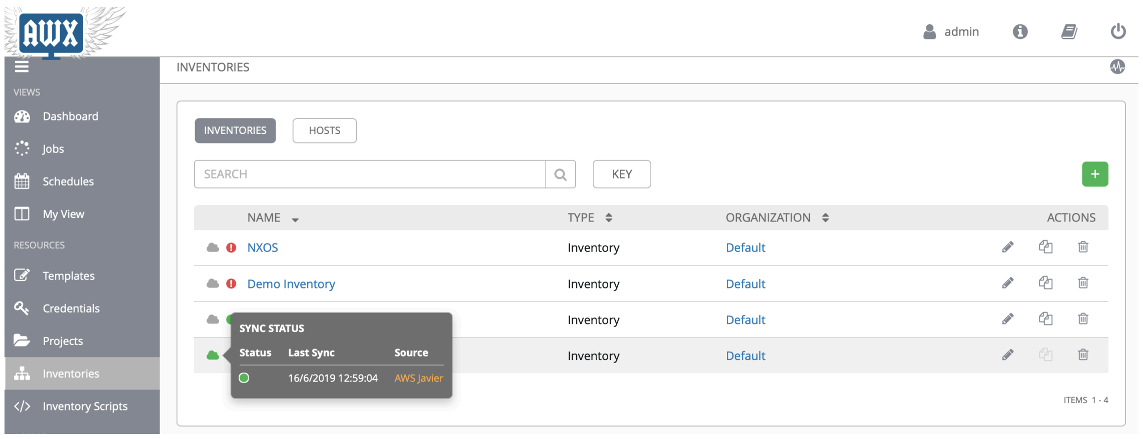


AWS Javier Personal

HOSTS	RELATED GROUPS	ACTIONS
34.245.182.76	778123585723, ami_3548444c, ec2, eu-west-1, eu-west-1b, instance_state_running, key_AWX_env0, platform_undefined, security_group_CentOS_7__x86_64___with_Updates_HVM_1805_01_AutogenByAWSMP_, tag_Name_AWX_Ansible, type_t2_medium, vpc_id_vpc_57def831	

ITEMS 1 - 1

You can check the latest sync details.

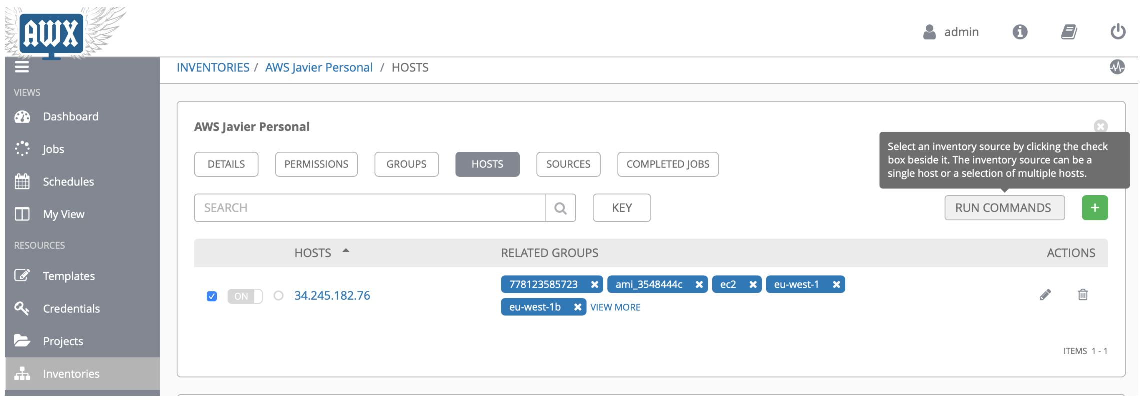


NAME	TYPE	ORGANIZATION	ACTIONS
NXOS	Inventory	Default	
Demo Inventory	Inventory	Default	
	Inventory	Default	
	Inventory	Default	

SYNC STATUS

Status	Last Sync	Source
○	16/6/2019 12:59:04	AWS Javier

ITEMS 1 - 4

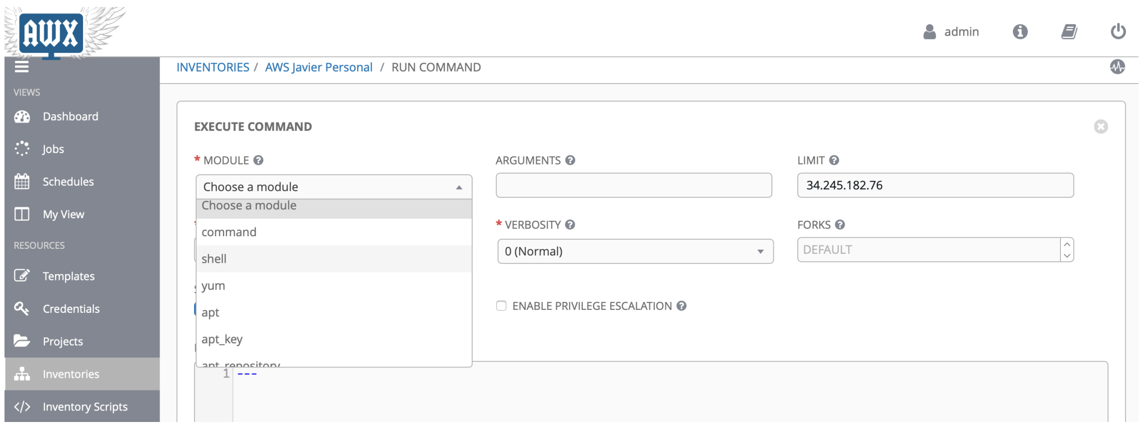


Select an inventory source by clicking the check box beside it. The inventory source can be a single host or a selection of multiple hosts.

HOSTS	RELATED GROUPS	ACTIONS
✓ ON 34.245.182.76	778123585723, ami_3548444c, ec2, eu-west-1, eu-west-1b, VIEW MORE	

ITEMS 1 - 1

In addition, you can run ad-hoc commands as shown below:



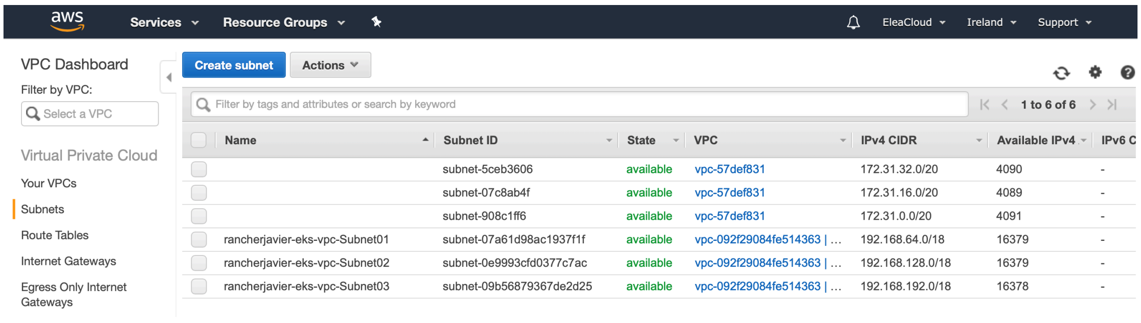
* MODULE: Choose a module	ARGUMENTS:	LIMIT: 34.245.182.76
	* VERBOSITY: 0 (Normal)	FORKS: DEFAULT
	☐ ENABLE PRIVILEGE ESCALATION	

```
---
```

Playbooks

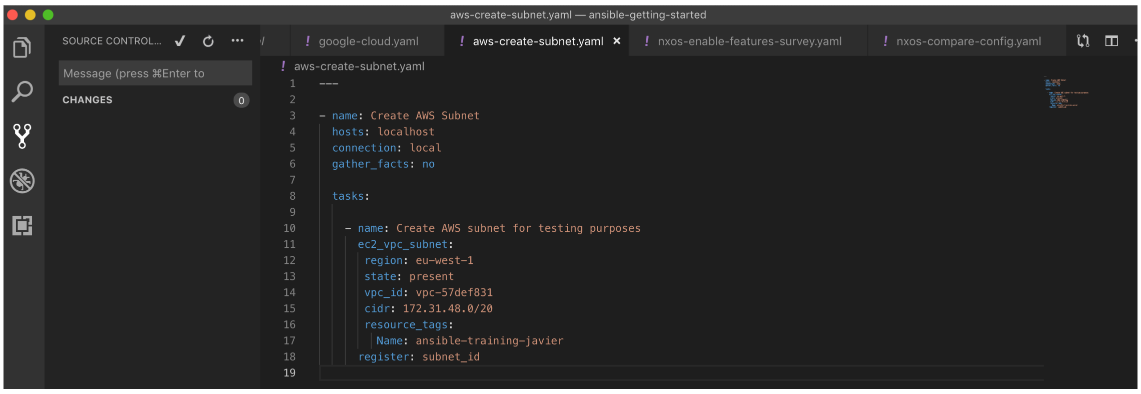
In the following example, an AWS subnet is created.

