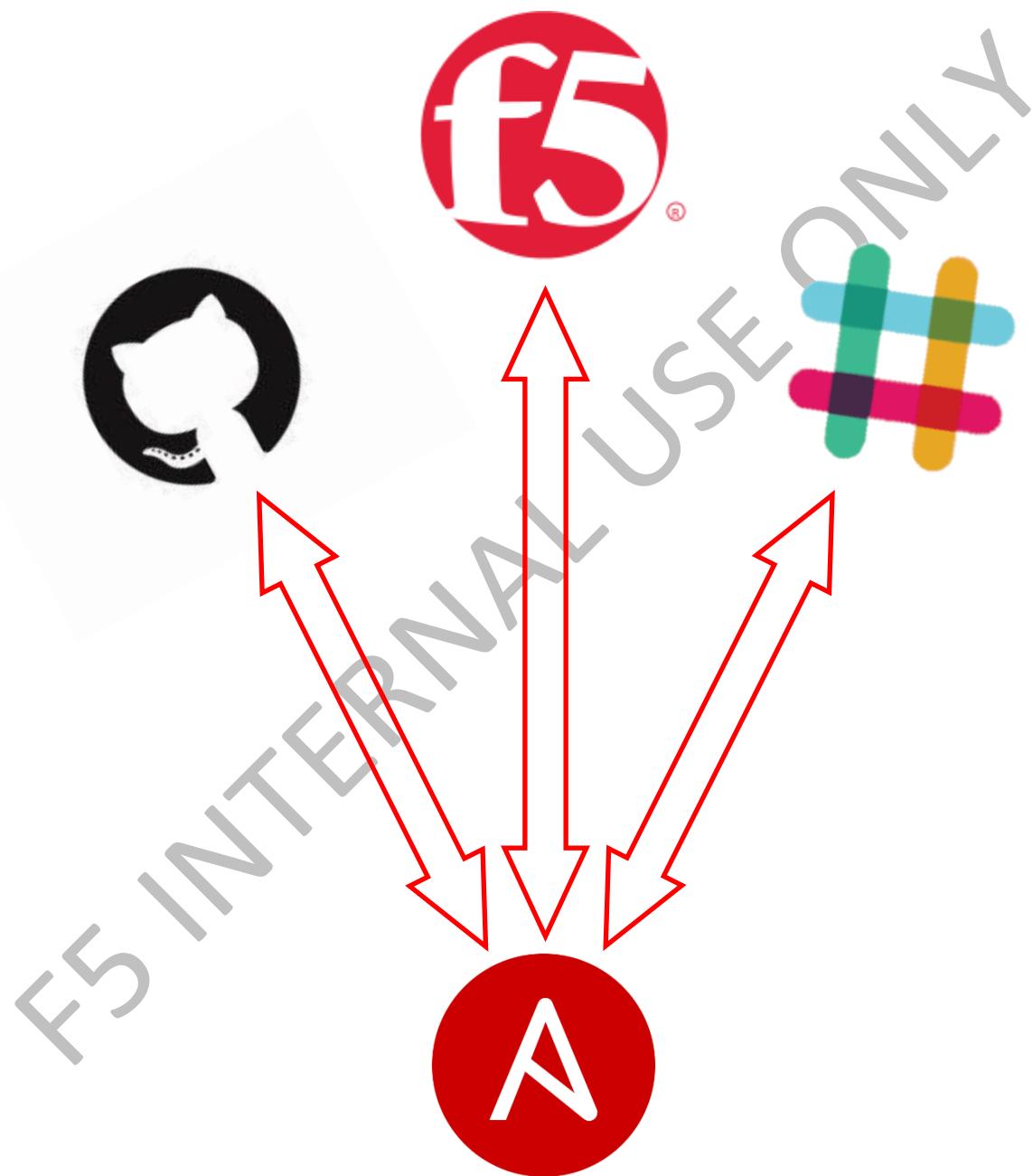


Ansible Tower with Github, Slack and BIGIP



Contents

A. Full Disclosure and References	3
1. Installation and Creation of Github Account and Repository.....	4
1.1 Create your very own Github Account!	4
1.2 Create your SSH keypair.....	4
1.3 [Optional] Download and install Git	7
1.4 Adding Local Repository to Github	9
1.5 Inventory File from Local Repository to Github.....	9
2. Slack Account Creation.....	11
2.1 Create your own Slack Workspace and Channel!.....	11
2.2 Slack Legacy API Token Creation.....	13
3. Ansible Tower Installation and Configuration	14
3.1 Ansible Tower Installation	15
3.2 Ansible Tower Licensing.....	17
3.3 Ansible Tower Dashboard	19
3.4 Exploring the Ansible Tower Dashboard and Interface	20
3.5 Explanation of Dashboard and interface objects.....	21
3.6 Ansible Tower Settings Menu	22
3.7 Ansible Tower Organization Creation	24
3.8 Creating an Ansible Tower Operator User Account.....	27
3.9 Creating an Ansible Tower Team	29
3.10 Ansible Tower Credential Creation for Source Control Management (Github)	31
3.11 Ansible Tower Project Creation	33
3.12 [Optional] Adding a Project manually.....	35
3.13 Creation of an Ansible Tower Inventory	36
3.14 [Optional] Using awx-manage to manually import inventory file	40
3.15 Modification of playbook to use Ansible Vault.....	42
3.16 Ansible Tower Notification Configuration with Slack	47
3.17 Ansible Tower Job Template Configuration.....	49
3.18 Configuring Ansible Tower Surveys	52
3.19 Role-based Access Control.....	54
3.20 Configuring an Ansible Workflow Template	56
3.21 Running <i>dev</i> F5 modules	63
3.22 Upgrading.....	63

A. Full Disclosure and References

All product names, logos, and brands are property of their respective owners. All company, product and service names used in this website are for identification purposes only. Use of these names, logos, and brands does not imply endorsement.

Materials (wordings and imagery) from the following sites may have been reproduced in this document :

<https://help.github.com/articles/adding-an-existing-project-to-github-using-the-command-line/>

<https://help.github.com/articles/adding-a-file-to-a-repository-using-the-command-line/>

<https://git-scm.com/docs/git-config>

<https://git-scm.com/docs/git-pull>

<https://git-scm.com/docs/git-add>

<https://git-scm.com/docs/git-commit>

<https://help.github.com/articles/set-up-git/>

<https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/>

<http://releases.ansible.com/ansible-tower/setup/>

<http://docs.ansible.com/ansible-tower/3.2.3/html/userguide/>

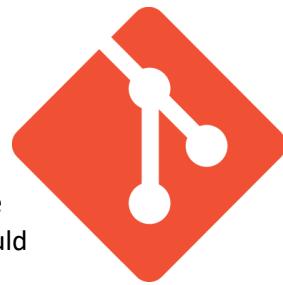
<http://docs.ansible.com/ansible-tower/3.2.3/html/administration/>

This document is intended for internal use only and shall not be distributed outside of F5 Networks.

1. Installation and Creation of Github Account and Repository

1.1 Create your very own Github Account!

Access <https://github.com/> and sign up for your *Unlimited* (free) public repositories Github account. Complete the email verify process and start a project. Remember your repository name (you will need it later) and initialize the repository with a README. Your README can contain any string that would describe your Github repository.



Note

If you already have a Github account and you would like to use it for this Lab, skip to <>

Your new Github repository should look like this:

The screenshot shows a GitHub repository page. At the top, there's a navigation bar with links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the navigation, the repository name 'penghonengineer / ansible-repo' is displayed, with a green box around it and a green arrow pointing to the text 'Replace with your own account/repository'. The main content area shows a 'No description, website, or topics provided.' message, an 'Edit' button, and a commit history. The first commit is from 'penghonengineer' titled 'Initial commit' made 'a minute ago'. The commit message contains the text 'ansible-repo', which is also highlighted with a green box. At the bottom, there are links for 'Terms', 'Privacy', 'Security', 'Status', 'Help', 'Contact GitHub', 'API', 'Training', 'Shop', 'Blog', and 'About'.

1.2 Create your SSH keypair

Using the provided root credentials SSH to your assigned Ansible Tower Controller/Git Local Repository and generate a keypair. This keypair will be used to automatically authenticate to Github when you push the latest changes from your local repository to your Github master branch:

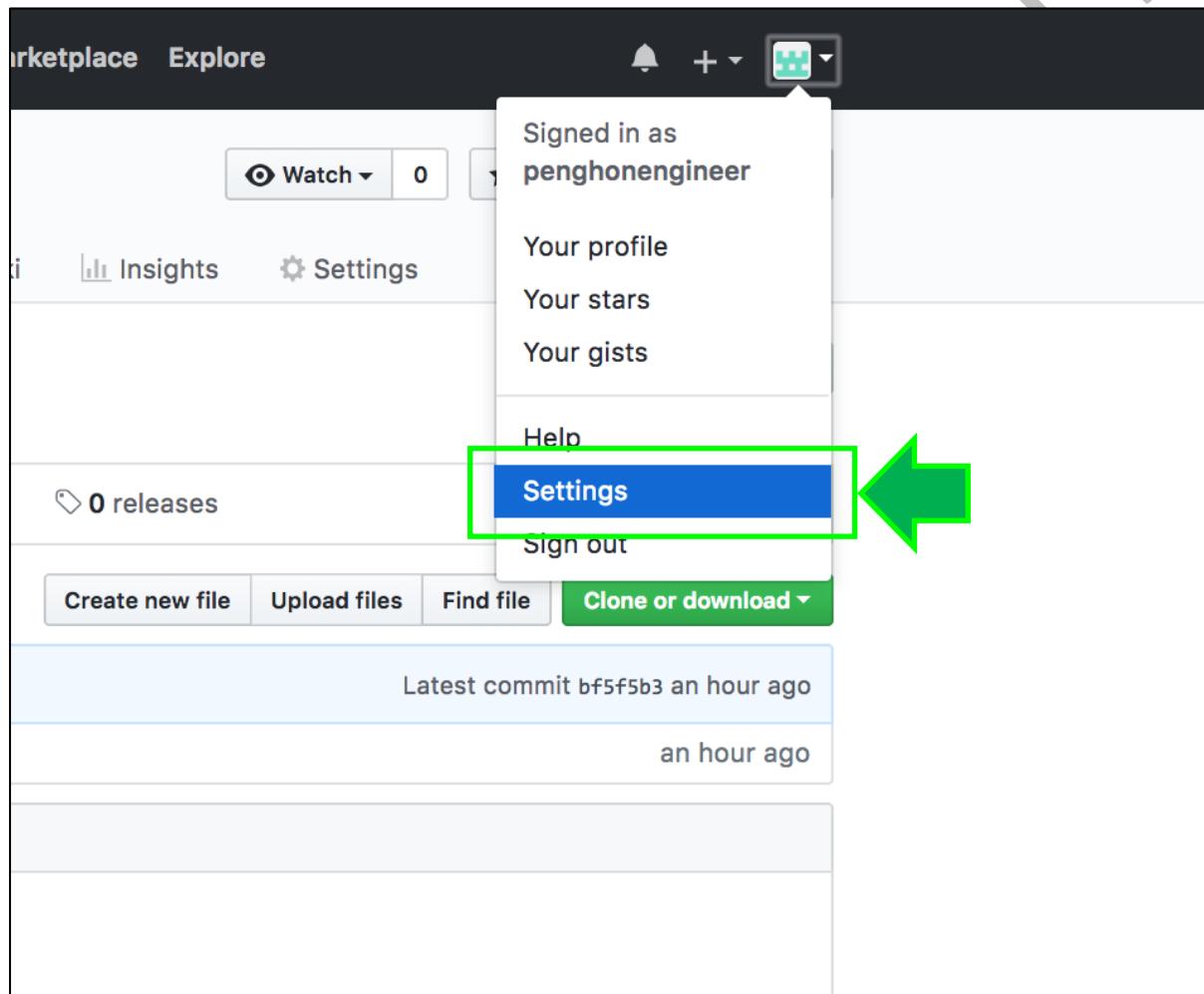
```
[root@localhost ~ ]# ssh-keygen -t rsa -b 4096
```

Note

For the purpose of this Lab do not use any SSH passphrase and accept all default parameters when generating your SSH private key.

Copy out the contents of the **SSH public key** to a text editor and log back into your Github account. We will be adding the SSH public key from your assigned Ansible Tower Controller/Git Local Repository to your Github account.

From your Github account select **Settings** from the menu as shown below:



Select “SSH and GPG keys” from the left menu and click on **New SSH key**:

The screenshot shows the GitHub "Personal settings" page. On the left, a sidebar lists options: Profile, Account, Emails, Notifications, Billing, **SSH and GPG keys**, Security, and Blocked users. The "SSH and GPG keys" option is highlighted with a red box and has a large green arrow pointing to it from the left. On the right, the "SSH keys" section is displayed, showing a message: "There are no SSH keys associated with your account." A "New SSH key" button is located in the top right corner of this section. Below it, the "GPG keys" section is shown with a similar message and a "New GPG key" button.

Add in the previously copied SSH public key starting with “ssh-rsa” and provide a meaningful title for this public key and click on “Add SSH key”:

The screenshot shows the "SSH keys / Add new" page. The left sidebar includes the "SSH and GPG keys" option, which is highlighted with a red box and has a large green arrow pointing to it from the left. The main form has a "Title" field containing "Local Repo SSH key" (which is also highlighted with a red box) and a "Key" field containing a long SSH public key (preceded by "ssh-rsa"). At the bottom is a "Add SSH key" button.

Once Github accepts the public key, you should be able to see a summary page of the available SSH keys associated to your Github account. Notice that your new key has not been used before. This is expected and will change once you are able to sync and commit file changes from your local repo to Github:

The screenshot shows the "SSH keys" summary page. The left sidebar includes the "SSH and GPG keys" option. The main area displays a table of keys. One row for "Local Repo SSH key" is shown, with its status being "Never used — Read/write". A "Delete" button is visible at the end of this row. A large green arrow points to the "Never used — Read/write" status text.

[Optional] Check that the fingerprint of the public key that you have uploaded is the same as what you have on the Ansible Controller:

```
[root@localhost ~]# ssh-keygen -lf ~/.ssh/id_rsa.pub  
4096 03:e5:82:30:ee:57:da:32:9b:c4:83:e9:69:34:01:fd penghon@engineer.com (RSA)
```

1.3 [Optional] Download and install Git

We do not need the latest and greatest Git. Download the available Git (git **version 1.8.3.1**) from the yum repositories.

Using root credentials, SSH to your Ansible Tower Controller and install the latest available Git package from the yum repositories:

```
[root@localhost ~]# yum install git -y
```

Check that git has been installed successfully:

```
[root@localhost ~]# git --version
```

We will now create a project directory to use as local repository. You will use this local repository to push your playbooks and inventory files to Github.