This is the list of subnets before running my playbook:



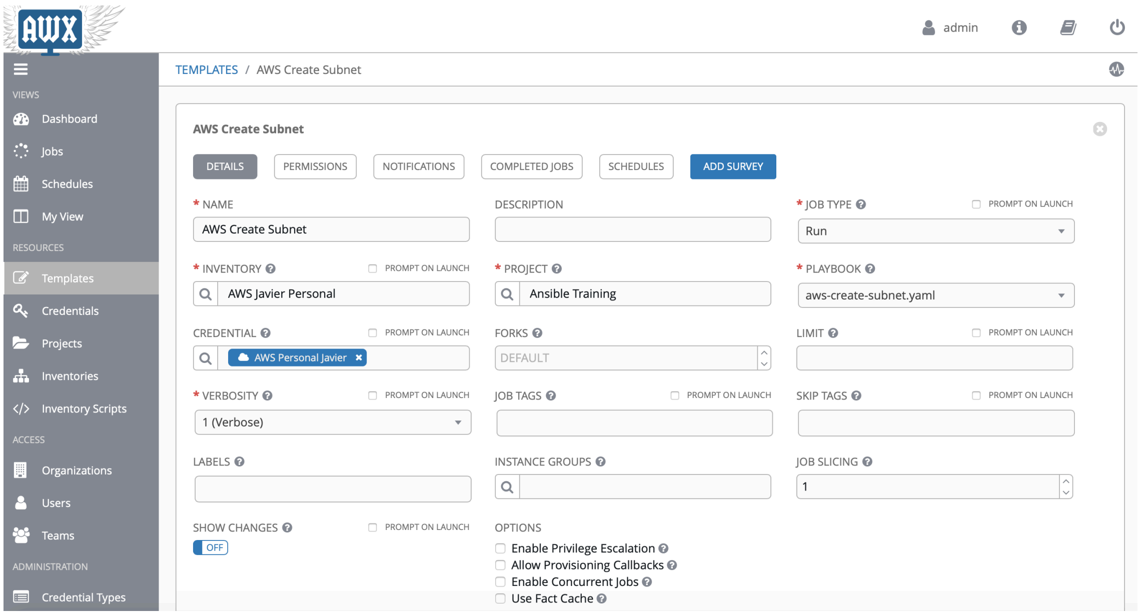
Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	IPv6 C
	subnet-5ceb3606	available	vpc-57def831	172.31.32.0/20	4090	-
	subnet-07c8ab4f	available	vpc-57def831	172.31.16.0/20	4089	-
	subnet-908c1ff6	available	vpc-57def831	172.31.0.0/20	4091	-
rancherjavier-eks-vpc-Subnet01	subnet-07a61d98ac1937f1f	available	vpc-092f29084fe514363 | ...	192.168.64.0/18	16379	-
rancherjavier-eks-vpc-Subnet02	subnet-0e9993cfd0377c7ac	available	vpc-092f29084fe514363 | ...	192.168.128.0/18	16379	-
rancherjavier-eks-vpc-Subnet03	subnet-09b56879367de2d25	available	vpc-092f29084fe514363 | ...	192.168.192.0/18	16378	-

For testing purposes, I am just including the values hardcoded.



```
---

- name: Create AWS Subnet
  hosts: localhost
  connection: local
  gather_facts: no

  tasks:

    - name: Create AWS subnet for testing purposes
      ec2_vpc_subnet:
       region: eu-west-1
       state: present
       vpc_id: vpc-57def831
       cidr: 172.31.48.0/20
       resource_tags:
         Name: ansible-training-javier
      register: subnet_id

```

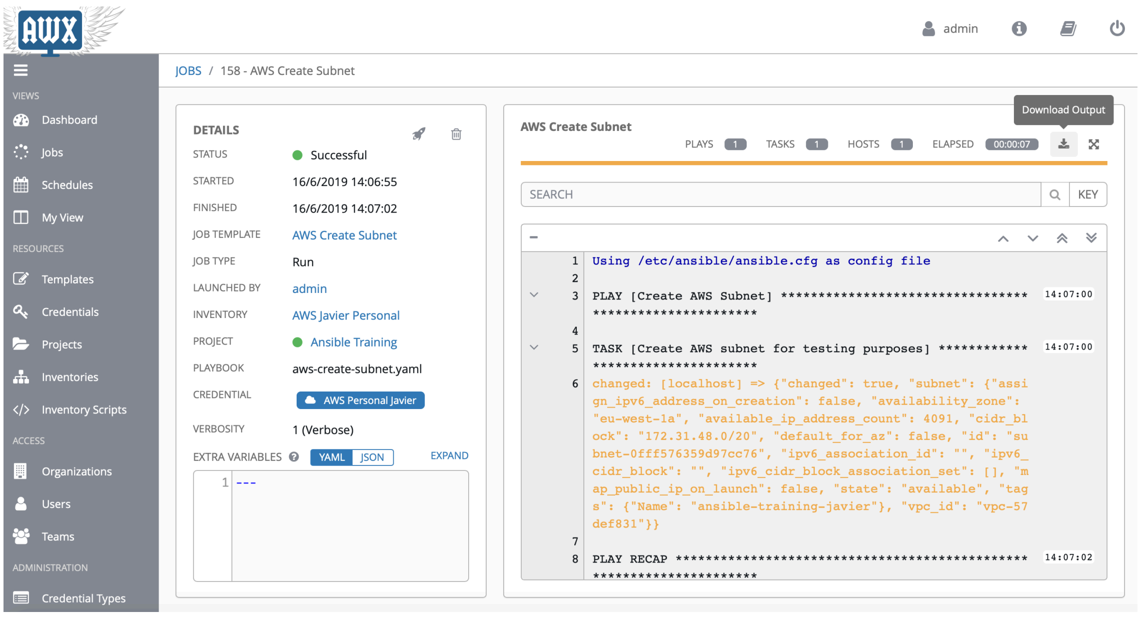
Lets create the template in AWX:



* NAME: AWS Create Subnet	DESCRIPTION:	* JOB TYPE (PROMPT ON LAUNCH): Run
* INVENTORY (PROMPT ON LAUNCH): AWS Javier Personal	* PROJECT: Ansible Training	* PLAYBOOK: aws-create-subnet.yaml
CREDENTIAL (PROMPT ON LAUNCH): AWS Personal Javier	FORKS: DEFAULT	LIMIT (PROMPT ON LAUNCH):
* VERBOSITY (PROMPT ON LAUNCH): 1 (Verbose)	JOB TAGS (PROMPT ON LAUNCH):	SKIP TAGS (PROMPT ON LAUNCH):
LABELS:	INSTANCE GROUPS:	JOB SLICING: 1
SHOW CHANGES (PROMPT ON LAUNCH): OFF	OPTIONS: ☐ Enable Privilege Escalation ☐ Allow Provisioning Callbacks ☐ Enable Concurrent Jobs ☐ Use Fact Cache	

Running the playbook:

NOTE: The output can be downloaded and shared:



JOBS / 158 - AWS Create Subnet

DETAILS

STATUS	● Successful
STARTED	16/6/2019 14:06:55
FINISHED	16/6/2019 14:07:02
JOB TEMPLATE	AWS Create Subnet
JOB TYPE	Run
LAUNCHED BY	admin
INVENTORY	AWS Javier Personal
PROJECT	● Ansible Training
PLAYBOOK	aws-create-subnet.yaml
CREDENTIAL	AWS Personal Javier
VERBOSITY	1 (Verbose)
EXTRA VARIABLES	YAML JSON