Create your local repository. You can replace the path */git/ansible-repo/* with any directory that you would like to use instead. Throughout this document we will use */git/ansible-repo/* has the local repository. If you would like to use a different folder, just remember to replace any instance of */git/ansible-repo* with your desired directory:

```
[root@localhost ~]# mkdir -p /git/ansible-repo
```

Access your local repository:

```
[root@localhost ~]# cd /git/ansible-repo
```

Initialize your local repository:

```
[root@localhost ansible-repo]# git init .
```

Add the URL for the remote repository where you will be pushing from your local repository. We will be using SSH to securely push commits to Github. Remember to change the strings that are highlighted in red to your own Github account and repository URL:

```
[root@localhost ansible-repo]#  
git remote add origin git@github.com:penghonengineer/ansible-repo.git
```

Setting push.default to simple, which is same as upstream except that if the branch and the local branch are different, git will not push the commits.

```
[root@localhost ansible-repo]#  
git config --global push.default simple
```

Setting the default user name to be recorded in any newly created commits.

```
[root@localhost ansible-repo]#  
git config --global user.name penghonengineer
```

Setting the default user email to be recorded in any newly created commits.

```
[root@localhost ansible-repo]#  
git config --global user.email penghon@engineer.com
```

Merging Github master branch with local repository:

```
[root@localhost ansible-repo]#  
git pull git@github.com:penghonengineer/ansible-repo master
```

Create *testfile* in local repository:

```
[root@localhost ansible-repo]# touch testfile
```

Adding new content to git index and prepare content to be staged for the next commit:

```
[root@localhost ansible-repo]# git add testfile
```

Adding a commit message and storing the the contents of the index in a new commit:

```
[root@localhost ansible-repo]#  
git commit -m "Uploading test file" testfile
```

Set the branch to master once and for subsequent commits you can omit the branch that you are merging your local repository to:

```
[root@localhost ansible-repo]#  
git push --set-upstream origin master
```

Once you can see the *testfile* appearing in your Github repository, you are all set!

If you have made a mistake with any of your git config command, you can edit it via:

```
git config --global --unset { parameter }
```

Or:

```
Git config --global --edit
```

Or:

```
git config --global --replace-all { parameter } "{ correct value }"
```

1.4 Adding Local Repository to Github

We will be using sample (but working) BIGIP playbooks in this lab in order to save time.

From your Ansible Tower Controller/Local Git Repository, copy the sample BIGIP playbook **/var/tmp/sample_files/change_bigip_hostname.yaml** to your git local repo directory **/git/ansible-repo/** and perform similar steps to earlier section on Github:

```
[root@localhost ~]# cp /var/tmp/sample_files/change_bigip_hostname.yaml /git/ansible-repo/
[root@localhost ~]# cd /git/ansible-repo/
[root@localhost ansible-repo]# ls
change_bigip_hostname.yaml README.md
[root@localhost ansible-repo]# git add change_bigip_hostname.yaml
[root@localhost ansible-repo]#
git commit -m "Sample BIGIP playbook" change_bigip_hostname.yaml
[master 95a68c5] Sample BIGIP playbook
 1 file changed, 17 insertions(+)
 create mode 100644 change_bigip_hostname.yaml
[root@localhost ansible-repo]# git push
Counting objects: 4, done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 529 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To git@github.com:penghonengineer/ansible-repo.git
 1bf32af..95a68c5 master -> master
[root@localhost ansible-repo]#
```

1.5 Inventory File from Local Repository to Github

Copy the **/var/tmp/sample_files/inventory** to your local git repository and upload to your Github account:

```
[root@localhost ansible-repo]#
cp /var/tmp/sample_files/inventory /git/ansible-repo/
[root@localhost ansible-repo]# git add inventory
[root@localhost ansible-repo]#
git commit -m "Adding sample inventory file"
[master 3b1d64d] Adding sample inventory file
 1 file changed, 2 insertions(+)
 create mode 100644 inventory
[root@localhost ansible-repo]# git push
Counting objects: 4, done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 324 bytes | 0 bytes/s, done.
```

```
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To git@github.com:penghonengineer/ansible-repo.git
  95a68c5..3b1d64d master -> master
```

Once the git push has been completed, validate that the files have been uploaded to Github:

No description, website, or topics provided.

Add topics

13 commits 1 branch 0 releases 2 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

File	Message	Date
README.md	Initial commit	16 days ago
change_bigip_hostname.yaml	Fixed wrong variable	11 days ago
inventory	Adding sample inventory file	12 days ago
save_config.yaml	Adding now BIGIP playbook	11 days ago

Repeat the above steps by adding `/var/tmp/sample_files/save_config.yaml` to your Github repository.

2. Slack Account Creation

2.1 Create your own Slack Workspace and Channel!

Using a browser access the following URL: <https://slack.com>

Follow the instructions after clicking on “Create a new workspace”. If you already have your own slack workspace you can skip the following few pages until reach the Legacy API token section.



Start with a workspace

 Find your Slack workspace
Join or sign in to existing workspaces. >

 Create a new workspace
Get your company or organization on Slack. >

A large green arrow points from the "Create a new workspace" button to the second screenshot below.

Create a new workspace

To make a workspace from scratch, please confirm your email address.

Confirm

Key in the Stack confirmation code. Complete the profile field and continue to password configuration.

Key in your password, create the slack workspace and skip the “Send Invitations”.

Create a channel within your newly created Slack workspace. We will be using this channel to send notifications from your Ansible Tower.

Create a channel

Channels are where your members communicate. They're best when organized around a topic – #leads, for example.

Public

Anyone in your workspace can view and join this channel.

Name

tower_notifications

Names must be lowercase, without spaces or periods, and shorter than 22 characters.

Purpose (optional)

Alerts

What's this channel about?

Send invites to: (optional)

Search by name

Select up to 1000 people to add to this channel.

Cancel

Create Channel

F5

2.2 Slack Legacy API Token Creation

Using a new browser tab, access <https://api.slack.com/custom-integrations/legacy-tokens> and select “Create token”.

Copy and save the value of this token to a text editor. You will need the value of this token when you configure Ansible Tower Slack notification.

Legacy tokens

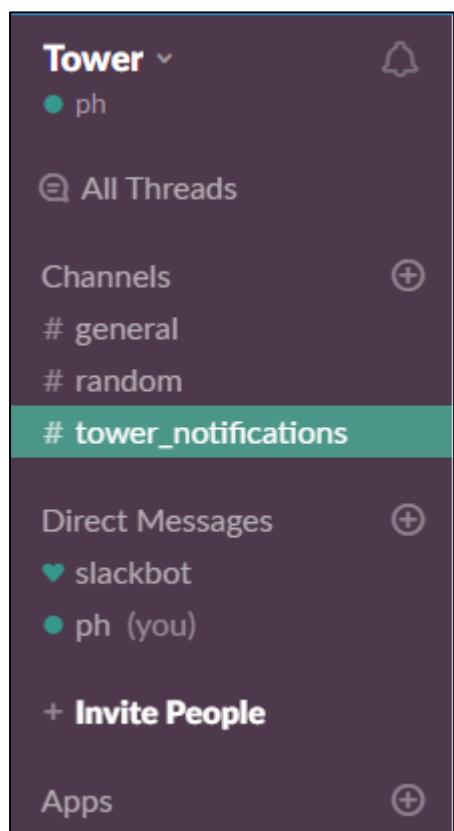
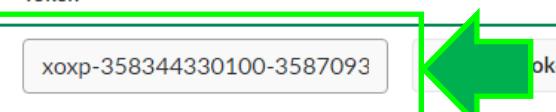
⚠️ You're viewing documentation on legacy custom integrations, an older way for teams to build into their Slack team. To securely utilize the newest platform features like message buttons & the Events API, build [internal integrations](#) as part of a [Slack app](#) just for your team instead. [Get started.](#)

Legacy token generator

Use this handy tool to quickly generate tokens for testing and development.

By creating a test API token, you agree to the [Slack API Terms of Service](#).

Workspace	User	Token
Tower	penghon	xoxp-358344330100-3587093



Your Slack workspace have the #general and #random default channels configured together with the channel that you have created to receive notifications from your Ansible Tower.

[Optional] Download the slack app on your mobile phone and log into your newly created slack workspace.

3. Ansible Tower Installation and Configuration

We will be installing Ansible Tower 3.2.3 and will upgrade to 3.2.4 (latest version as of May 2018) as an optional task.

Just a quick word on the perquisites and requirements for a Tower installation.

Ansible Tower minimum requirements as of 3.2.3 are as follows:

- Supported Operating Systems:
 - Red Hat Enterprise Linux 7.2 or later 64-bit
 - CentOS 7.2 or later 64-bit
 - Ubuntu 14.04 LTS 64-bit
 - Ubuntu 16.04 LTS 64-bit
- Currently supported version of Mozilla Firefox or Google Chrome
 - Other HTML5 compliant web browsers may work but are not fully tested or supported.
- 2 CPUs minimum
 - 2 CPUs recommended per 20 forks
- 2 GB RAM minimum (4+ GB RAM recommended)
 - 2 GB RAM (minimum)
 - 4GB RAM per 100 forks
- 20 GB of dedicated hard disk space
 - 10GB of the 20 GB requirement must be dedicated to /var/ where Tower stores its files and working directories
 - Storage volume rated for minimum baseline of 750 IOPS.
- 20 GB of dedicated hard disk space for nodes containing a database (150 GB+ recommended)
 - The storage volume should be rated for a high baseline IOPS (1000 or more.)
- 64-bit support required
- PostgreSQL version 9.6 required for Ansible Tower 3.2 and later
- Ansible Engine version 2.2 required for Ansible Tower 3.2 and later
- For AWS EC2:
 - M3.medium or larger
 - For 100 managed hosts, m3.xlarge or larger is needed

For the purpose of this lab your Ansible Tower will be running on the following specifications:

- CentOS Linux release 7.3.1611 (Core)
- 1 CPU
- 2 GB RAM
- 30 GB of virtualized disk of questionable IOPS
- Single instance of Ansible Tower with integrated installation (PostgreSQL will be installed on the same machine as the Ansible Tower)

For ease of installation and avoidance of permission related issues every task in this lab will be using the root credentials.



3.1 Ansible Tower Installation

Download the Ansible Tower installation/upgrade tool from <http://releases.ansible.com/ansible-tower/setup/>

In order to save time this file has been already downloaded to the `/var/tmp` folder of your designated Ansible Tower controller.

Extract the files and access the `ansible-tower-setup-3.2.3` directory:

```
[root@localhost tmp]# tar xvzf ansible-tower-setup-3.2.3.tar.gz  
[root@localhost tmp]# cd ansible-tower-setup-3.2.3
```

Modify the `/var/tmp/ansible-tower-setup-3.2.3/inventory` file with the following modifications highlighted in red:

```
[tower]  
localhost ansible_connection=local  
  
[database]  
  
[all:vars]  
admin_password='abc123!@'  
  
pg_host=""  
pg_port=""  
  
pg_database='awx'  
pg_username='awx'  
pg_password='abc123!@'  
  
rabbitmq_port=5672  
rabbitmq_vhost=tower  
rabbitmq_username=tower  
rabbitmq_password='abc123!@'  
rabbitmq_cookie=cookiemonster  
  
# Needs to be true for fqdns and ip addresses  
rabbitmq_use_long_name=false  
  
# Isolated Tower nodes automatically generate an RSA key for authentication;  
# To disable this behavior, set this value to false  
# isolated_key_generation=true
```

Once the necessary changes above has been made, run `./setup.sh`:

```
[root@localhost ansible-tower-setup-3.2.3]# ./setup.sh
```

The setup.sh script will install all necessary dependencies such as epel-release and Ansible Engine.

To track the installation process, run the following

```
[root@localhost ~]# tail -f /var/log/messages
```

The following task will require quite a fair bit of time:

```
TASK [awx_install : Migrate the Tower database schema (may take awhile when upgrading).]
```

Once the installation process has completed check that all necessary processes are up and running:

```
[root@localhost ansible-tower-setup-3.2.3]#  
ansible-tower-service status | grep -i running  
Redirecting to /bin/systemctl status postgresql-9.6.service  
  Active: active (running) since Tue 2018-05-08 22:34:22 SGT; 1 day 11h ago  
Redirecting to /bin/systemctl status rabbitmq-server.service  
  Active: active (running) since Tue 2018-05-08 22:34:27 SGT; 1 day 11h ago  
Redirecting to /bin/systemctl status nginx.service  
  Active: active (running) since Tue 2018-05-08 22:34:28 SGT; 1 day 11h ago  
Redirecting to /bin/systemctl status supervisord.service  
  Active: active (running) since Tue 2018-05-08 22:34:28 SGT; 1 day 11h ago
```

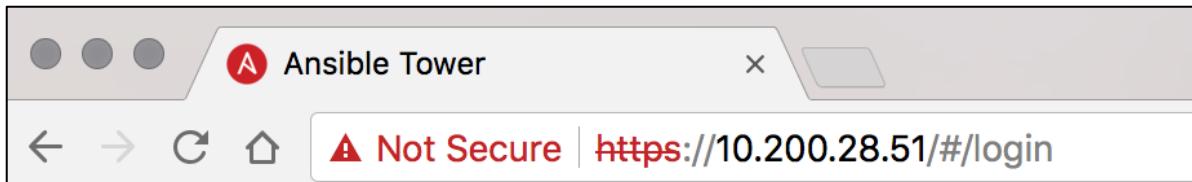
Four essential services must be in running:

- Postgresql
- Rabbitmq-server
- Nginx
- Supervisord

3.2 Ansible Tower Licensing

After your Ansible Tower has been successfully installed, access the Tower GUI by using your browser and key in your Ansible Controller management IP.

For example:



Log into the Tower UI with the admin credentials that was specified in the `/var/tmp/ansible-tower-setup-3.2.3/inventory` file (check the file if you do not remember the admin password).

A copy of an NFR (Not for Resale) Ansible Tower Enterprise License has been made available on your Ansible Tower controller as `/var/tmp/sample_files/ansible_tower_license.txt`.

Copy the contents of the above license file and create a local file. You will need to upload this file later via the Tower UI.

Ansible Tower login page looks like the following:

The screenshot shows the Ansible Tower login page. At the top left is the Ansible Tower logo with the text "ANSIBLE TOWER by Red Hat". Below it is the message "Welcome to Ansible Tower! Please sign in.". There are two input fields: "USERNAME" and "PASSWORD", each with a corresponding text input box. To the right of the "PASSWORD" field is a green "SIGN IN" button.

Key in the username "admin" and the password.

Select the browse button, upload the Ansible Tower License that you have created previously and agree to the EULA.

TOWER LICENSE

Welcome to Ansible Tower! Please complete the steps below to acquire a license.

1 Please click the button below to visit Ansible's website to get a Tower license key.

REQUEST LICENSE

2 Choose your license file, agree to the End User License Agreement, and click submit.

* LICENSE FILE

BROWSE No file selected.

* END USER LICENSE AGREEMENT

ANSIBLE TOWER BY RED HAT END USER LICENSE AGREEMENT

This end user license agreement ("EULA") governs the use of the Ansible Tower software and any related updates, upgrades, versions, appearance, structure and organization (the "Ansible Tower Software"), regardless of the delivery mechanism.