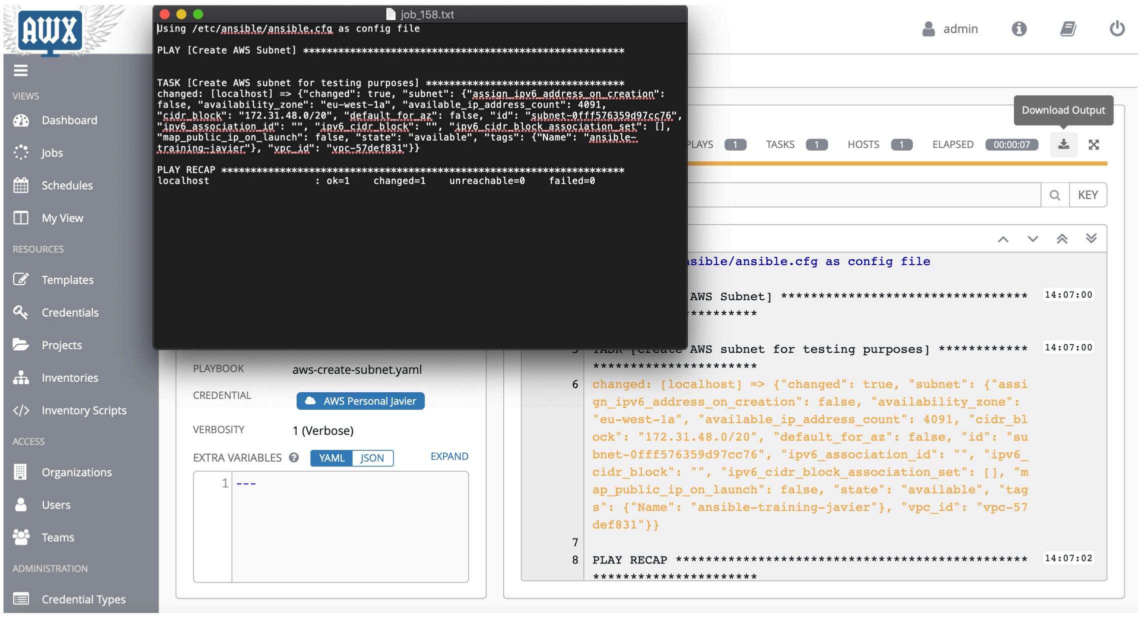
```
---
```

AWS Create Subnet

PLAYS 1 TASKS 1 HOSTS 1 ELAPSED 00:00:07

```
Using /etc/ansible/ansible.cfg as config file

PLAY [Create AWS Subnet] *********************************         14:07:00
*********************

TASK [Create AWS subnet for testing purposes] ************         14:07:00
*********************
changed: [localhost] => {"changed": true, "subnet": {"assi
gn_ipv6_address_on_creation": false, "availability_zone":
"eu-west-1a", "available_ip_address_count": 4091, "cidr_bl
ock": "172.31.48.0/20", "default_for_az": false, "id": "su
bnet-0fff576359d97cc76", "ipv6_association_id": "", "ipv6_
cidr_block": "", "ipv6_cidr_block_association_set": [], "m
ap_public_ip_on_launch": false, "state": "available", "tag
s": {"Name": "ansible-training-javier"}, "vpc_id": "vpc-57
def831"}}

PLAY RECAP ***********************************************         14:07:02
*********************
```

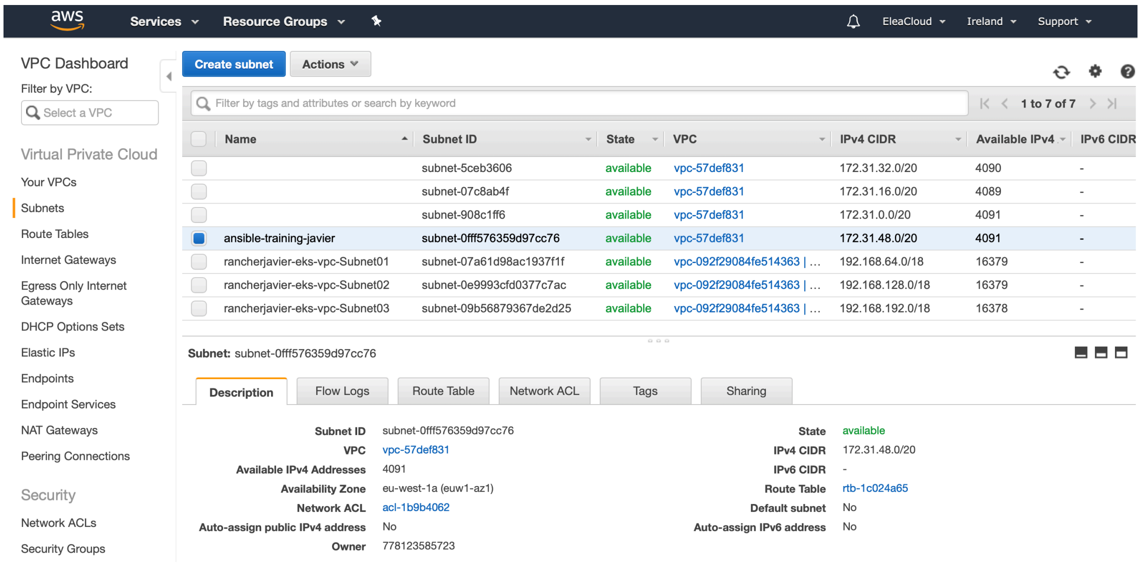
A text file with the output is generated:



job_158.txt

```
Using /etc/ansible/ansible.cfg as config file

PLAY [Create AWS Subnet] *********************************************************

TASK [Create AWS subnet for testing purposes] ************************************
changed: [localhost] => {"changed": true, "subnet": {"assign_ipv6_address_on_creation":
false, "availability_zone": "eu-west-1a", "available_ip_address_count": 4091,
"cidr_block": "172.31.48.0/20", "default_for_az": false, "id": "subnet-0fff576359d97cc76",
"ipv6_association_id": "", "ipv6_cidr_block": "", "ipv6_cidr_block_association_set": [],
"map_public_ip_on_launch": false, "state": "available", "tags": {"Name": "ansible-
training-javier"}, "vpc_id": "vpc-57def831"}}

PLAY RECAP ***********************************************************************
localhost                  : ok=1    changed=1    unreachable=0    failed=0
```

The new AWS subnet is created:



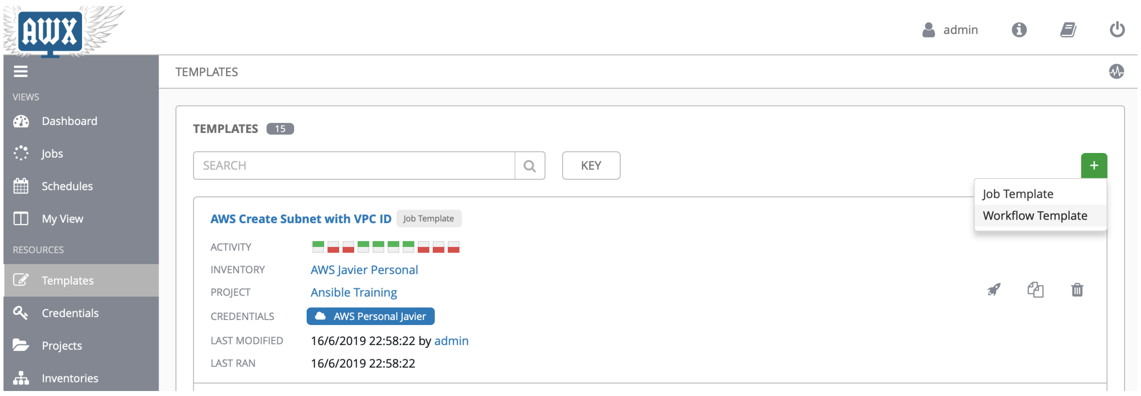
Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	IPv6 CIDR
	subnet-5ceb3606	available	vpc-57def831	172.31.32.0/20	4090	-
	subnet-07c8ab4f	available	vpc-57def831	172.31.16.0/20	4089	-
	subnet-908c1ff6	available	vpc-57def831	172.31.0.0/20	4091	-
ansible-training-javier	subnet-0fff576359d97cc76	available	vpc-57def831	172.31.48.0/20	4091	-
rancherjavier-eks-vpc-Subnet01	subnet-07a61d98ac1937f1f	available	vpc-092f29084fe514363 | …	192.168.64.0/18	16379	-
rancherjavier-eks-vpc-Subnet02	subnet-0e9993cfd0377c7ac	available	vpc-092f29084fe514363 | …	192.168.128.0/18	16379	-
rancherjavier-eks-vpc-Subnet03	subnet-09b56879367de2d25	available	vpc-092f29084fe514363 | …	192.168.192.0/18	16378	-

Subnet: subnet-0fff576359d97cc76

Subnet ID	subnet-0fff576359d97cc76	State	available
VPC	vpc-57def831	IPv4 CIDR	172.31.48.0/20
Available IPv4 Addresses	4091	IPv6 CIDR	-
Availability Zone	eu-west-1a (euw1-az1)	Route Table	rtb-1c024a65
Network ACL	acl-1b9b4062	Default subnet	No
Auto-assign public IPv4 address	No	Auto-assign IPv6 address	No
Owner	778123585723		

Let's take a glance of workflow templates and discuss how to inherit variables.