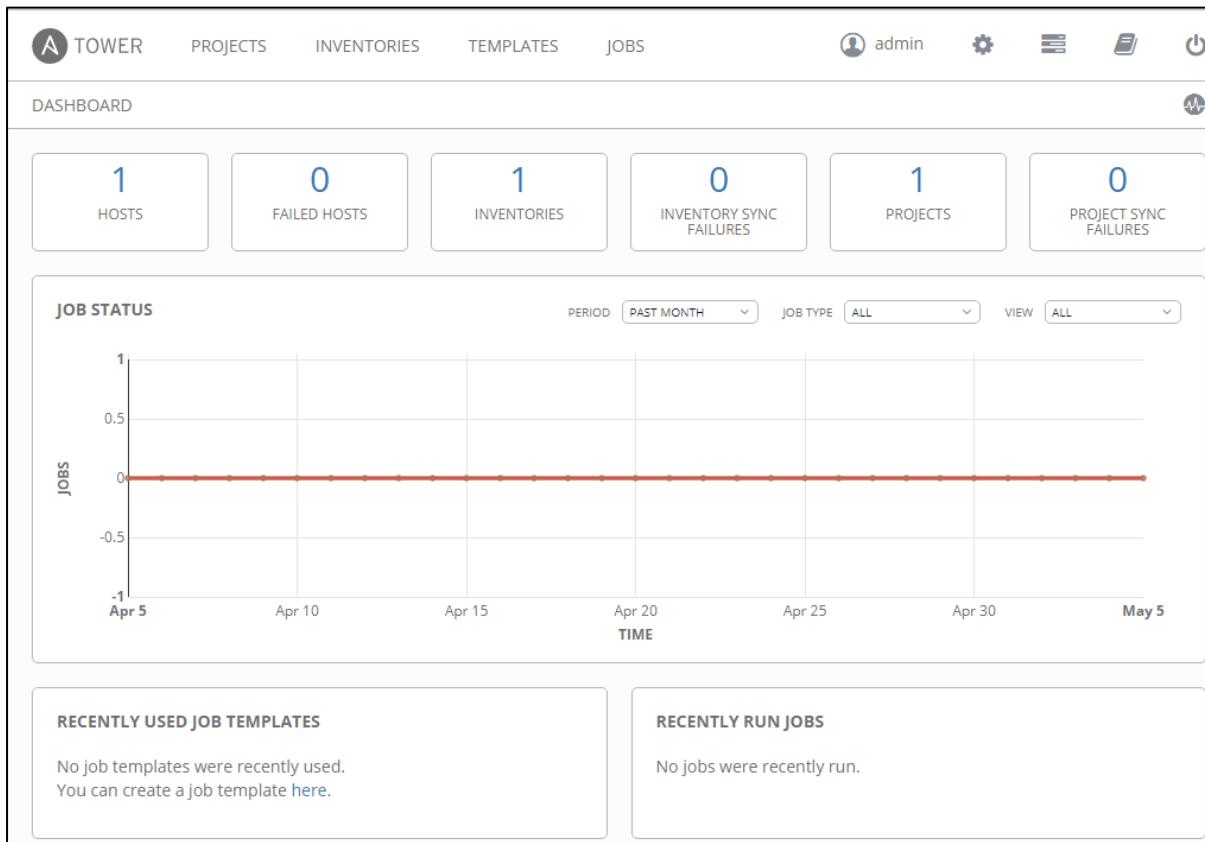
1. License Grant. Subject to the terms of this EULA, Red Hat, Inc. and its affiliates ("Red Hat") grant to you ("You") a non-exclusive, non-transferable, limited right to use the Ansible Tower Software for your internal business purposes.

I agree to the End User License Agreement

SUBMIT

Once the system accepts the license file you will be brought to the Ansible Tower dashboard.

3.3 Ansible Tower Dashboard



Several objects are already been pre-defined (following list is not exhaustive):

- Demo Project
- Demo Inventory
- Default organization
- Demo Credentials
- Demo Job Template

3.4 Exploring the Ansible Tower Dashboard and Interface

Take a minute to examine the Ansible Tower Dashboard and interface.

The screenshot shows the Ansible Tower dashboard with several key components highlighted:

- Top Navigation:** TOWER, PROJECTS, INVENTORIES, TEMPLATES, JOBS. The JOBS tab is active.
- Dashboard Summary:** A row of six boxes showing counts: 60 HOSTS, 0 FAILED HOSTS, 8 INVENTORIES, 0 INVENTORY SYNC FAILURES, 5 PROJECTS, and 0 PROJECT SYNC FAILURES.
- Job Status Graph:** A line chart titled "JOB STATUS" showing the number of jobs over time from April 6 to May 6. The Y-axis is labeled "JOBS" and ranges from 0 to 29. The X-axis is labeled "TIME". The graph shows a sharp peak of 29 jobs around April 21, with other smaller peaks and troughs.
- Recently Used Templates:** A table titled "RECENTLY USED TEMPLATES" with columns: NAME, ACTIVITY, and ACTIONS. The entries are:

NAME	ACTIVITY	ACTIONS
ESX modify VM memory with Survey with Vault credentials	● ● ● ●	🔗 🔎
Modify ESX CentOS VM memory	● ●	🔗 🔎
Shutdown all CentOS VMs on ESX Servers	● ● ●	🔗 🔎
Full ESX VMs restoration	● ●	🔗 🔎
172.28.28.31 vm snapshot restoration	● ● ● ●	🔗 🔎
- Recent Job Runs:** A table titled "RECENT JOB RUNS" with columns: NAME and TIME. The entries are:

NAME	TIME
ESX modify VM memory with Survey with Vault credentials	3/5/2018 15:35:44
ESX modify VM memory with Survey with Vault credentials	2/5/2018 09:49:56
ESX modify VM memory with Survey with Vault credentials	2/5/2018 09:43:43
ESX modify VM memory with Survey with Vault credentials	2/5/2018 09:40:04
Modify ESX CentOS VM memory	30/4/2018 23:29:00

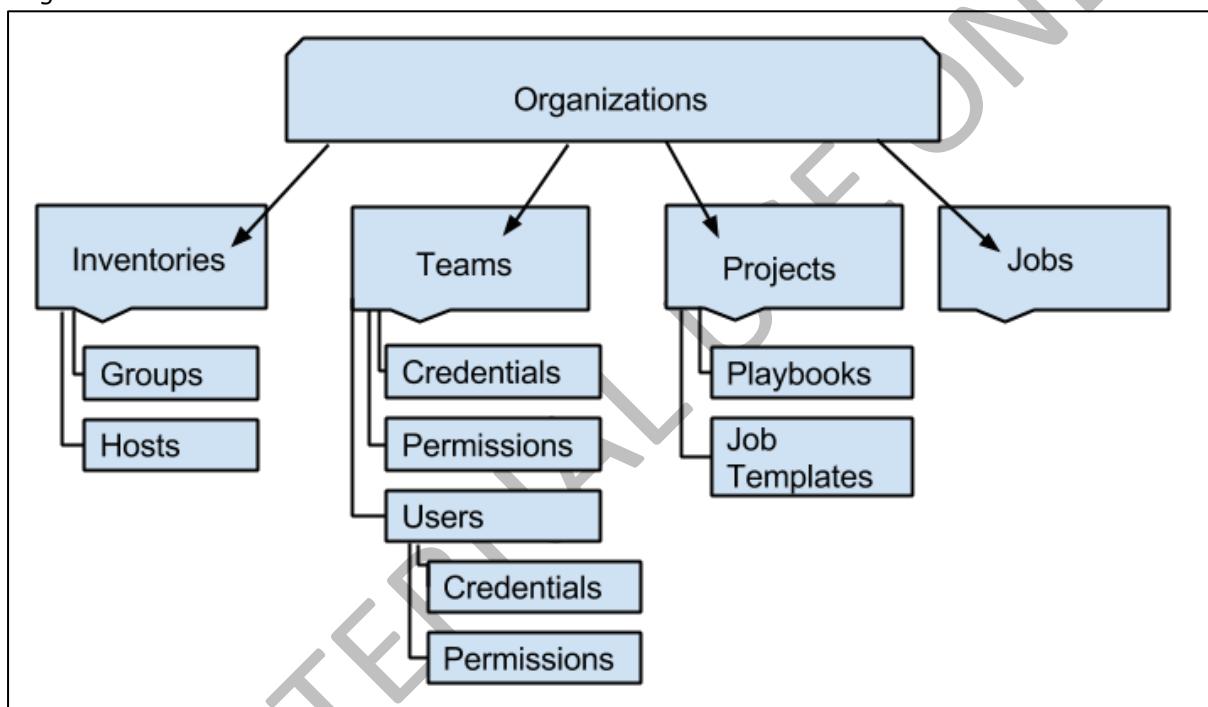
Annotations with arrows point to specific sections:

- An arrow points from the text "Quick access to 4 major Tower object categories" to the top navigation bar.
- An arrow points from the text "Object Summary and visual graphing of Job Status with red meaning failed jobs and green meaning successful jobs runs" to the Job Status graph.
- An arrow points from the text "Quick view of recently used templates" to the Recently Used Templates table.
- An arrow points from the text "Quick view of recent job runs" to the Recent Job Runs table.

3.5 Explanation of Dashboard and interface objects

Quick Access Menu	Purpose and Functionality
Projects	A Project is a logical collection of Ansible playbooks, represented in Tower.
Inventories	An Inventory is a collection of hosts against which jobs may be launched, the same as an Ansible inventory file. Inventories are divided into groups and these groups contain the actual hosts.
Templates	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. A job is an execution of a job template.
Jobs	This shows the list of jobs that have run in the past.

Ansible Tower with Enterprise license enables multitenancy through the use of different *Organizations*.



Organization is the highest level in the Tower object hierarchy as shown above.

3.6 Ansible Tower Settings Menu

The Settings menu allows you to create your organizations, add credentials, add users and teams, schedule management jobs, modify your Tower's configuration, and more.

ORGANIZATIONS Group all of your content to manage permissions across departments in your company.	USERS Allow others to sign into Tower and own the content they create.	TEAMS Split up your organization to associate content and control permissions for groups.
CREDENTIALS Add passwords, SSH keys, and other credentials to use when launching jobs against machines, or when syncing inventories or projects.	CREDENTIAL TYPES Create custom credential types to be used for authenticating to network hosts and cloud sources	MANAGEMENT JOBS Manage the cleanup of old job history, activity streams, data marked for deletion, and system tracking info.
INVENTORY SCRIPTS Create and edit scripts to dynamically load hosts from any source.	NOTIFICATIONS Create templates for sending notifications with Email, HipChat, Slack, and SMS.	INSTANCE GROUPS View list and capacity of Tower instances.
CONFIGURE TOWER Edit Tower's configuration.	ABOUT TOWER View information about this version of Ansible Tower.	VIEW YOUR LICENSE View and edit your license information.

Settings	Purpose and Functionality
Organizations	An Organization is a logical collection of Users, Teams, Projects, and Inventories, and is the highest level in the Tower object hierarchy. This Organizations displays all of the existing organizations for your installation of Tower and will allow you to manage new organizations to your Ansible Tower.
Users	A User is someone who has access to Tower with associated permissions and credentials. This Users link allows you to manage all Tower users.
Teams	A Team is a subdivision of an organization with associated users, projects, credentials, and permissions. Teams provide a means to implement role-based access control schemes and delegate responsibilities across organizations. . This Teams link allows you to manage Teams for Tower.
Credentials	Credentials are utilized by Tower for authentication when launching Jobs against machines, synchronizing with inventory sources, and importing project content from a version control system. You can add credential objects from the following predefined credential types: <ul style="list-style-type: none"> • Amazon Web Services • Ansible Tower • Google Compute Engine • Insights • Machine • Microsoft Azure Resource Manager • Network • OpenStack

	<ul style="list-style-type: none"> • Red Hat CloudForms • Red Hat Satellite 6 • Red Hat Virtualization • Source Control • Vault • VMware vCenter
Credential Types	Creation and management of custom credential type in standard format using a YAML/JSON-like definition.
Management Jobs	Management Jobs assist in the cleaning of old data from Tower, including system tracking information, job histories, and activity streams. You can use this if you have specific retention policies or need to decrease the storage used by your Tower database.
Inventory Scripts	Tower includes built-in support for syncing dynamic inventory from cloud sources such as Amazon AWS, Google Compute Engine, among others. Tower also offers the ability to use a custom script to pull from your own inventory source.
Notifications	Creation and management of notification. You can configure Ansible Tower to send notification of job success or failure to the following predefined notification types: <ul style="list-style-type: none"> • Email • Slack • Twilio • PagerDuty • HipChat • Webhook • IRC
Instance Groups	Manage Tower cluster instance groups.
Configure Tower	In Ansible Tower version 3.1.0, you can configure various Tower settings within the Tower user interface, in the following tabs: Authentication: Enable simplified login for your Tower applications. Jobs: Update settings pertaining to Jobs within Tower. System: Define system-level features and functions. User Interface: Set the level of data collection for use in usability analytics; and configure custom logos and login messages.
About Tower	A funky picture that shows the version of the Ansible Tower.
View Your License	Management of your Ansible Tower license.

3.7 Ansible Tower Organization Creation

An Ansible Tower Organization is a logical collection of Users, Teams, Projects, and Inventories, and is the highest level in the Tower object hierarchy. Tower creates a default organization automatically and it is the only organization for non-Enterprise Tower installations.

Since we are using Enterprise license for this lab, the first object that we will be creating will be your own Tower organization.

Using the admin account, click on Settings and select the Organizations link.

The screenshot shows the Ansible Tower dashboard. At the top, there are navigation links: TOWER, PROJECTS, INVENTORIES, TEMPLATES, JOBS, and a Settings icon. A large green arrow points to the Settings icon, with the text "Click on Settings" overlaid. Below the navigation, there's a section titled "DASHBOARD" with various metrics: 1 HOSTS, 0 FAILED HOSTS, 1 INVENTORIES, 0 INVENTORY SYNC FAILURES, 1 PROJECTS, and 0 PROJECT SYNC FAILURES. Underneath is a "JOB STATUS" chart showing zero jobs over time from April 5 to May 5. At the bottom, there are two boxes: "RECENTLY USED JOB TEMPLATES" (empty) and "RECENTLY RUN JOBS" (empty).

The screenshot shows the "SETTINGS" page. It has several sections: "ORGANIZATIONS" (highlighted with a green box and a large green arrow pointing to it with the text "Click on the Organization link"), "TEAMS", "CREDENTIALS", "CREDENTIAL TYPES", and "MANAGEMENT JOBS". The "admin" user is logged in at the top right.

Select **+ADD** to create a new organization:

The screenshot shows the 'ORGANIZATIONS' page in Ansible Tower. It lists two organizations: 'Default' and 'PH Tower of Babel'. Each organization has a summary card with counts for USERS, TEAMS, INVENTORIES, PROJECTS, and JOB TEMPLATES. A large green arrow points to the '+ADD' button in the top right corner, which is used to create a new organization.

Fill the *Organization* name, and optionally its description, ignoring instance groups and click on **SAVE**.

The screenshot shows the 'NEW ORGANIZATION' dialog box. It has tabs for DETAILS, USERS, and NOTIFICATIONS. Under DETAILS, there are fields for NAME ('Penghon's Awesome Organization') and DESCRIPTION ('My first Tower Organization'). There is also a field for INSTANCE GROUPS which is currently empty. A large green arrow points to the 'SAVE' button. Below the dialog, the 'ORGANIZATIONS' page is shown again with the newly created 'Default' organization listed.

Tower Instance groups are instances of Ansible Tower that are grouped according to Organizations, Inventories or Job Templates for job execution load balancing. We will not be covering instance groups in this lab.

After saving the configuration, the newly created organization should look similar to the following:

The screenshot shows two main sections of the F5 Tower interface. The top section is a modal dialog titled "Penghon's Awesome Organization" with tabs for DETAILS, USERS, and NOTIFICATIONS. It contains fields for NAME (Penghon's Awesome Organization), DESCRIPTION (My first Tower Organization), and INSTANCE GROUPS. A green arrow points from this modal down to the bottom section. The bottom section is a list of organizations under the heading "ORGANIZATIONS 2". It includes a search bar, a key button, and a "+ ADD" button. Two organizations are listed: "Default" and "Penghon's Awesome Organization". The "Penghon's Awesome Organization" entry is highlighted with a green border. Both entries show counts for USERS, TEAMS, INVENTORIES, PROJECTS, JOB TEMPLATES, and ADMINS. The bottom right of the screen shows "ITEMS 1 - 2".

Penghon's Awesome Organization

DETAILS USERS NOTIFICATIONS

* NAME: Penghon's Awesome Organization

DESCRIPTION: My first Tower Organization

INSTANCE GROUPS

CANCEL SAVE

ORGANIZATIONS 2

SEARCH KEY + ADD

Default

0 USERS 0 TEAMS
1 INVENTORIES 1 PROJECTS
1 JOB TEMPLATES 0 ADMINS

Penghon's Awesome Organization

0 USERS 0 TEAMS
0 INVENTORIES 0 PROJECTS
0 JOB TEMPLATES 0 ADMINS

ITEMS 1 - 2

3.8 Creating an Ansible Tower Operator User Account

Select the *USERS* from the Settings menu:

The screenshot shows the Ansible Tower Settings page. At the top right, there is a green box labeled "Settings" with a gear icon. A large green arrow points to the "USERS" section, which is also highlighted with a green box. The "USERS" section contains the text: "Allow others to sign into Tower and own the content they create." Below the main content area, there are several other sections: ORGANIZATIONS, CREDENTIALS, INVENTORY SCRIPTS, CONFIGURE TOWER, USERS (which is the current section), CREDENTIAL TYPES, NOTIFICATIONS, ABOUT TOWER, MANAGEMENT JOBS, INSTANCE GROUPS, and VIEW YOUR LICENSE.

Click **+ADD** to add a new user:

The screenshot shows the Ansible Tower USERS list page. At the top right, there is a green button labeled "+ADD". The page displays a table with columns: USERNAME, FIRST NAME, LAST NAME, and ACTIONS. There is one entry in the table: "admin".

Fill up the various fields from the new user configuration form and click on **SAVE**. Remember the username and password you have created for this user as you will be logging in as this user later during the lab.

NEW USER

DETAILS **ORGANIZATIONS** **TEAMS** **PERMISSIONS**

* FIRST NAME Operator	* LAST NAME One	* ORGANIZATION <input type="text" value="Penghon's Awesome Organization"/>
* EMAIL operator1@penghon.local	* USERNAME op1	* PASSWORD <input type="password" value="*****"/> <input type="button" value="SHOW"/>
* CONFIRM PASSWORD <input type="password" value="*****"/> <input type="button" value="SHOW"/>	USER TYPE <input type="button" value="Normal User"/>	
<input type="button" value="CANCEL"/> <input type="button" value="SAVE"/>		

USERS **1**

SEARCH	KEY	+ ADD
admin		<input type="button" value=""/>

ITEMS 1 - 1

3.9 Creating an Ansible Tower Team

Teams provides a means to implement role-based access control schemes and delegate responsibilities across organizations. Common credentials and access control can be applied to a group of users rather than individuals.

Click on the TEAMS link from the Settings Menu:

The screenshot shows the Ansible Tower Settings menu. At the top right, there is a green speech bubble with the word "Settings". Below it is a navigation bar with links for TOWER, PROJECTS, INVENTORIES, TEMPLATES, JOBS, and a user icon labeled "admin". To the right of the user icon is a gear icon, which is highlighted with a red box. A large green arrow points from the text "Click on the TEAMS link" to the "TEAMS" section. The "TEAMS" section is also highlighted with a red box. Other sections visible include ORGANIZATIONS, CREDENTIALS, CREDENTIAL TYPES, MANAGEMENT JOBS, INVENTORY SCRIPTS, NOTIFICATIONS, INSTANCE GROUPS, CONFIGURE TOWER, ABOUT TOWER, and VIEW YOUR LICENSE.

Select **+ADD** to create the Team:

The screenshot shows the Ansible Tower Teams list page. At the top right is a green button labeled "+ ADD". Below it is a table with columns for NAME, ORGANIZATION, and ACTIONS. The first row shows a team named "BIGIP Operators" associated with "Penghon's Awesome Organization".

Fill in the *name* of your Team and select the *Organization* and click on **SAVE**:

The screenshot shows the "NEW TEAM" dialog. It has tabs for DETAILS, USERS, and PERMISSIONS. The DETAILS tab is selected. It contains fields for NAME (filled with "BIGIP Operators"), DESCRIPTION (filled with "Team of Operators"), and ORGANIZATION (a dropdown menu showing "Penghon's Awesome Organization"). At the bottom right are "CANCEL" and "SAVE" buttons.

After adding the Team, it should be listed in the Team list:

The screenshot shows a list of teams. There is one item listed: 'BIGIP Operators'. The organization is 'Penghon's Awesome Organization'. The interface includes a search bar, a key field, and a green '+ ADD' button.

Add the previously created Operator account as **member** of the Team:

The screenshot shows a modal dialog titled 'BIGIP OPERATORS | ADD USERS'. It lists users: 'admin' (unchecked) and 'op1' (checked). The role is 'Operator' and the count is 'One'. Below, it says 'Please assign roles to the selected users/teams' and shows 'Operator One' assigned to 'Member'. A green arrow points to the 'op1' checkbox, another points to the 'Member' selection, and a third points to the 'SAVE' button.

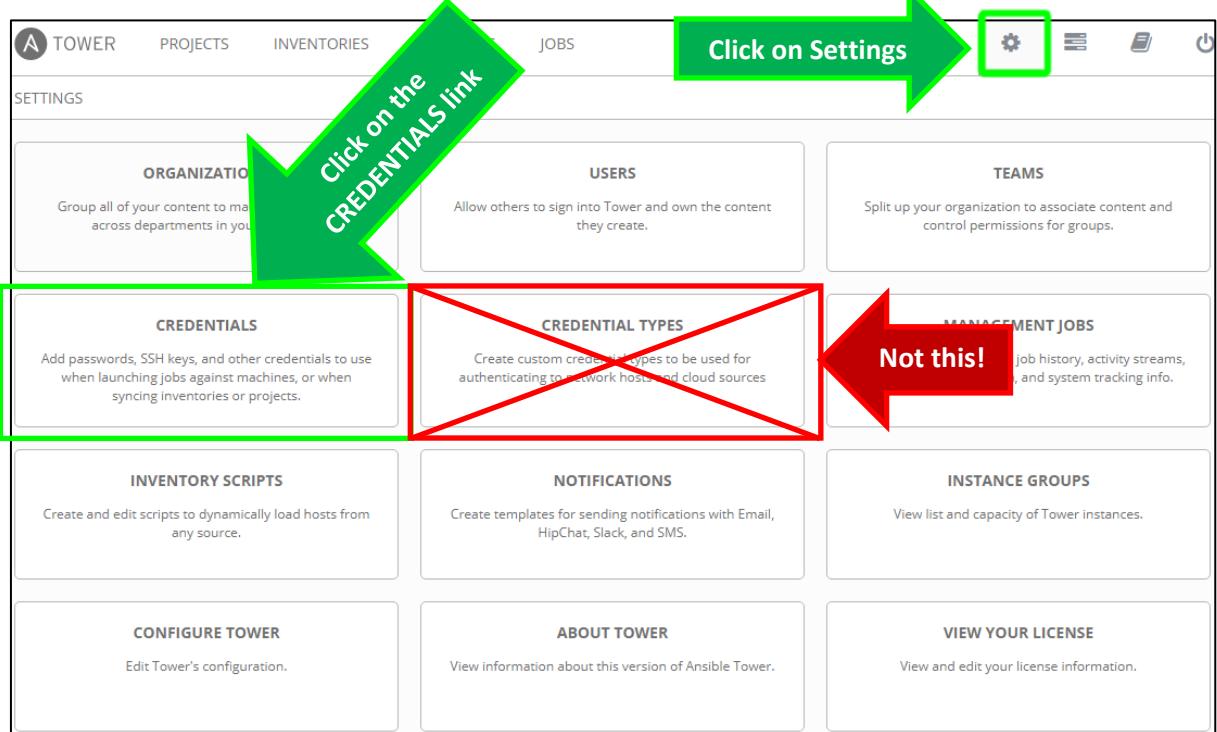
The latest member of the Team should appear in the User list as shown below:

The screenshot shows the 'BIGIP Operators' user list. It lists two users: 'admin' and 'op1'. Under 'ROLE', 'op1' is assigned 'SYSTEM ADMINISTRATOR' and 'MEMBER'. A green arrow points to the 'MEMBER' role entry.

We will be adding permissions for the Team to the various objects such as Projects, Job Templates and Workflow Templates later in the lab.

3.10 Ansible Tower Credential Creation for Source Control Management (Github)
 Github will be used as the source of truth for the Project which you will create later.

Select the Credentials link from the Settings button:



Select **+ADD** and fill in the various fields. Note that you will be storing the **SSH private key** from the keypair that you had previously generated, your **Github username** and **GitHub password** on your Ansible Tower:

The screenshot shows the 'NEW CREDENTIAL' dialog box. The 'DETAILS' tab is selected. In the 'NAME' field, 'Penghon The engineer's github account' is entered. In the 'DESCRIPTION' field, 'My github credentials' is entered. In the 'ORGANIZATION' field, 'Penghon's Awesome Organization' is selected. Under 'CREDENTIAL TYPE', 'Source Control' is chosen. In the 'TYPE DETAILS' section, the 'USERNAME' is 'penghonenginner' and the 'PASSWORD' is masked. In the 'SCM PRIVATE KEY' section, a long RSA private key is pasted into the text area. At the bottom, there is a 'PRIVATE KEY PASSPHRASE' field with a 'SHOW' button, and buttons for 'CANCEL' and 'SAVE'.

Click on **SAVE** once the various fields have been filled up:

Penghon The Engineer's github account

DETAILS **PERMISSIONS**

* NAME ?
Penghon The Engineer's github account

DESCRIPTION ?
My github credentials

ORGANIZATION
Penghon's Awesome Organization

* CREDENTIAL TYPE ?
Source Control

TYPE DETAILS

USERNAME
penghonenginner

PASSWORD
REPLACE ENCRYPTED

SCM PRIVATE KEY

REPLACE ENCRYPTED

PRIVATE KEY PASSPHRASE
SHOW

CANCEL **SAVE**

The screenshot shows a modal dialog titled 'Penghon The Engineer's github account'. It has two tabs at the top: 'DETAILS' (selected) and 'PERMISSIONS'. Under 'DETAILS', there are fields for 'NAME' (filled with 'Penghon The Engineer's github account'), 'DESCRIPTION' (filled with 'My github credentials'), and 'ORGANIZATION' (filled with 'Penghon's Awesome Organization'). A section for 'CREDENTIAL TYPE' is shown with 'Source Control' selected. 'TYPE DETAILS' includes 'USERNAME' ('penghonenginner') and 'PASSWORD' ('REPLACE ENCRYPTED'). A large gray box labeled 'SCM PRIVATE KEY' contains 'REPLACE' and 'ENCRYPTED' buttons. Below that is a 'PRIVATE KEY PASSPHRASE' field with a 'SHOW' button. At the bottom right are 'CANCEL' and 'SAVE' buttons, with 'SAVE' being highlighted.

Notice that the Private Key and password field are encrypted immediately upon saving. Ansible Tower utilizes an encryption/decryption algorithm that uses a variation of Fernet: a symmetric encryption cipher utilizing AES-256 in CBC mode alongside a SHA-256 HMAC.

3.11 Ansible Tower Project Creation

Project is a logical collection of Ansible playbooks, represented in Tower. Playbooks and playbook directories can either be source controlled with Git, Subversion, Mercurial or Red Hat insights or they can be manually added to Ansible Tower.

The dashboard shows the following metrics:

- HOSTS: 1
- FAILED HOSTS: 0
- INVENTORIES: 1
- INVENTORY SYNC FAILURES: 0
- PROJECTS: 1
- PROJECT SYNC FAILURES: 0

JOB STATUS chart (Past Month): A line graph showing zero jobs running from April 5 to May 5.

RECENTLY USED JOB TEMPLATES: No job templates were recently used. You can create a job template [here](#).

RECENTLY RUN JOBS: No jobs were recently run.

A green arrow points to the "PROJECTS" link in the top navigation bar, with the text "Click on the PROJECTS link".

Select **+ADD** and fill in the various fields as follows (remember to substitute with your own URL in the field within the red box):

NEW PROJECT

DETAILS PERMISSIONS NOTIFICATIONS

* NAME: Penghon's New Ansible Tower Project

DESCRIPTION: My 1st Ansible Tower Project

* ORGANIZATION: Penghon's Awesome Organization

* SCM TYPE: Git

* SCM URL: https://github.com/penghonengineer/ansible-test (highlighted with a red box)

SCM BRANCH/TAG/COMMIT: master

SCM CREDENTIAL: Penghon The Engineer's github account

SCM UPDATE OPTIONS:

- Clean
- Delete on Update
- Update on Launch

CACHE TIMEOUT (SECONDS): 0

CANCEL SAVE

A red arrow points to the "SCM URL" field. A green arrow points to the "Update on Launch" checkbox with the text "Each time a job runs using this Project, Ansible Tower will initiate an update with SCM".

After saving the Project, Ansible Tower will sync and download the Playbooks and Inventory files from the SCM system (in this case Github). **Click** on the pulsating green dot to display the live updates of the initial sync between Ansible Tower and Github:

The screenshot shows the Ansible Tower interface. The top panel, 'PROJECTS', lists two projects: 'Demo Project' and 'Penghon's New Ansible Tower Project'. The bottom panel, 'RESULTS', displays the details for 'Penghon's New Ansible Tower Project'. The 'STATUS' field is highlighted with a green box and shows 'Successful'. The 'STANDARD OUT' section contains command-line output related to the sync process.

NAME	Penghon's New Ansible Tower Project
STATUS	Successful
STARTED	7/5/2018 13:51:41
FINISHED	7/5/2018 13:51:58
ELAPSED	17.031 seconds
LAUNCH TYPE	Manual
PROJECT	Penghon's New Ansible Tower Project
CREDENTIAL	Penghon The Engineer's github account

STANDARD OUT	
<pre>Identity added: /tmp/awx_2_31_5G5/credential_2 (/tmp/awx_2_31_5G5/credential_2) Using /etc/ansible/ansible.cfg as config file [DEPRECATION WARNING]: DEFAULT_ASK_SUDO_PASS option, In favor of Ansible Become, which is a generic framework. See become_ask_pass. , use become instead. This feature will be removed in version 2.8. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg. PLAY [all] **** TASK [delete project directory before update] **** skipping: [localhost]</pre>	

If the status shows successful, you have successfully validated and cloned the files from Github.