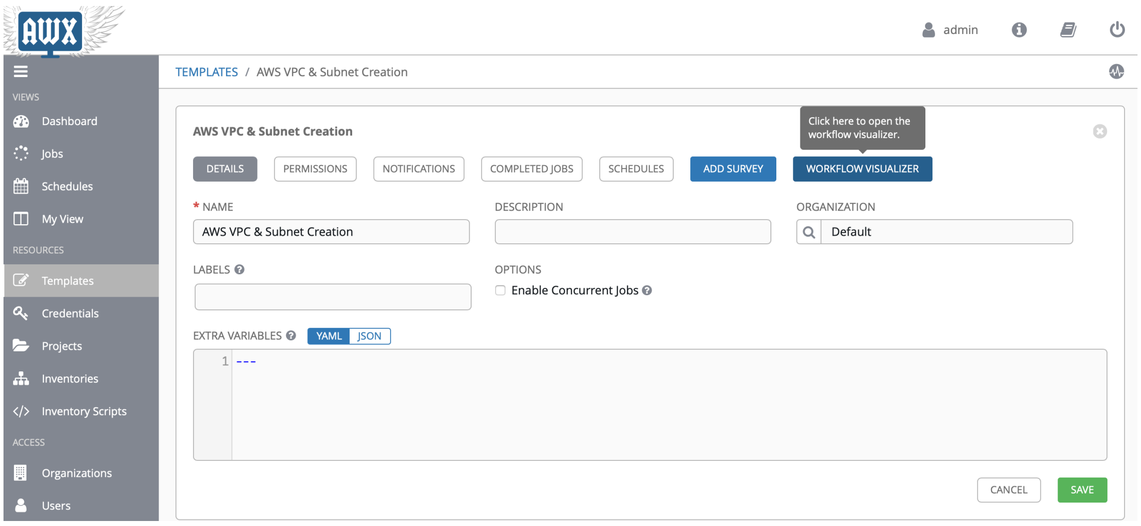
Click on Templates and add a new Workflow Template:



AWS Create Subnet with VPC ID Job Template

ACTIVITY	🟩🟥🟥🟩🟩🟩🟩🟥🟥🟥
INVENTORY	AWS Javier Personal
PROJECT	Ansible Training
CREDENTIALS	AWS Personal Javier
LAST MODIFIED	16/6/2019 22:58:22 by admin
LAST RAN	16/6/2019 22:58:22

Once the name is saved, click on Workflow Visualizer



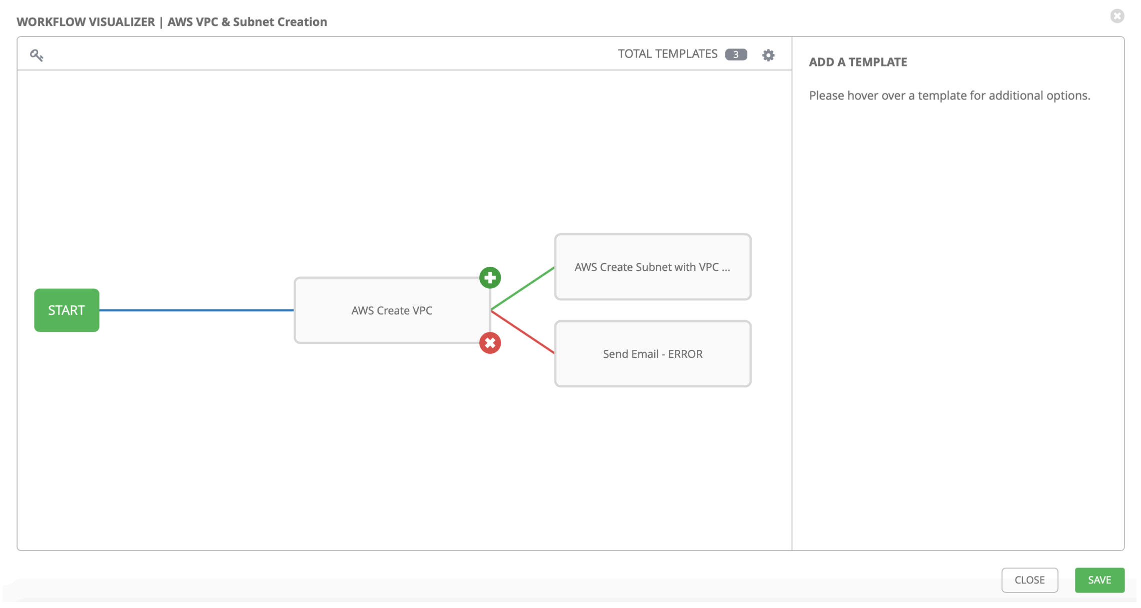
AWS VPC & Subnet Creation

* NAME	DESCRIPTION	ORGANIZATION
AWS VPC & Subnet Creation		Default
LABELS	OPTIONS	
	Enable Concurrent Jobs	

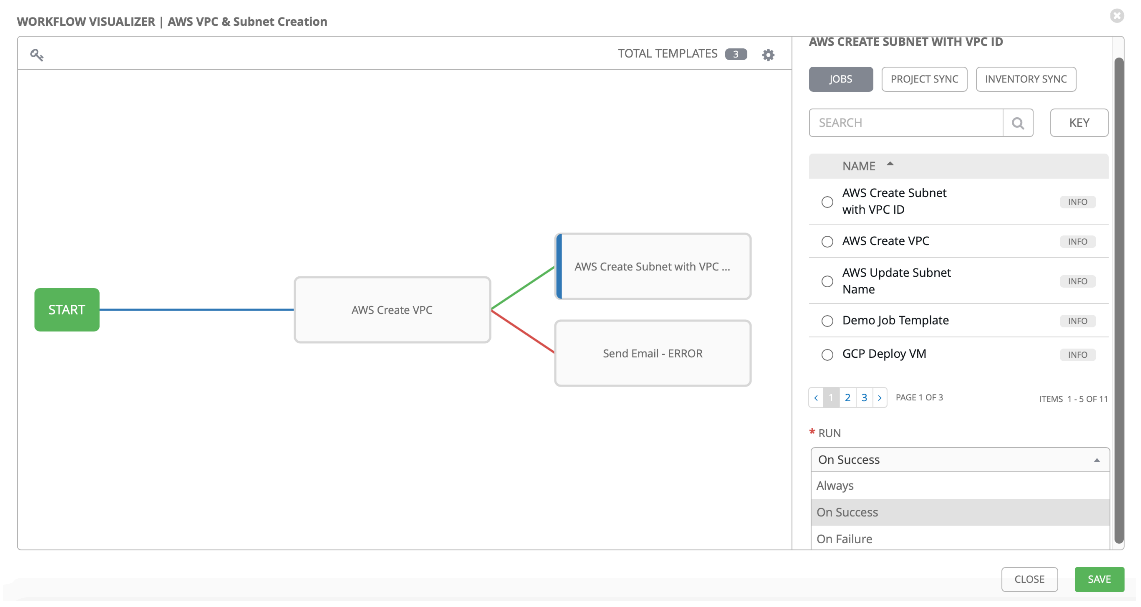
EXTRA VARIABLES

```
---
```

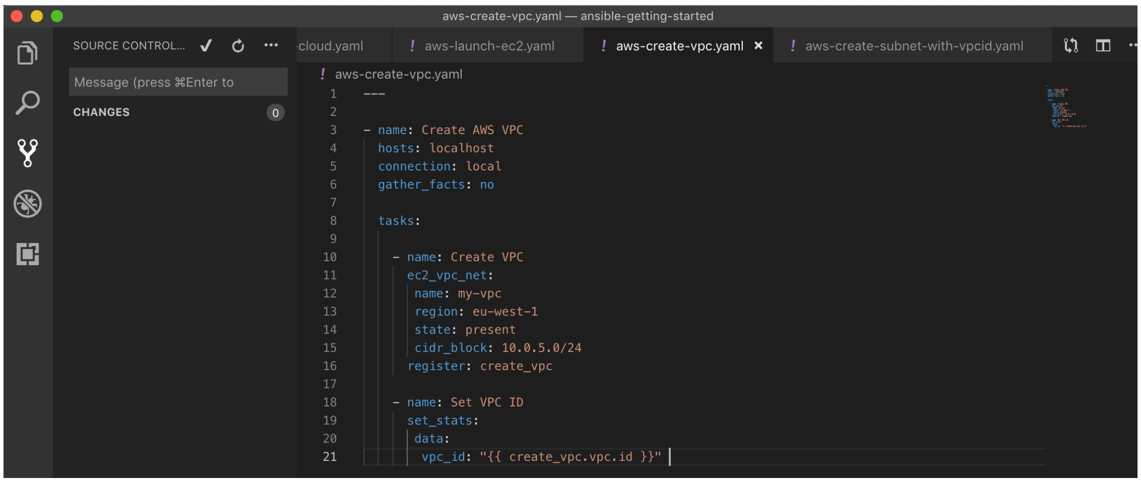
I have created the workflow below:


WORKFLOW VISUALIZER | AWS VPC & Subnet Creation
TOTAL TEMPLATES 3
START
AWS Create VPC
AWS Create Subnet with VPC ...
Send Email - ERROR
ADD A TEMPLATE
Please hover over a template for additional options.
CLOSE
SAVE

In order to add templates, click over a template and select a job, project sync or inventory sync.

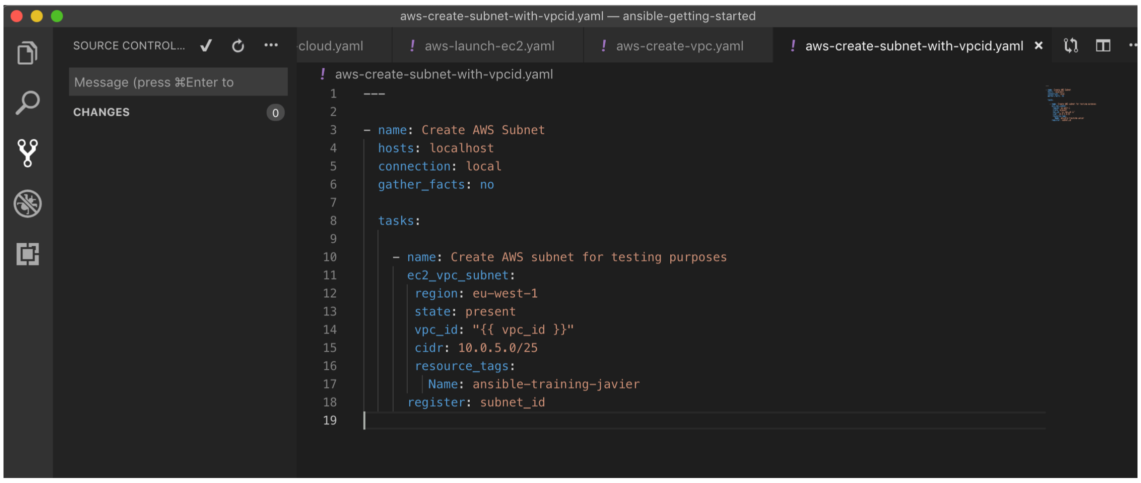


In order to pass the "vpc_id" from the AWS Create VPC playbook to "AWS Create Subnet with VPC ID", you can use the "set_stats" module:



```
---

- name: Create AWS VPC
  hosts: localhost
  connection: local
  gather_facts: no

  tasks:

    - name: Create VPC
      ec2_vpc_net:
       name: my-vpc
       region: eu-west-1
       state: present
       cidr_block: 10.0.5.0/24
      register: create_vpc

    - name: Set VPC ID
      set_stats:
       data:
        vpc_id: "{{ create_vpc.vpc.id }}"
```

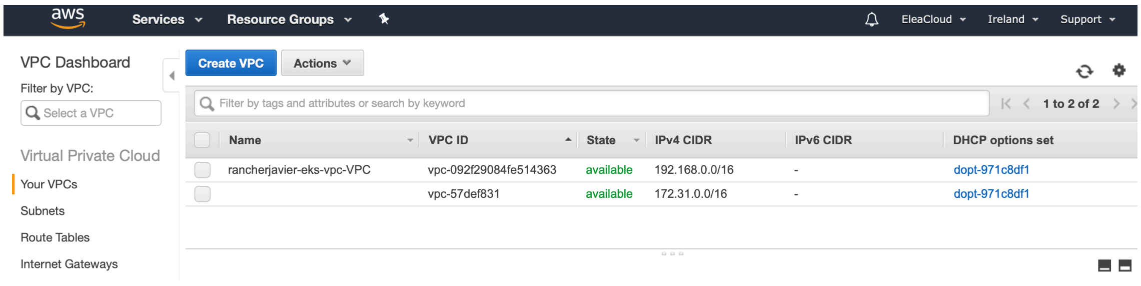
The create subnet playbook inherits that value.


aws-create-subnet-with-vpcid.yaml — ansible-getting-started
SOURCE CONTROL...
Message (press ⌘Enter to
CHANGES
0
cloud.yaml
aws-launch-ec2.yaml
aws-create-vpc.yaml
aws-create-subnet-with-vpcid.yaml
aws-create-subnet-with-vpcid.yaml

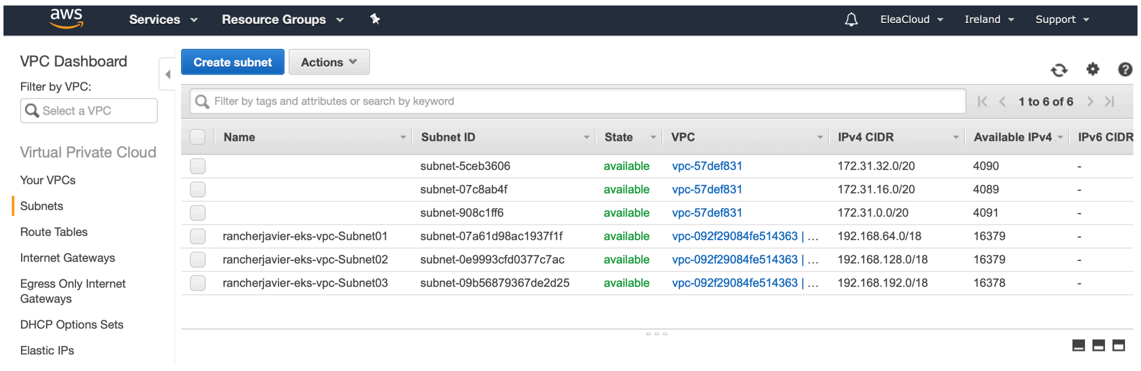
```
---

- name: Create AWS Subnet
  hosts: localhost
  connection: local
  gather_facts: no

  tasks:

    - name: Create AWS subnet for testing purposes
      ec2_vpc_subnet:
       region: eu-west-1
       state: present
       vpc_id: "{{ vpc_id }}"
       cidr: 10.0.5.0/25
       resource_tags:
         Name: ansible-training-javier
      register: subnet_id
```

Let´s run the workflow.

This is the list of VPCs and subnets created.