[Optional] To further validate whether Ansible Tower has successfully synced with Github, access your Ansible Tower Controller via SSH:

```
root@localhost ~]#
cd /var/lib/awx/projects/_6__penghons_new_ansible_tower_project/
[root@localhost _6__penghons_new_ansible_tower_project]# ls -la
```

3.12 [Optional] Adding a Project manually

In Ansible Tower the default path: `/var/lib/awx/projects` is the “Project Base Path”. As per the online help:

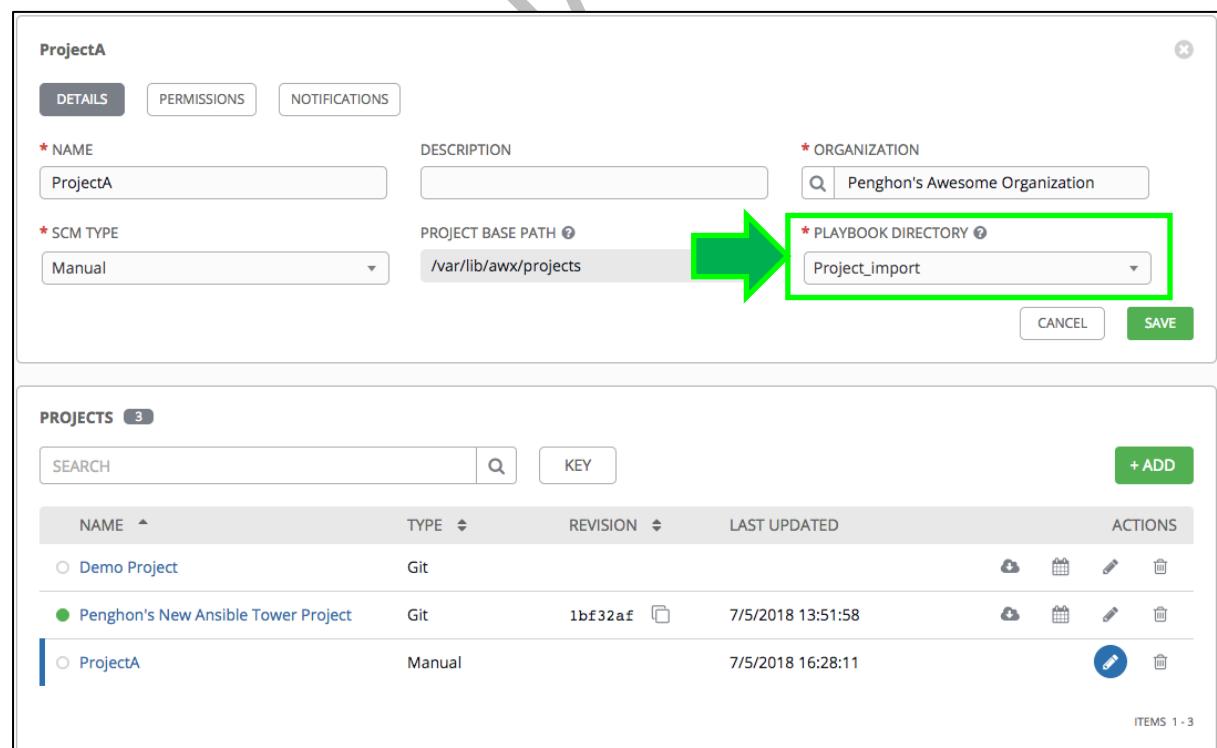
“Base path used for locating playbooks. Directories found inside this path will be listed in the playbook directory drop-down. Together the base path and selected playbook directory provide the full path used to locate playbooks.
Change `PROJECTS_ROOT` under “Configure Tower” to change this location.”

Directories, under the **`PROJECTS_ROOT`** path, are considered projects. YAML files within the directories are considered as playbooks or inventories host files.

To manually import your playbooks and/or inventory files, create a directory under the default `/var/lib/awx/projects` folder. Copy over the sample BIGIP playbook `/var/tmp/sample_files/change_bigip_hostname.yaml` to your newly created directory. For example:

```
root@localhost projects]#  
mkdir /var/lib/awx/projects/Project_import  
root@localhost projects]# cp /var/tmp/sample_files/change_bigip_hostname.yaml /var/lib/  
/awx/projects/Project_import/.  
root@localhost projects]# chown awx.awx -R /var/lib/awx/projects/Project_import/Project  
_import/
```

The Ansible Tower Project name does not have to be the same as the directory under the **`PROJECTS_ROOT`** folder. From Ansible Tower create a new Project and select the appropriate playbook directory:



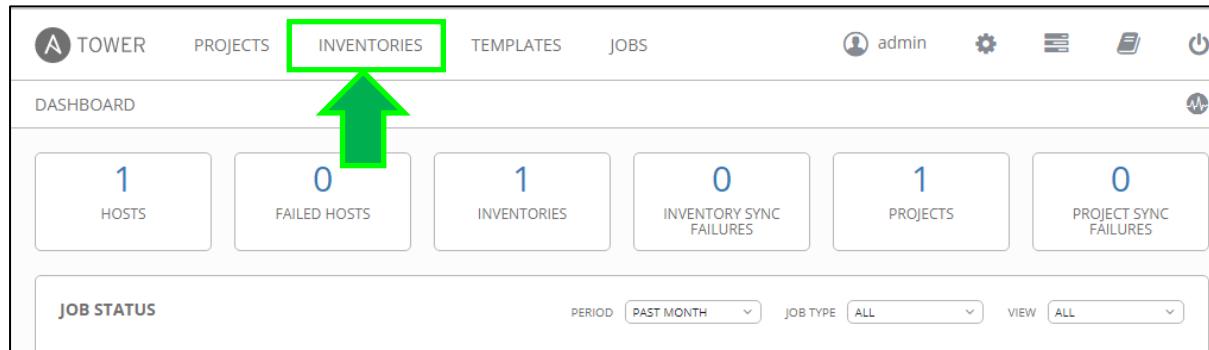
The screenshot shows the 'ProjectA' creation dialog and the 'PROJECTS' list. In the dialog, the 'PLAYBOOK DIRECTORY' dropdown is highlighted with a green arrow and a green border, showing 'Project_import'. The 'PROJECTS' list below shows three projects: 'Demo Project' (Git), 'Penghon's New Ansible Tower Project' (Git), and 'ProjectA' (Manual). The 'ProjectA' row has a blue edit icon in the 'ACTIONS' column.

NAME	TYPE	REVISION	LAST UPDATED	ACTIONS
Demo Project	Git		7/5/2018 13:51:58	
Penghon's New Ansible Tower Project	Git	1bf32af	7/5/2018 13:51:58	
ProjectA	Manual		7/5/2018 16:28:11	

Once the new Project has been created, you will be able to use the inventory host files or the playbooks to create Job Templates.

3.13 Creation of an Ansible Tower Inventory

An Ansible Tower Inventory is similar in concept as an Ansible Engine inventory file. Select the INVENTORIES link as per below:



Click on **+ADD**, provide a meaningful name for the inventory and associate to the right organization.
Click on **SAVE**:

The screenshot shows the 'CREATE INVENTORY' form. It has tabs for DETAILS, PERMISSIONS, GROUPS, HOSTS, SOURCES, and COMPLETED JOBS. The DETAILS tab is selected. The NAME field contains 'BIGIP'. The DESCRIPTION and INSTANCE GROUPS fields are empty. The ORGANIZATION field shows 'Penghon's Awesome Organization'. Below these are INSIGHTS CREDENTIAL and VARIABLE sections. At the bottom are CANCEL and SAVE buttons.

Your inventory should look similar to the following:

NAME	TYPE	ORGANIZATION	ACTIONS
BIGIP	Inventory	Penghon's Awesome Organization	
Demo Inventory	Inventory	Default	

There are 5 types of permissions for Inventory:

- **Admin** allows read, run, and edit privileges (applies to all resources)
- **Use** allows use of a resource in a job template (applies all resources except job templates)
- **Update** allows updating of project via the SCM Update (applies to projects and inventories)
- **Ad Hoc** allows use of Ad Hoc commands (applies to inventories)
- **Execute** allows launching of a job template (applies to job templates)

For the purpose of this Lab, the only permissions that matter would be applied to the Job Template, which is to allow push button permission to execute the Job Template. We will be covering this in the later section of the lab.

From your Ansible Tower, from the Inventory that you had created, select SOURCES and select the **+ADD SOURCE**:

The screenshot shows the Ansible Tower interface. At the top, there are navigation links: TOWER, PROJECTS, INVENTORIES, TEMPLATES, and JOBS. On the right side, there are user and system icons. Below the navigation, the path 'INVENTORIES / BIGIP / SOURCES' is displayed. A modal window titled 'BIGIP' is open, showing tabs for DETAILS, PERMISSIONS, GROUPS, HOSTS, SOURCES (which is selected), and COMPLETED JOBS. A green button labeled '+ ADD SOURCE' is visible. The main area of the modal has a placeholder text 'PLEASE ADD ITEMS TO THIS LIST'.

Configure source of the inventory file as “Source from a Project”, and select the Project that was created previously. The inventory file which was synced from Github previously will appear for selection:

The screenshot shows the 'CREATE SOURCE' dialog. At the top, it says 'CREATE SOURCE' and has a 'DETAILS' tab selected. Below it, there are fields for 'NAME' (set to 'BIGIP hosts and groups'), 'DESCRIPTION' (empty), and 'SOURCE'. The 'SOURCE' field is set to 'Sourced from a Project'. A large green arrow points down to the 'PROJECT' field, which contains 'Penghon's New Ansible Tower Project'. Another green arrow points from the 'PROJECT' field towards the 'SOURCE' field. To the right of the 'PROJECT' field, there is a 'CREDENTIAL' dropdown and an 'INVENTORY FILE' dropdown set to 'inventory'. Under 'VERBOSITY', a dropdown is set to '1 (INFO)'. In the 'UPDATE OPTIONS' section, 'Overwrite Variables' is checked. At the bottom, there are 'ENVIRONMENT VARIABLES' sections for YAML and JSON, and 'CANCEL' and 'SAVE' buttons.

Click on **SAVE**, and click on the sync symbol as shown:

Screenshot of the Ansible Tower interface showing the 'SOURCES' tab for a project named 'BIGIP'. The source listed is 'BIGIP hosts and groups', described as 'Sourced from a Project'. In the top right corner, there is a 'SYNC ALL' button with a green arrow pointing to it. Other buttons include '+ ADD SOURCE' and various actions like edit and delete.

Click on the pulsing cloud as shown below to see the result of the job:

Screenshot of the Ansible Tower interface showing the 'SOURCES' tab for a project named 'BIGIP'. The source listed is 'BIGIP hosts and groups', described as 'Sourced from a Project'. A green arrow points to the cloud icon next to the source name. Other buttons include 'SYNC ALL' and '+ ADD SOURCE'.

Examine the result of the inventory sync. Once successful, Ansible Tower will import the hosts from the source into your newly created Tower inventory:

Screenshot of the Ansible Tower interface showing the results of a sync job. The 'RESULTS' pane on the left displays details about the sync job, including the source name 'BIGIP hosts and groups', status 'Successful', and log information. The 'STANDARD OUT' pane on the right shows the command-line output of the sync process, which includes logs such as 'Updating inventory 2: BIGIP' and 'Inventory import completed for BIGIP hosts and groups in 2.9s'.

NAME	BIGIP hosts and groups
STATUS	Successful
EXPLANATION	
LICENSE ERROR	False
STARTED	7/5/2018 23:09:20
FINISHED	7/5/2018 23:09:48
ELAPSED	27.867 seconds
LAUNCH TYPE	Manual
SOURCE	Sourced from a Project
OVERWRITE	False
OVERWRITE VARS	True

```
8.643 INFO Updating inventory 2: BIGIP
8.800 INFO Reading Ansible inventory source: /var/lib/awx/projects/_6__penghons_ne
11.140 INFO Processing JSON output...
11.141 INFO Loaded 1 groups, 1 hosts
11.171 INFO Inventory variables unmodified
11.219 INFO Group "BIGIP" added
11.297 INFO Host "bigip11" added
11.326 INFO Host "bigip11" added to group "BIGIP"
11.513 INFO Inventory import completed for BIGIP hosts and groups in 2.9s
```

Validate that both group and host have been imported successfully:

BIGIP

DETAILS PERMISSIONS GROUPS HOSTS SOURCES COMPLETED JOBS

SEARCH + ADD GROUP

ITEMS 1 - 1

BIGIP

DETAILS PERMISSIONS GROUPS HOSTS SOURCES COMPLETED JOBS

SEARCH + ADD HOST

ITEMS 1 - 1

Click on the host link for further details. Notice that SCM controlled hosts cannot be turned off via Ansible Tower. Why?

bigip11

DETAILS FACTS GROUPS

* HOST NAME DESCRIPTION

VARIABLES

```
1 ansible_host: 172.28.28.110
```

CANCEL SAVE

3.14 [Optional] Using awx-manage to manually import inventory file

For the purpose of this lab task you will be using awx-manage to manually import the inventory file to your inventory hosts.

Create the inventory on Ansible Tower:

INVENTORIES / CREATE INVENTORY

NEW INVENTORY

DETAILS PERMISSIONS GROUPS HOSTS SOURCES COMPLETED JOBS

* NAME: Manually_Added_Inventory

DESCRIPTION

* ORGANIZATION: Penghon's Awesome Organization

INSIGHTS CREDENTIAL

INSTANCE GROUPS

VARIABLES YAML JSON

1 ---

CANCEL SAVE

Result:

NAME	TYPE	ORGANIZATION	ACTIONS
BIGIP	Inventory	Penghon's Awesome Organization	
Demo Inventory	Inventory	Default	
Manually_Added_Inventory	Inventory	Penghon's Awesome Organization	

Copy the sample inventory file `/var/tmp/sample_files/inventory` into the previously created project folder `/var/lib/awx/projects/Project_import/`:

```
[root@localhost Project_import]# awx-manage inventory_import --source /var/lib/awx/projects/Project_import/inventory --inventory-name Manually_Added_Inventory
3.916 INFO Updating inventory 3: Manually_Added_Inventory
4.055 INFO Reading Ansible inventory source: /var/lib/awx/projects/Project_import/inventory
5.640 INFO Processing JSON output...
5.641 INFO Loaded 1 groups, 1 hosts
5.649 INFO Inventory variables unmodified
5.670 INFO Group "BIGIP" added
5.688 INFO Host "bigip11" added
```

```
5.706 INFO Host "bigip11" added to group "BIGIP"
5.824 INFO Inventory import completed for (Manually_Added_Inventory - 9) in 1.9s
```

Check that the group and host are added into the **Manually_Added_Inventory** inventory

The screenshot shows the Ansible Tower interface. At the top, there are tabs for TOWER, PROJECTS, INVENTORIES, TEMPLATES, and JOBS. On the right, there are user and settings icons. Below the tabs, it says 'INVENTORIES / Manually_Added_Inventory / GROUPS'. The main area is titled 'Manually_Added_Inventory' and has tabs for DETAILS, PERMISSIONS, GROUPS (which is selected), HOSTS, SOURCES, and COMPLETED JOBS. There is a search bar and a 'RUN COMMANDS' button. A green button labeled '+ ADD GROUP' is visible. Below the tabs, there is a 'GROUPS' dropdown menu with 'BIGIP' selected. To the right, there is an 'ACTIONS' section with edit and delete icons. At the bottom right, it says 'ITEMS 1 - 1'.

Notice that a manually added host can be switched off and cannot be synced:

The screenshot shows the Ansible Tower interface. At the top, there are tabs for DETAILS, PERMISSIONS, GROUPS, HOSTS (selected), SOURCES, and COMPLETED JOBS. On the right, there are 'RUN COMMANDS' and '+ ADD HOST' buttons. Below the tabs, there is a 'HOSTS' dropdown menu with 'ON' selected. A green box highlights the 'ON' switch for the first host, and a green arrow points to it. To the right, there is an 'ACTIONS' section with edit and delete icons. At the bottom right, it says 'ITEMS 1 - 1'. Below this, there is another section with tabs for INVENTORIES and HOSTS, a search bar, and a '+ ADD' button. The host list table has columns for NAME, TYPE, ORGANIZATION, and ACTIONS. It lists three hosts: 'BIGIP' (Inventory, Penghon's Awesome Organization), 'Inventory' (Inventory, Default), and 'Manually_Added_Inventory' (Inventory, Penghon's Awesome Organization). Each host has an edit and delete icon in the ACTIONS column. At the bottom right of this section, it says 'ITEMS 1 - 3'. A large watermark 'F5' is diagonally across the page.

3.15 Modification of playbook to use Ansible Vault

We will be using the Ansible Vault `encrypt_string` to create a single encrypted variable which is the password for the admin user on BIGIP.

SSH to your Ansible Tower Controller and run the following:

```
[root@localhost ~]# echo -n 'admin' | ansible-vault encrypt_string
New Vault password:
Confirm New Vault password:
Reading plaintext input from stdin. (ctrl-d to end input)
!vault |
$ANSIBLE_VAULT;1.1;AES256      39666432363636313832363161336439663
832376666313134623934366366393739666433356561
6636366136346138663838373034313366393336646237380a35323533306365656562
3838353632      6663643434663835396630303136316639313531666237646430316
4316434663432376630343839      3362303131333634650a39643934336133331383
8313136656561663366356539316538313662376333
Encryption successful
```

Remember the vault secret that was used to encrypt the BIGIP admin password and copy out the highlighted lines in red to a text editor.