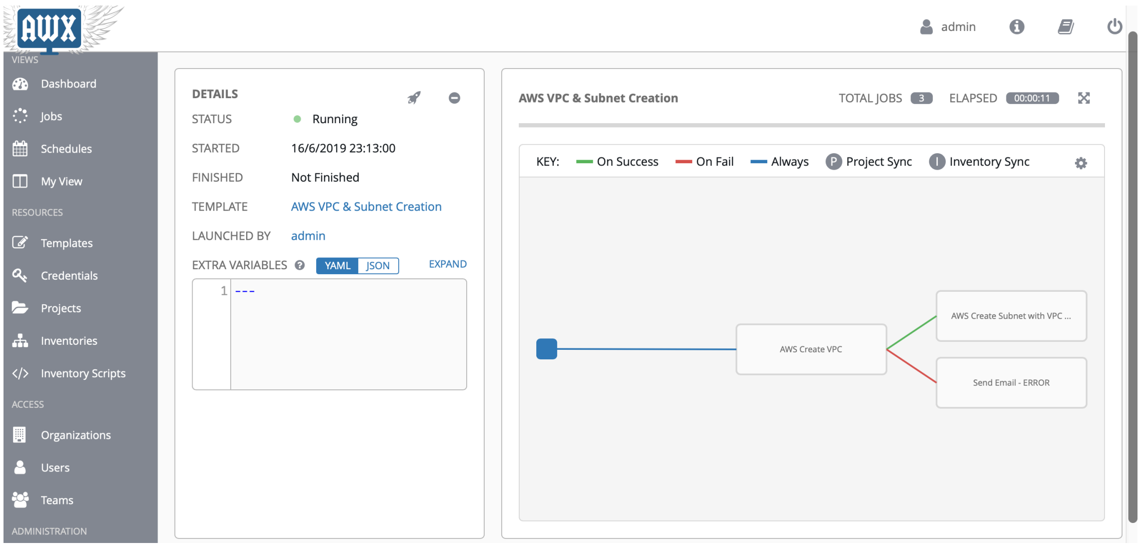


Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP options set
rancherjavier-eks-vpc-VPC	vpc-092f29084fe514363	available	192.168.0.0/16	-	dopt-971c8df1
	vpc-57def831	available	172.31.0.0/16	-	dopt-971c8df1



Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	IPv6 CIDR
	subnet-5ceb3606	available	vpc-57def831	172.31.32.0/20	4090	-
	subnet-07c8ab4f	available	vpc-57def831	172.31.16.0/20	4089	-
	subnet-908c1ff6	available	vpc-57def831	172.31.0.0/20	4091	-
rancherjavier-eks-vpc-Subnet01	subnet-07a61d98ac1937f1f	available	vpc-092f29084fe514363 | ...	192.168.64.0/18	16379	-
rancherjavier-eks-vpc-Subnet02	subnet-0e9993cfd0377c7ac	available	vpc-092f29084fe514363 | ...	192.168.128.0/18	16379	-
rancherjavier-eks-vpc-Subnet03	subnet-09b56879367de2d25	available	vpc-092f29084fe514363 | ...	192.168.192.0/18	16378	-

Running the workflow:



DETAILS

STATUS ● Running

STARTED 16/6/2019 23:13:00

FINISHED Not Finished

TEMPLATE AWS VPC & Subnet Creation

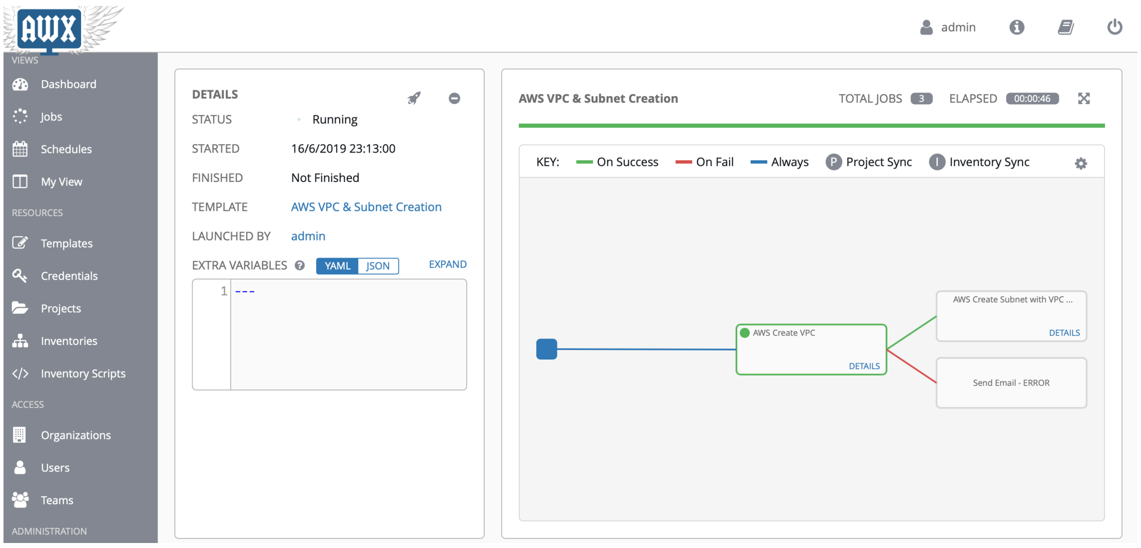
LAUNCHED BY admin

EXTRA VARIABLES YAML JSON EXPAND

```
---
```

AWS VPC & Subnet Creation TOTAL JOBS 3 ELAPSED 00:00:11

First job is successfully completed:



DETAILS

STATUS Running

STARTED 16/6/2019 23:13:00

FINISHED Not Finished

TEMPLATE AWS VPC & Subnet Creation

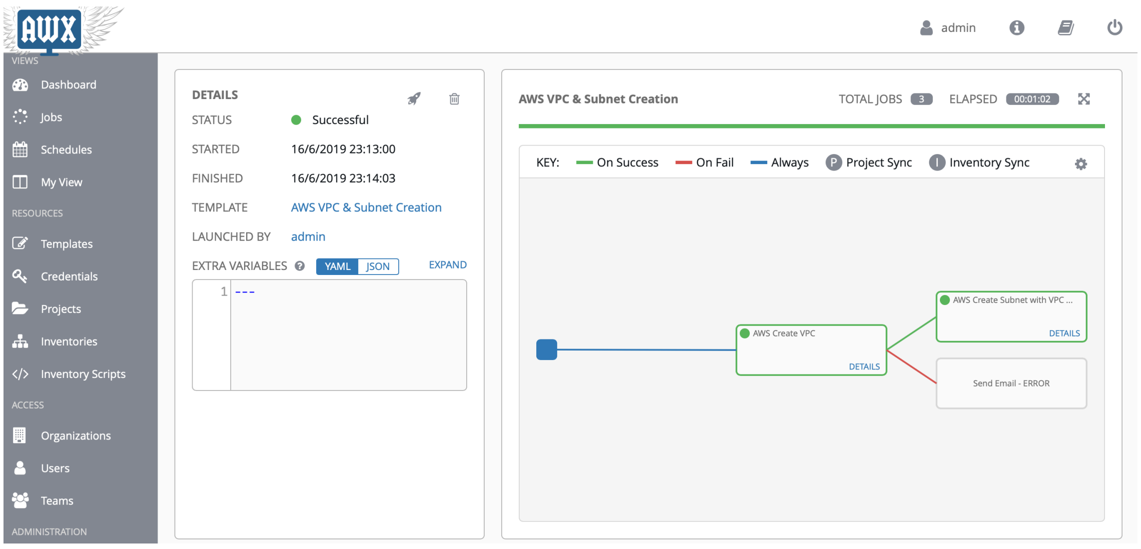
LAUNCHED BY admin

EXTRA VARIABLES YAML JSON EXPAND

```
---
```

AWS VPC & Subnet Creation TOTAL JOBS 3 ELAPSED 00:00:46

Now, the second job is also executed:



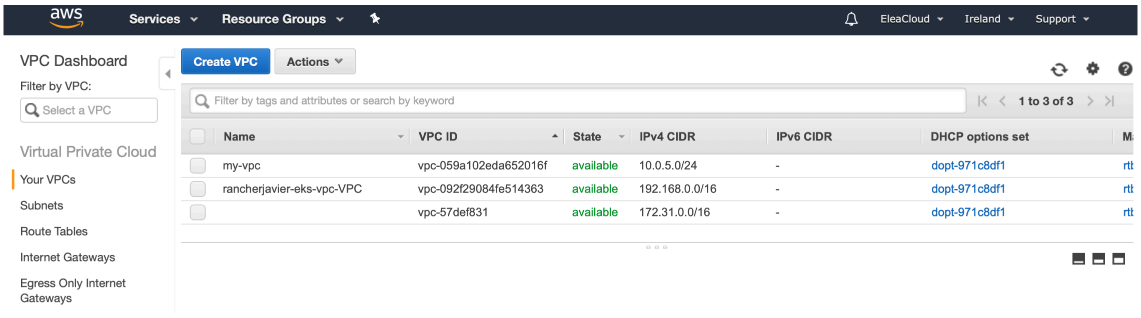
DETAILS

STATUS	Successful
STARTED	16/6/2019 23:13:00
FINISHED	16/6/2019 23:14:03
TEMPLATE	AWS VPC & Subnet Creation
LAUNCHED BY	admin
EXTRA VARIABLES	YAML JSON

```
---
```

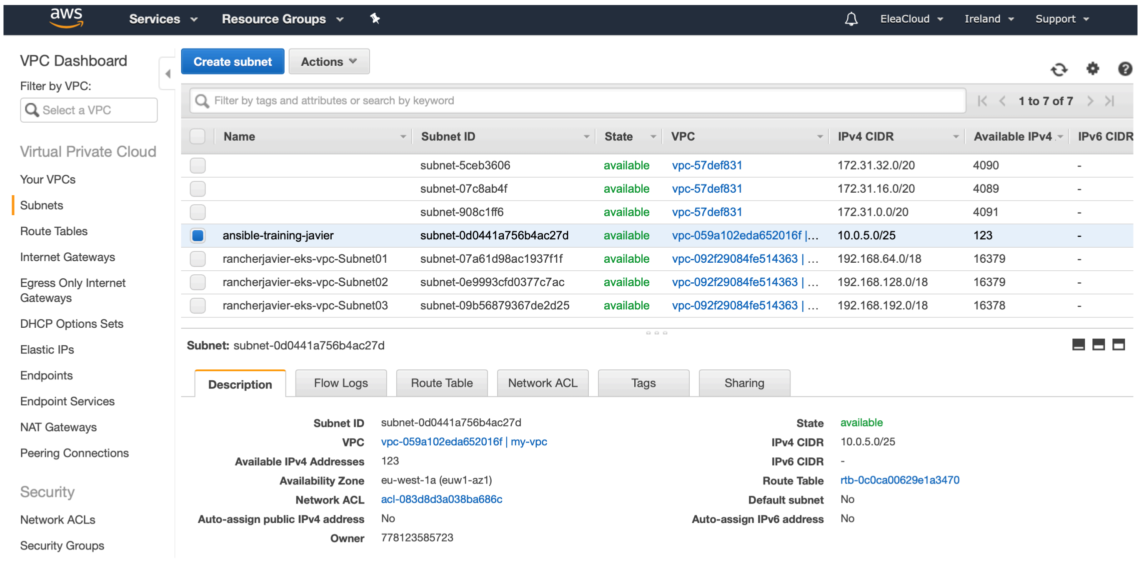
AWS VPC & Subnet Creation TOTAL JOBS 3 ELAPSED 00:01:02

The new VPC called "my-vpc" is created.



Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP options set	M
my-vpc	vpc-059a102eda652016f	available	10.0.5.0/24	-	dopt-971c8df1	rt
rancherjavier-eks-vpc-VPC	vpc-092f29084fe514363	available	192.168.0.0/16	-	dopt-971c8df1	rt
	vpc-57def831	available	172.31.0.0/16	-	dopt-971c8df1	rt

A new subnet called "ansible-training-javier" is also provisioned in the VPC created in the same workflow.



Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	IPv6 CIDR
	subnet-5ceb3606	available	vpc-57def831	172.31.32.0/20	4090	-
	subnet-07c8ab4f	available	vpc-57def831	172.31.16.0/20	4089	-
	subnet-908c1ff6	available	vpc-57def831	172.31.0.0/20	4091	-
ansible-training-javier	subnet-0d0441a756b4ac27d	available	vpc-059a102eda652016f |...	10.0.5.0/25	123	-
rancherjavier-eks-vpc-Subnet01	subnet-07a61d98ac1937f1f	available	vpc-092f29084fe514363 | ...	192.168.64.0/18	16379	-
rancherjavier-eks-vpc-Subnet02	subnet-0e9993cfd0377c7ac	available	vpc-092f29084fe514363 | ...	192.168.128.0/18	16379	-
rancherjavier-eks-vpc-Subnet03	subnet-09b56879367de2d25	available	vpc-092f29084fe514363 | ...	192.168.192.0/18	16378	-

Subnet: subnet-0d0441a756b4ac27d

Subnet ID	subnet-0d0441a756b4ac27d	**State**	available
VPC	vpc-059a102eda652016f | my-vpc	**IPv4 CIDR**	10.0.5.0/25
Available IPv4 Addresses	123	**IPv6 CIDR**	-
Availability Zone	eu-west-1a (euw1-az1)	**Route Table**	rtb-0c0ca00629e1a3470
Network ACL	acl-083d8d3a038ba686c	**Default subnet**	No
Auto-assign public IPv4 address	No	**Auto-assign IPv6 address**	No
Owner	778123585723		

Deploying to Google Cloud Platform

Inventory

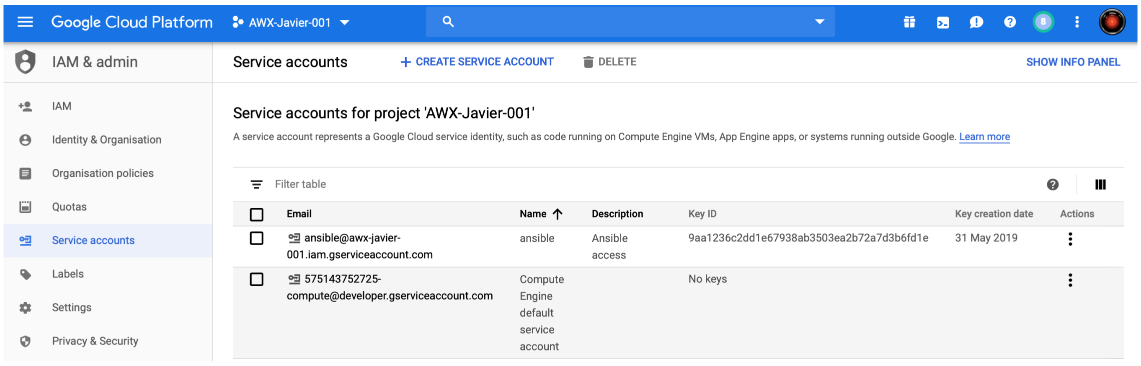
You will need to install additional packages in order to gather information about GCP-based hosts:

```
pip install requests google-auth apache-libcloud
```

First, you have to create a new service account for Ansible. Open the Google Cloud dashboard and select your project from the top header.

Select IAM & admin then go to the Service Accounts section. You can create a new service account by clicking on the Create Service Account button at the top.

From the pop-up, we can create a service account and download the JSON credentials file as shown below.



Service accounts for project 'AWX-Javier-001'

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. Learn more

Email	Name	Description	Key ID	Key creation date	Actions
ansible@awx-javier-001.iam.gserviceaccount.com	ansible	Ansible access	9aa1236c2dd1e67938ab3503ea2b72a7d3b6fd1e	31 May 2019	⋮
575143752725-compute@developer.gserviceaccount.com	Compute Engine default service account		No keys		⋮

Considering Ansible Engine, once we have the credentials we should put them in roles/gce/vars/secrets.yml:

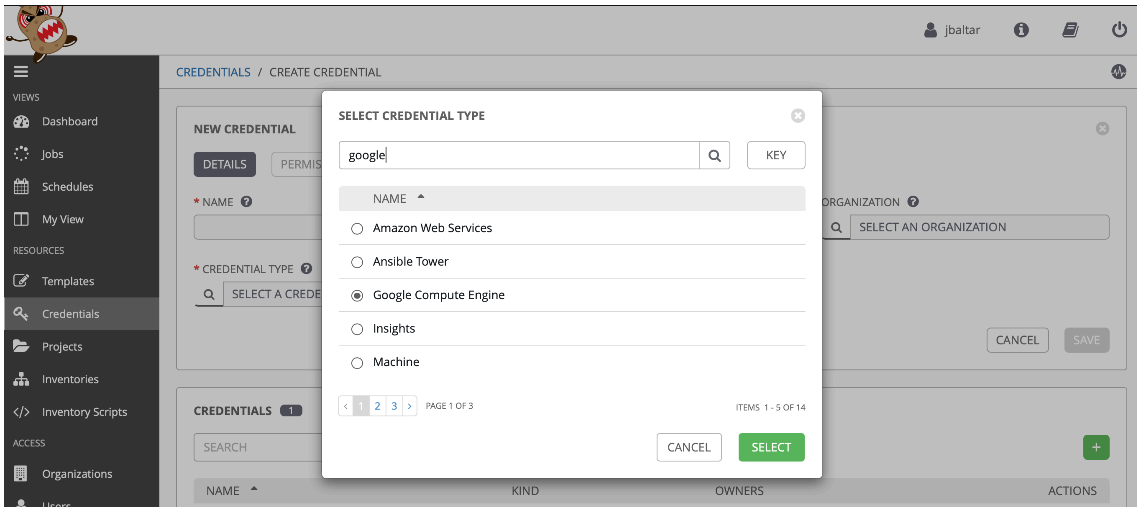
```
---
gs_access_key: XXXXXXXXXXXXXXXXXX
gs_secret_key: XXXXXXXXXXXXXXXXXX
```

Now, we encrypt them:

```
$ ansible-vault encrypt roles/gce/vars/secrets.yml
```

From AWX perspective, the configuration is similar to AWS.