Ansible Vault uses AES shared secret to encrypt the string. The above represents the encrypted form of your BIGIP password '**admin**'. You will now need to create an Ansible Vault credential on Ansible Tower with the shared secret.

From Ansible Tower, select the Credentials link:

The screenshot shows the Ansible Tower 'SETTINGS' page. A green box highlights the 'CREDENTIALS' section, which contains the following text: 'Add passwords, SSH keys, and other credentials to use when launching jobs against machines, or when syncing inventories or projects.' A large green arrow points from the left towards this box. Other sections visible include 'ORGANIZATIONS', 'USERS', 'TEAMS', 'INVENTORY SCRIPTS', 'NOTIFICATIONS', 'MANAGEMENT JOBS', 'INSTANCE GROUPS', 'CONFIGURE TOWER', 'ABOUT TOWER', and 'VIEW YOUR LICENSE'.

Create a new credential object and use the pre-defined Credential Type of Vault. Use the vault secret that you had keyed in earlier:

The screenshot shows the 'NEW CREDENTIAL' dialog box. The 'DETAILS' tab is selected. The 'NAME' field contains 'BIGIP_passphrase', the 'DESCRIPTION' field is empty, and the 'ORGANIZATION' dropdown is set to 'Penghon's Awesome Organization'. The 'CREDENTIAL TYPE' dropdown is set to 'Vault'. Under 'TYPE DETAILS', the 'VAULT PASSWORD' field contains '.....' and has a 'SHOW' button. A green box highlights the 'VAULT PASSWORD' field. At the bottom right are 'CANCEL' and 'SAVE' buttons.

Select **SAVE** and the newly created credential will appear similar to what you see below:

CREDENTIALS 3			
NAME	KIND	OWNERS	ACTIONS
BIGIP_passphrase	Vault	admin, Penghon's Awesome Organization	 
Demo Credential	Machine	admin	 
Penghon The Engineer's github account	Source Control	admin, Penghon's Awesome Organization	 

ITEMS 1 - 3

F5 INTERNAL USE ONLY

We will now need to modify the existing playbook to use the encrypted variable:

From your local git repository, modify and add the following lines highlight red below from the */git/ansible-repo/change_bigip_hostname.yaml* file:

```
---
```

```
- name: Testing importing group and host vars via Tower
  hosts: all
  gather_facts: no
  connection: local
  vars:
    bigip_admin_credentials: !vault |
      $ANSIBLE_VAULT;1.1;AES256
      3966643236363631383236316133643
      9663832376666313134623934366366393739666433356561
      6636366136346138
      663838373034313366393336646237380a353235333063656565623838353632666364
      3434663835396630303136316639313531666237646430316431643466343237663034
      38393362303131333634650a3964393433613333313838313136656561663366356539
      316538313662376333
```

```
tasks:
  - name: Setting the hostname of the target BIGIP
    bigip_hostname:
      hostname: "{{ hostname }}"
      provider:
        user: admin
        server: "{{ ansible_host }}"
        password: "{{ bigip_admin_credentials }}"
        validate_certs: no
```

After modification, commit the file changes and push to Github:

```
[root@localhost ansible-repo]# git commit -m "Modified to add encrypted variable" change_bigip_hostname.yaml
[master 082922c] Modified to add encrypted variable
 1 file changed, 9 insertions(+), 1 deletion(-)
[root@localhost ansible-repo]# git push
Counting objects: 5, done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 798 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To git@github.com:penghonengineer/ansible-repo.git
 3b1d64d..082922c master -> master
```

Repeat the above steps for */git/ansible-repo save_config.yaml*

Validate that the file changes have been made on the playbook on Github:

This screenshot shows a GitHub repository page for 'penghonengineer / ansible-repo'. The repository has 8 commits, 1 branch, 0 releases, and 2 contributors. The latest commit was made 4 minutes ago. A green arrow points to the 'Start an SCM update' button next to the 'Penghon's New Ansible Tower Project' row.

File	Commit Message	Time Ago
README.md	Initial commit	5 days ago
change_bigip_hostname.yaml	Modified to add encrypted variable	4 minutes ago
inventory	Adding sample inventory file	15 hours ago
README.md		

Issue the SCM update from Ansible Tower:

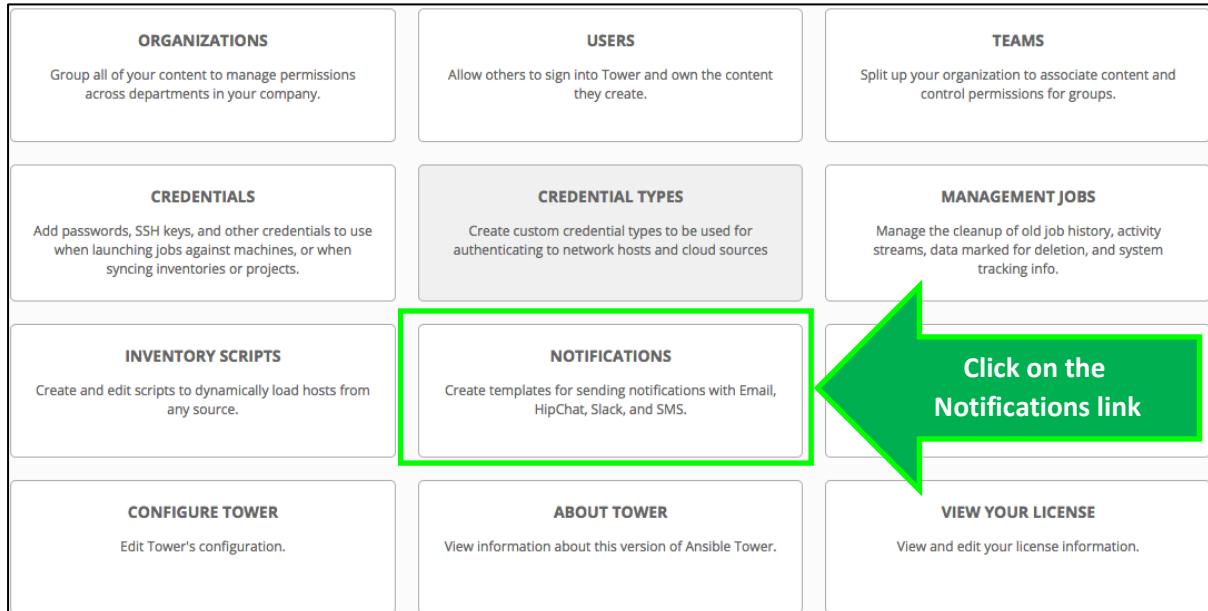
This screenshot shows the Ansible Tower interface under the 'PROJECTS' tab. It lists three projects: 'Demo Project', 'Penghon's New Ansible Tower Project', and 'ProjectA'. A green arrow points to the 'Start an SCM update' button for the 'Penghon's New Ansible Tower Project' row.

NAME	TYPE	REVISION	LAST UPDATED	ACTIONS
Demo Project	Git			
Penghon's New Ansible Tower Project	Git	3b1d64d	7/5/2018 22:00:00	
ProjectA	Manual		7/5/2018 16:28:11	

3.16 Ansible Tower Notification Configuration with Slack

You will be configuring a Slack notifier so that you can be notified of the job status whenever a job template completes.

Select NOTIFICATIONS from Settings:



Select **+ADD** and fill in the fields and ensure that slack is selected as the type.

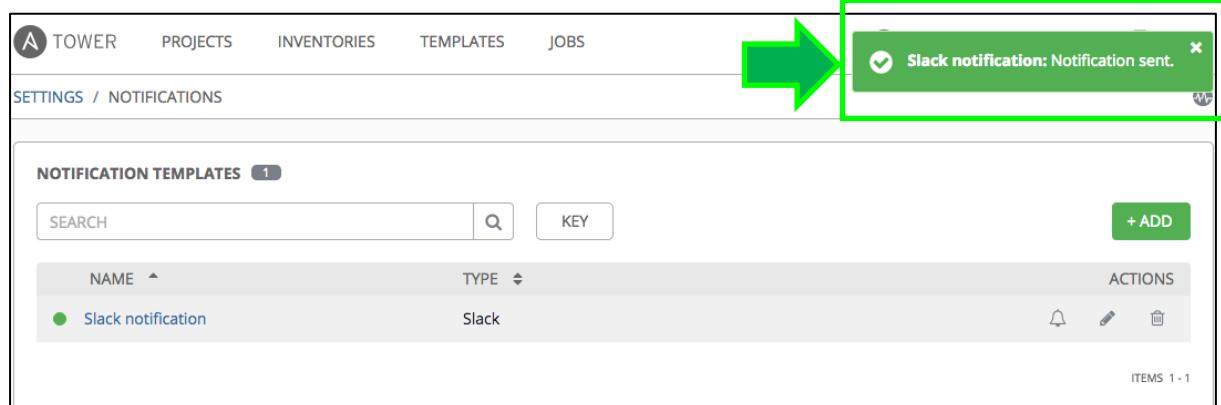
The destination channel is the previous channel that you had created in Slack:

The screenshot shows the 'New Notification Template' dialog. It has fields for 'NAME' (Slack notification), 'DESCRIPTION' (Slack channel for Tower notification), 'ORGANIZATION' (Penghon's Awesome Organization), 'TYPE' (Slack), and 'DESTINATION CHANNELS' (tower_notifications). A tooltip for 'DESTINATION CHANNELS' says: 'Enter one Slack channel per line. The pound symbol (#) is not required.' There are 'CANCEL' and 'SAVE' buttons at the bottom.

Click **SAVE** and the following should appear. Select the “test notification” button:

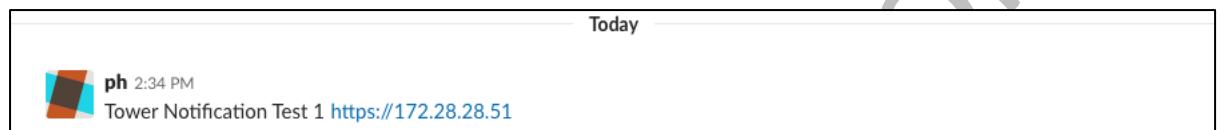
The screenshot shows the 'Notification Templates' list. It lists one template: 'Slack notification' (Type: Slack). A green arrow points to the 'Test notification' button next to the template entry. The table includes columns for 'NAME', 'TYPE', and 'KEY'. There are '+ADD', 'OPTIONS', and 'DELETE' buttons at the top right of the list.

Ansible Tower will attempt to send a test message to your configured Slack channel. Once Tower is able to send the message, the following popup will be seen:



A screenshot of the Ansible Tower interface. At the top, there are navigation links: TOWER, PROJECTS, INVENTORIES, TEMPLATES, and JOBS. Below this, a sub-menu shows SETTINGS / NOTIFICATIONS. A green arrow points from the left towards a green notification box. The notification box contains a checkmark icon and the text "Slack notification: Notification sent." There is also a small 'X' icon in the top right corner of the box. The main content area is titled "NOTIFICATION TEMPLATES" and shows one item: "Slack notification" (Type: Slack). There are "ACTIONS" buttons (bell, edit, delete) and a note "ITEMS 1 - 1".

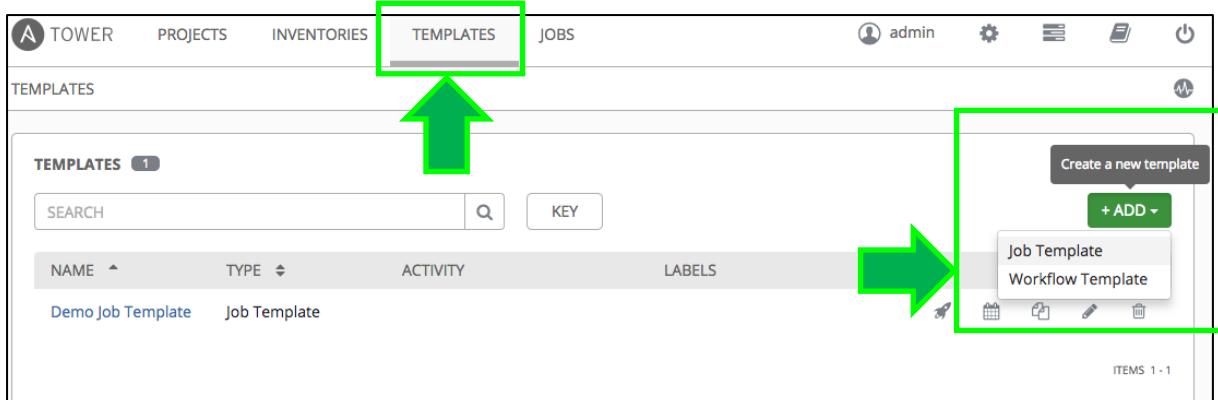
Check your Slack channel that you have received the test message from Ansible Tower:



A screenshot of a Slack message window. The header says "Today". A message from a user with a blue and orange profile picture is shown, timestamped at 2:34 PM. The message reads: "Tower Notification Test 1 <https://172.28.28.51>".

3.17 Ansible Tower Job Template Configuration

An Ansible Tower job template defines playbook execution with RBAC permissions, reusability with surveys, job notifications, and several other parameters. From the Ansible Tower dashboard, select TEMPLATES and +ADD to create a Job Template:



Fill in the fields as shown below and select the **change_bigip_hostname.yaml** Playbook, and the previously created vault credentials. In the EXTRA VARIABLES section, key in the **hostname** variable and replace the value with a hostname of your own choosing as shown below:

The screenshot shows the 'CREATE JOB TEMPLATE' dialog. At the top, it says 'TEMPLATES / CREATE JOB TEMPLATE'. Below that, there are tabs for DETAILS (selected), PERMISSIONS, NOTIFICATIONS, COMPLETED JOBS, and ADD SURVEY. The DETAILS tab contains fields for NAME ('Modify BIGIP hostname'), DESCRIPTION ('This simple playbook will change the BIGIP's host'), INVENTORY ('BIGIP'), CREDENTIAL ('Vault: BIGIP_passphrase'), and PLAYBOOK ('change_bigip_hostname.yaml'). Under OPTIONS, there are checkboxes for Enable Privilege Escalation, Allow Provisioning Callbacks, Enable Concurrent Jobs, and Use Fact Cache. The EXTRA VARIABLES section at the bottom has tabs for YAML (selected) and JSON. It contains a code editor with the following content:

```
1
2 hostname: bigip1.penghon.local
```

Select **SAVE** and the new Job Template will be shown as below:

The screenshot shows a list of templates. At the top, there is a search bar, a key icon, and a green '+ ADD' button. Below the header, there are columns for NAME, TYPE, ACTIVITY, and LABELS. The first entry is 'Demo Job Template' (Job Template). The second entry is 'Modify BIGIP hostname' (Job Template), which is highlighted with a blue selection bar. The bottom right corner of the list area says 'ITEMS 1 - 2'.

Modify the inherited notification to notify you when the Job instance is successful or when it failed:

The screenshot shows the configuration for the 'Modify BIGIP hostname' template. It has tabs for DETAILS, PERMISSIONS, NOTIFICATIONS (which is selected), and COMPLETED JOBS. In the NOTIFICATIONS tab, there is a search bar, a key icon, and a link to 'GO TO NOTIFICATIONS TO ADD A NEW TEMPLATE'. Below the search bar, there are columns for NAME, TYPE, and ACTIVITY. The entry is 'Slack notification' (Slack). To the right of the entry, there are two buttons: 'SUCCESS' (ON) and 'FAILURE' (ON), both highlighted with green arrows pointing to them. The bottom right corner says 'ITEMS 1 - 1'.