Click on credentials and choose Google Compute Engine.



SELECT CREDENTIAL TYPE

google

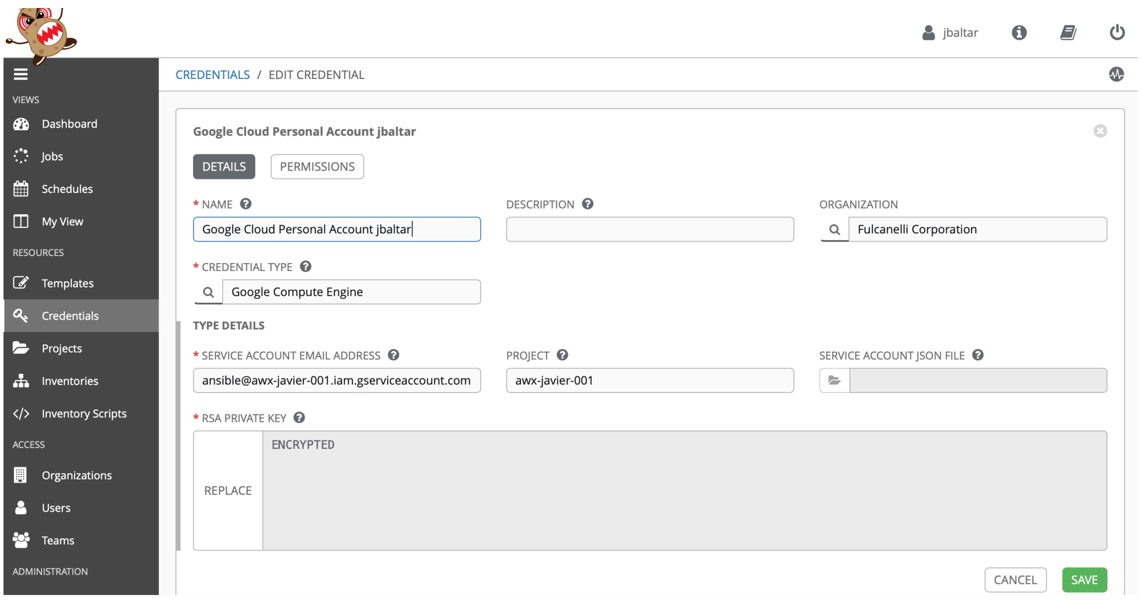
NAME
○ Amazon Web Services
○ Ansible Tower
● Google Compute Engine
○ Insights
○ Machine

PAGE 1 OF 3 ITEMS 1 - 5 OF 14

CANCEL SELECT

Add the service account details retrieved from the file downloaded:

Once the credential is saved, it is automatically encrypted.



Google Cloud Personal Account jbaltar

* NAME: Google Cloud Personal Account jbaltar	DESCRIPTION:	ORGANIZATION: Fulcanelli Corporation
* CREDENTIAL TYPE: Google Compute Engine		

TYPE DETAILS

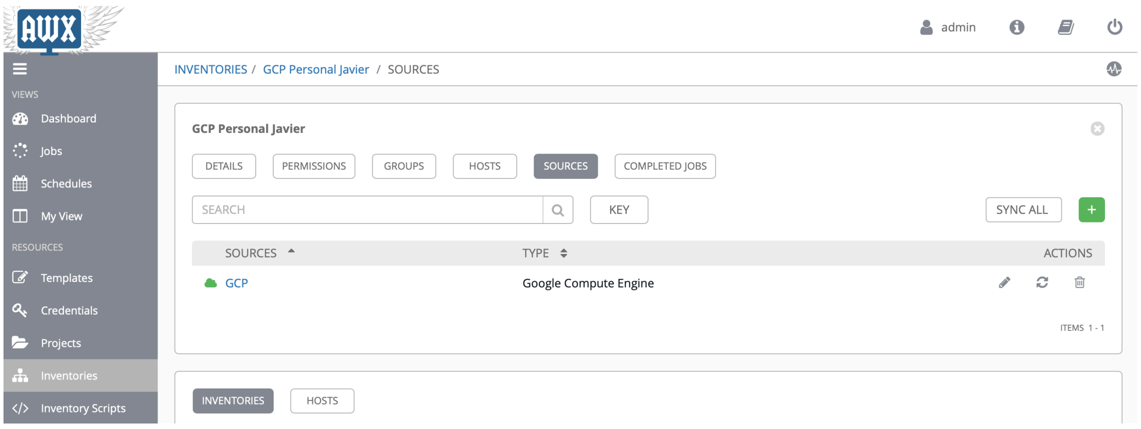
* SERVICE ACCOUNT EMAIL ADDRESS: ansible@awx-javier-001.iam.gserviceaccount.com	PROJECT: awx-javier-001	SERVICE ACCOUNT JSON FILE:

* RSA PRIVATE KEY: ENCRYPTED

You can generate a ssh key for Ansible and uploaded it to the Google Cloud console: Click on Compute Engine > Metadata > SSH Keys and add the new key.

The inventory configuration is similar to the one we did for AWS.

Click on Inventories > Sources and add a Google Compute Engine source type.



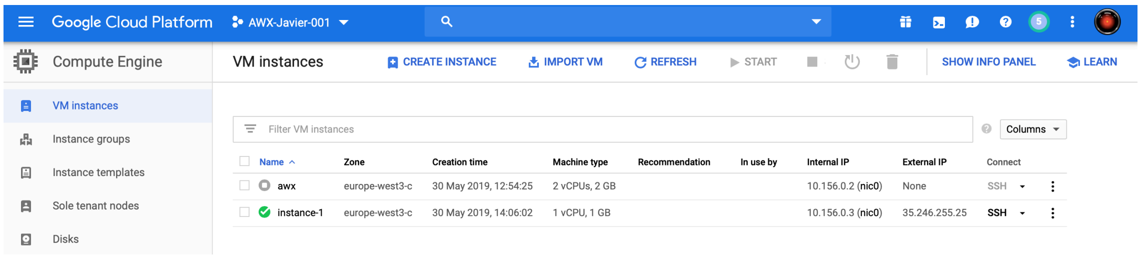
SOURCES	TYPE	ACTIONS
GCP	Google Compute Engine	

ITEMS 1 - 1

As an example, the following playbook launched a new VM in GCP.

```
- name: Create Google Cloud VM
  hosts: localhost
  gather_facts: no
  connection: local

  vars:
    machine_type: n1-standard-1
    image: centos7
    service_account_email: ansible@awx-
javier-001.iam.gserviceaccount.com
    credentials_file: /home/googlecloud/playbooks/key.json
    project_id: AWX-Javier-001

  tasks:
    - name: Launch VM
      gce:
          instance_names: instance-1
          machine_type: "{{ machine_type }}"
          image: "{{ image }}"
          service_account_email: "{{ service_account_email }}"
          credentials_file: "{{ credentials_file }}"
          project_id: "{{ project_id }}"
```

Running the playbook, the new virtual machine is created:



Name	Zone	Creation time	Machine type	Recommendation	In use by	Internal IP	External IP	Connect
awx	europe-west3-c	30 May 2019, 12:54:25	2 vCPUs, 2 GB			10.156.0.2 (nic0)	None	SSH
instance-1	europe-west3-c	30 May 2019, 14:06:02	1 vCPU, 1 GB			10.156.0.3 (nic0)	35.246.255.25	SSH