Launch the template job by selecting the rocket icon:

The screenshot shows the 'TEMPLATES' list again. The 'Modify BIGIP hostname' template is selected. In the 'ACTIONS' column for this entry, there is a button labeled 'Start a job using this template' with a rocket icon. This button is highlighted with a large green arrow. The bottom right corner says 'ITEMS 1 - 2'.

Ansible Tower will switch to the live update of the Job instance after the Job Template has been executed. Notice that the playbook hasn't been run. Why?

The screenshot shows the Ansible Tower interface. On the left, a sidebar lists various project components: PROJECTS, INVENTORIES, TEMPLATES, and JOBS. Under 'JOBS', it says 'JOBS / 10 - Modify BIGIP hostname'. The main area displays a 'DETAILS' panel for the job instance. The 'STATUS' is 'Running'. The 'TEMPLATE' is 'Modify BIGIP hostname'. The 'JOB TYPE' is 'Run'. It was 'LAUNCHED BY' 'admin'. The 'INVENTORY' is 'BIGIP'. The 'PROJECT' is 'Penghon's New Ansible Tower Project'. The 'PLAYBOOK' is 'change_bigip_hostname.yaml'. The 'VAULT CREDENTIAL' is 'BIGIP_passphrase'. 'FORKS' is 0. 'VERBOSITY' is 0 (Normal). The 'INSTANCE GROUP' is 'tower'. Under 'EXTRA VARIABLES', there is a code block:

```
1 hostname: bigip1.penghon.local
```

Upon successful execution of the job you would see the standard output as you would run ansible-playbook via Ansible Engine.

The screenshot shows the Ansible Tower interface again. The job instance 'Modify BIGIP hostname' is now in 'Successful' status, indicated by a green arrow pointing to the status field in the details panel. The log output on the right shows the execution process:

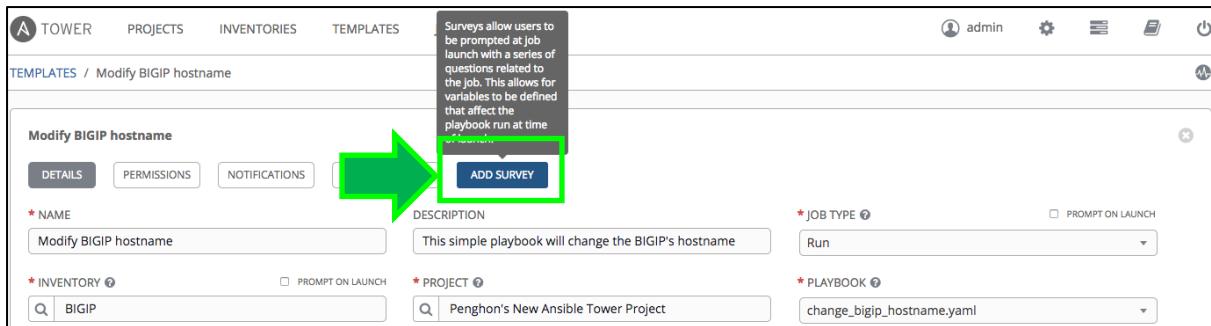
```
PLAY [Testing importing group and host vars via Tower] *****
TASK [Setting the hostname of the target BIGIP] *****
changed: [bigip1]
PLAY RECAP *****
bigip1 : ok=1    changed=1    unreachable=0    failed=0
```

Check that your slack channel has the job success/failure notification.

3.18 Configuring Ansible Tower Surveys

Ansible Tower Surveys only available for Enterprise-level licenses. Surveys set the extra variables but in a user-friendly question-and-answer manner. You can set limits or user input validation or create multiple-choice questions.

Access the previously created Job Template and click on **ADD SURVEY**:



The sample playbook **change_bigip_hostname.yaml** sets the BIGIP hostname using the `bigip_hostname` module. This module requires the `hostname` parameter. We will use this to demonstrate the purpose of the Ansible Template Survey.

Fill in the fields below to create a user-friendly survey that asks for the desired hostname for your BIGIP. Complete the survey by clicking on **+ADD** and subsequently the **SAVE** button.

The screenshot shows the 'ADD SURVEY PROMPT' dialog box. The 'SURVEY' tab is active. The 'PREVIEW' section displays the message 'PLEASE ADD A SURVEY PROMPT.'. The 'ADD SURVEY PROMPT' section contains the following configuration:

- PROMPT:** What is the hostname for the BIGIP?
- DESCRIPTION:** (empty)
- ANSWER VARIABLE NAME:** hostname
- ANSWER TYPE:** Text
- MINIMUM LENGTH:** 0
- MAXIMUM LENGTH:** 128
- DEFAULT ANSWER:** bigip1.penghon.local
- REQUIRED:**

At the bottom right of the dialog box are buttons for **CLEAR**, **+ADD**, **CANCEL**, **SAVE**, and **MOVE**.

Launch the Job Template and key in your desired hostname for your BIGIP.

The screenshot shows the Ansible Tower interface. In the background, there's a list of templates under 'TEMPLATES'. One template, 'Modify BIGIP hostname', is selected. In the foreground, a modal dialog titled 'LAUNCH JOB | Modify BIGIP hostname' is open. It contains a 'SURVEY' section with a question 'WHAT IS THE HOSTNAME OF THE BIGIP THAT YOU DESIRE?' and a text input field containing 'HUGEIP1.penghon.local'. Below the survey, it shows the selected 'INVENTORY' (BIGIP) and 'CREDENTIAL' (Vault: BIGIP_passphrase). At the bottom of the dialog are 'CANCEL' and 'LAUNCH' buttons. The main interface has tabs for 'PROJECTS', 'INVENTORIES', 'TEMPLATES', and 'JOBS' at the top. A user 'admin' is logged in. There are also various configuration options like 'PROMPT ON LAUNCH' and 'LIMIT'.

Complete the job run and check that it succeeds.

Access your Slack channel to determine if the job runs successful or not.

From your Ansible Tower, SSH to your assigned BIGIP to determine if the hostname has been changed or not.

3.19 Role-based Access Control

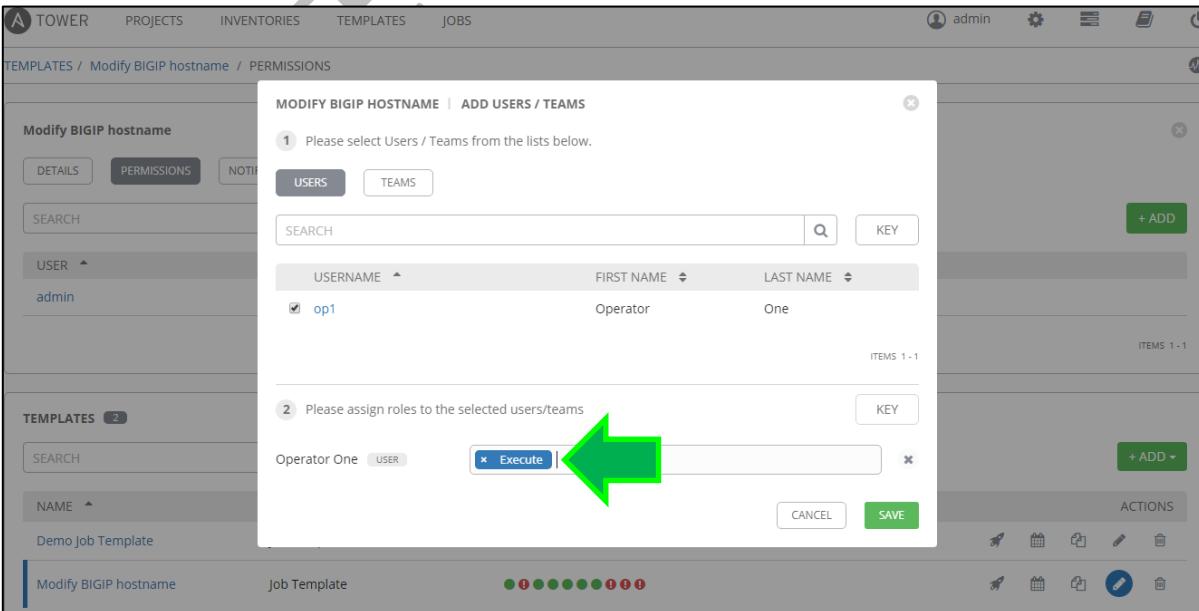
Ansible Tower has been designed primarily with RBAC as one of its main features. Role-Based Access Controls (RBAC) are built into Tower and allow Tower administrators to delegate access to server inventories, organizations, and more. Administrators can also centralize the management of various credentials, allowing end users to leverage a needed secret without ever exposing that secret to the end user. RBAC controls allow Tower to help you increase security and streamline management.

Now log out as admin. Log back in as user that you have created previously:



The image shows the Ansible Tower login interface. It features a logo with a red 'A' inside a circle, followed by the text 'ANSIBLE TOWER by Red Hat'. Below the logo, a message says 'Welcome to Ansible Tower! Please sign in.' There are two input fields: 'USERNAME' containing 'op1' and a redacted 'PASSWORD' field. A green 'SIGN IN' button is located to the right of the password field.

Does this user have access to the Job template previously created? Log out as Op1, log back in as admin. Enable permissions for the Job template for the User or Team that you have created previously:



The image shows the Ansible Tower 'MODIFY BIGIP HOSTNAME | ADD USERS / TEAMS' dialog. It displays a list of users ('op1') and teams ('admin'). A green arrow points to the 'Execute' button at the bottom of the dialog. The background shows the Ansible Tower interface with a 'Job Template' card.

If you have added execute permissions for the User, you will see the summary of the permissions as follows:

The screenshot shows a table of user permissions. The columns are labeled 'USER', 'ROLE', and 'TEAM ROLES'. There are two rows: one for 'admin' and one for 'op1'. The 'admin' row has 'ADMIN' and 'SYSTEM ADMINISTRATOR' listed under 'ROLE'. The 'op1' row has 'EXECUTE' listed under 'ROLE'. A green '+ ADD' button is visible at the top right.

USER	ROLE	TEAM ROLES
admin	ADMIN SYSTEM ADMINISTRATOR	
op1	EXECUTE	

ITEMS 1 - 2

Log out and log in as Op1 user.

The screenshot shows a table of job templates. The columns are labeled 'NAME', 'TYPE', 'ACTIVITY', 'LABELS', and 'ACTIONS'. There is one row for 'Modify BIGIP hostname', which is a 'Job Template'. The 'ACTIVITY' column shows several green and red circular icons. A green '+ ADD' button is visible at the top right.

NAME	TYPE	ACTIVITY	LABELS	ACTIONS
Modify BIGIP hostname	Job Template	● ● ● ● ● ● ● ● ● ●		🚀 🗓️ 🔎

ITEMS 1 - 1

Launch the Job. Does it work?

3.20 Configuring an Ansible Workflow Template

A workflow job template links together a sequence of disparate job templates and accomplishes the task of tracking the full set of jobs that were part of the release process as a single unit.

We will be chaining the execution of 2 playbooks in a single Job Workflow.

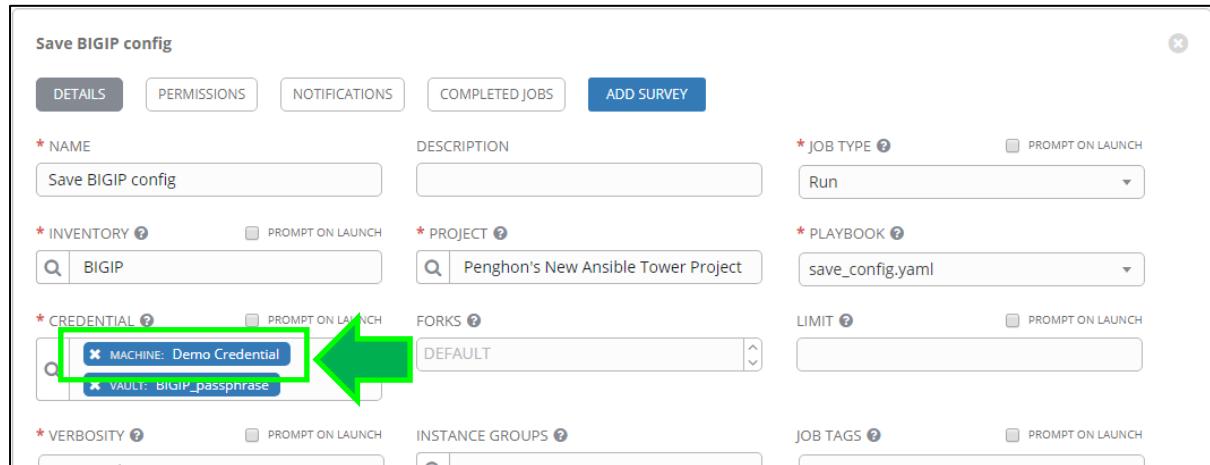
Create a new Job Template by referencing the **save_config.yaml** playbook:

The screenshot shows the 'CREATE JOB TEMPLATE' page in Ansible Tower. The 'PLAYBOOK' field is highlighted with a green box and an arrow pointing to it. The 'CREDENTIAL' field is also highlighted with a green box and an arrow pointing to it. The 'PLAYBOOK' field contains 'save_config.yaml' and the 'CREDENTIAL' field contains 'BIGIP_passphrase'.

In order for the Workflow Editor to be able to see the templates for inclusion into a Workflow you need to modify the 2 Job Templates and include the Demo Credential. Select the previously created Job Template and add the Demo Credential to the Credential field:

The screenshot shows the 'Modify BIGIP hostname' job template configuration. The 'CREDENTIAL' field is highlighted with a green box and an arrow pointing to it. The 'CREDENTIAL' field contains 'BIGIP_passphrase'.

Repeat for the other Job Template:



Save BIGIP config

DETAILS PERMISSIONS NOTIFICATIONS COMPLETED JOBS ADD SURVEY

* NAME Save BIGIP config DESCRIPTION

* JOB TYPE Run PROMPT ON LAUNCH

* INVENTORY BIGIP PROMPT ON LAUNCH * PROJECT Penghon's New Ansible Tower Project

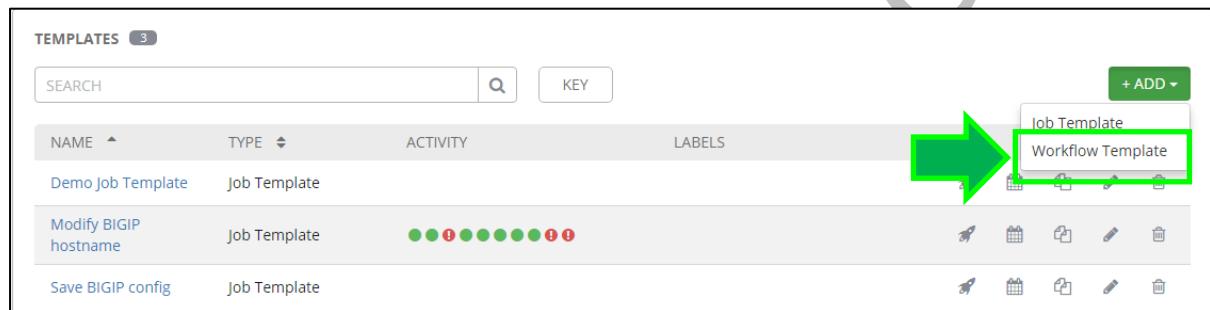
* CREDENTIAL Demo Credential PROMPT ON LAUNCH * PLAYBOOK save_config.yaml

* VAULT: BIGIP_passphrase

* FORKS DEFAULT LIMIT PROMPT ON LAUNCH

* VERBOSITY INSTANCE GROUPS JOB TAGS PROMPT ON LAUNCH

From Templates, click on **+ADD** and select Workflow Template:



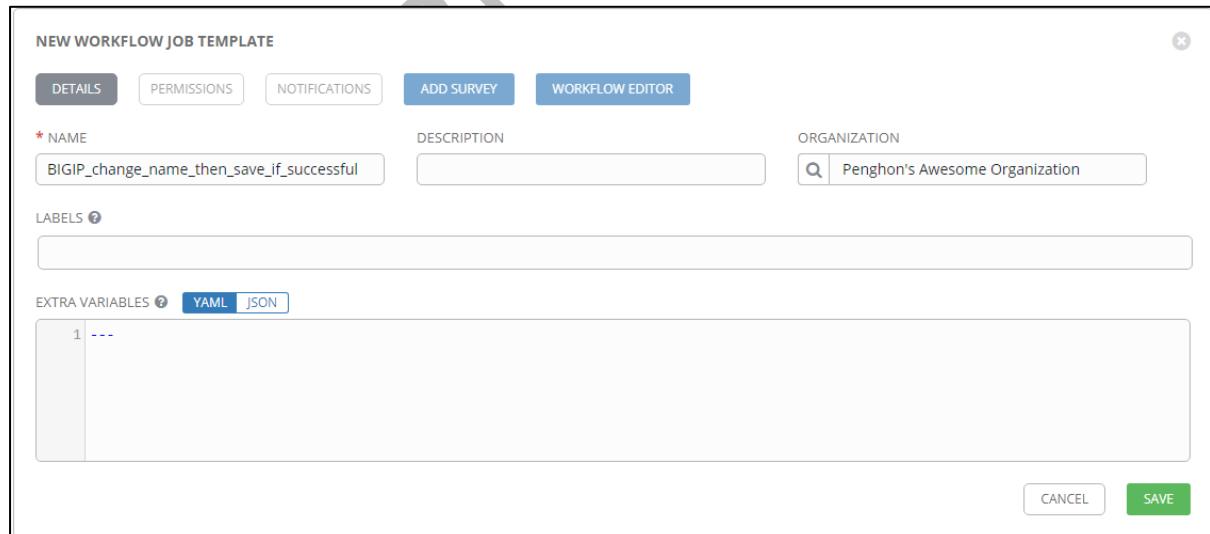
TEMPLATES 3

SEARCH KEY

+ ADD

NAME	TYPE	ACTIVITY	LABELS
Demo Job Template	Job Template		
Modify BIGIP hostname	Job Template	● ● ● ● ● ● ● ●	
Save BIGIP config	Job Template		

Create a new Workflow Template by filling in the necessary fields and saving the configuration:



NEW WORKFLOW JOB TEMPLATE

DETAILS PERMISSIONS NOTIFICATIONS ADD SURVEY WORKFLOW EDITOR

* NAME BIGIP_change_name_then_save_if_successful DESCRIPTION

ORGANIZATION Penghon's Awesome Organization

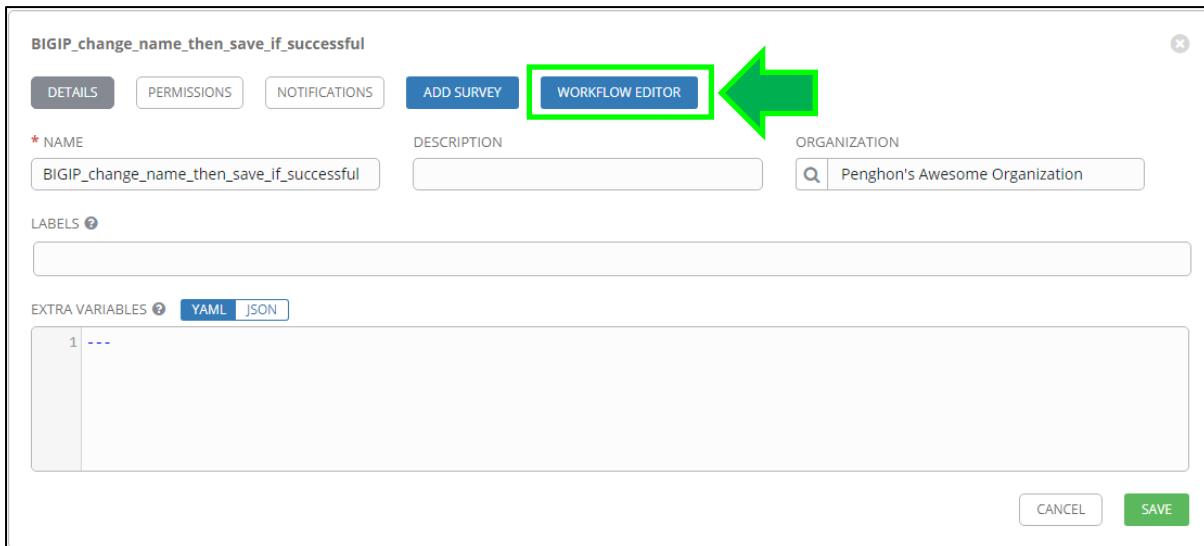
LABELS

EXTRA VARIABLES YAML JSON

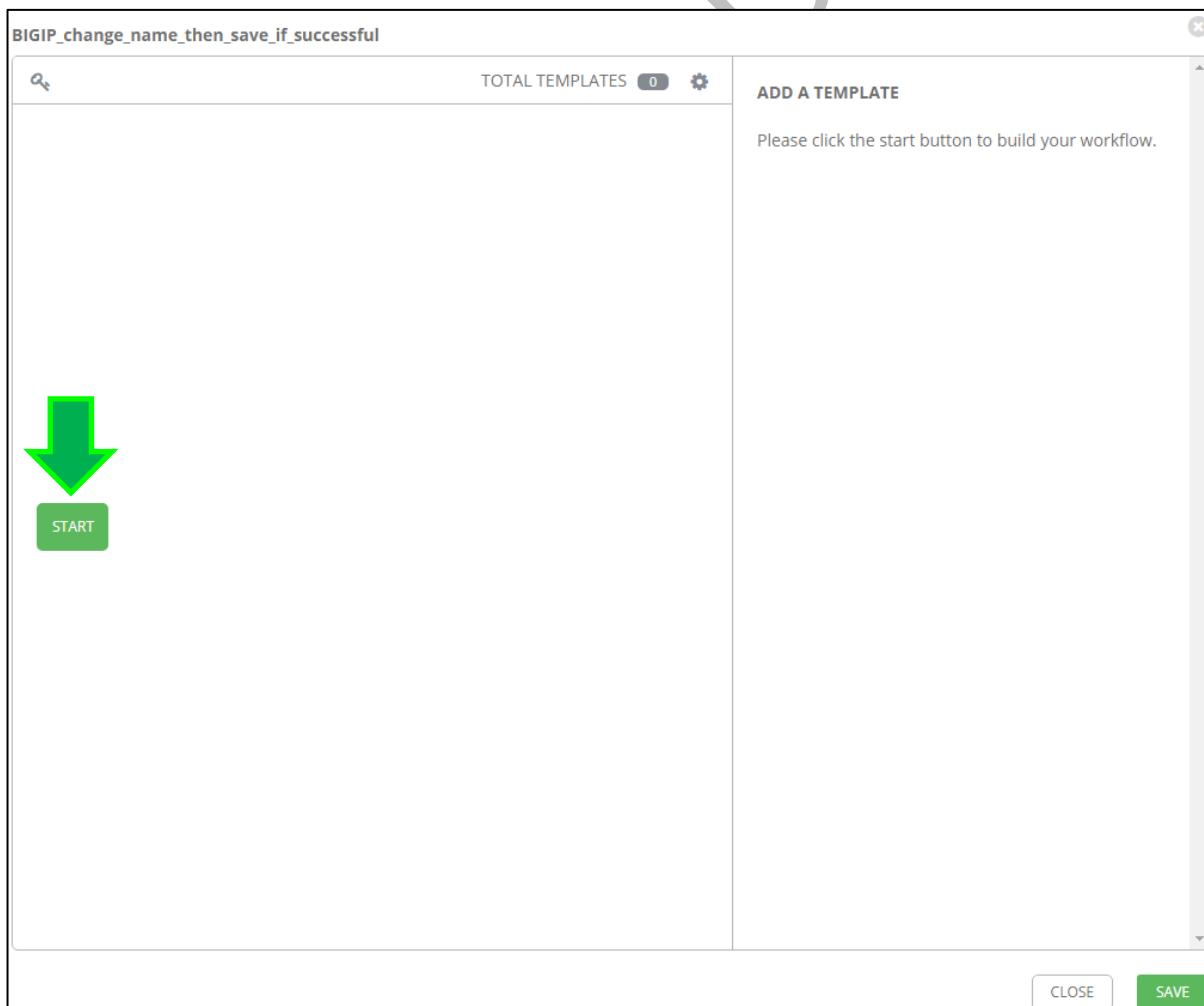
1 ---

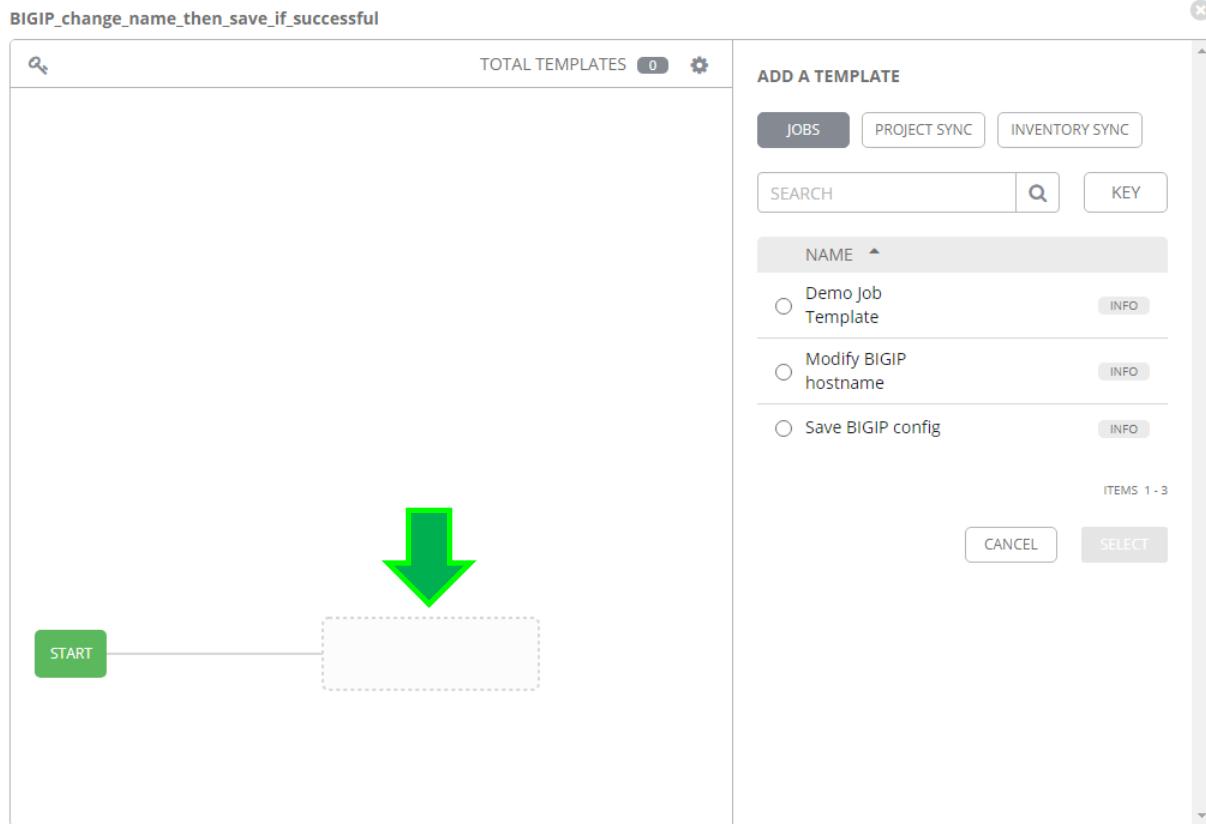
CANCEL SAVE

Select the WORKFLOW EDITOR:

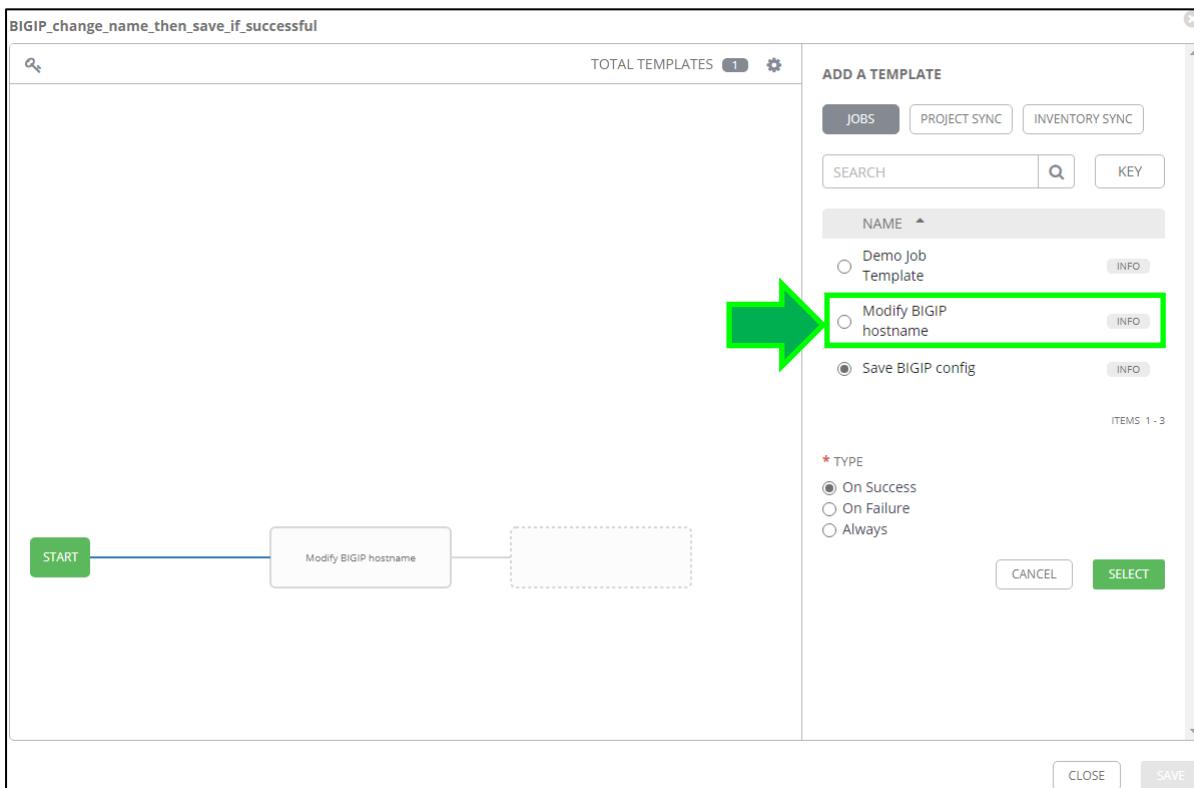


The Workflow editor will appear and you will configure the desired workflow by clicking on the green start button:

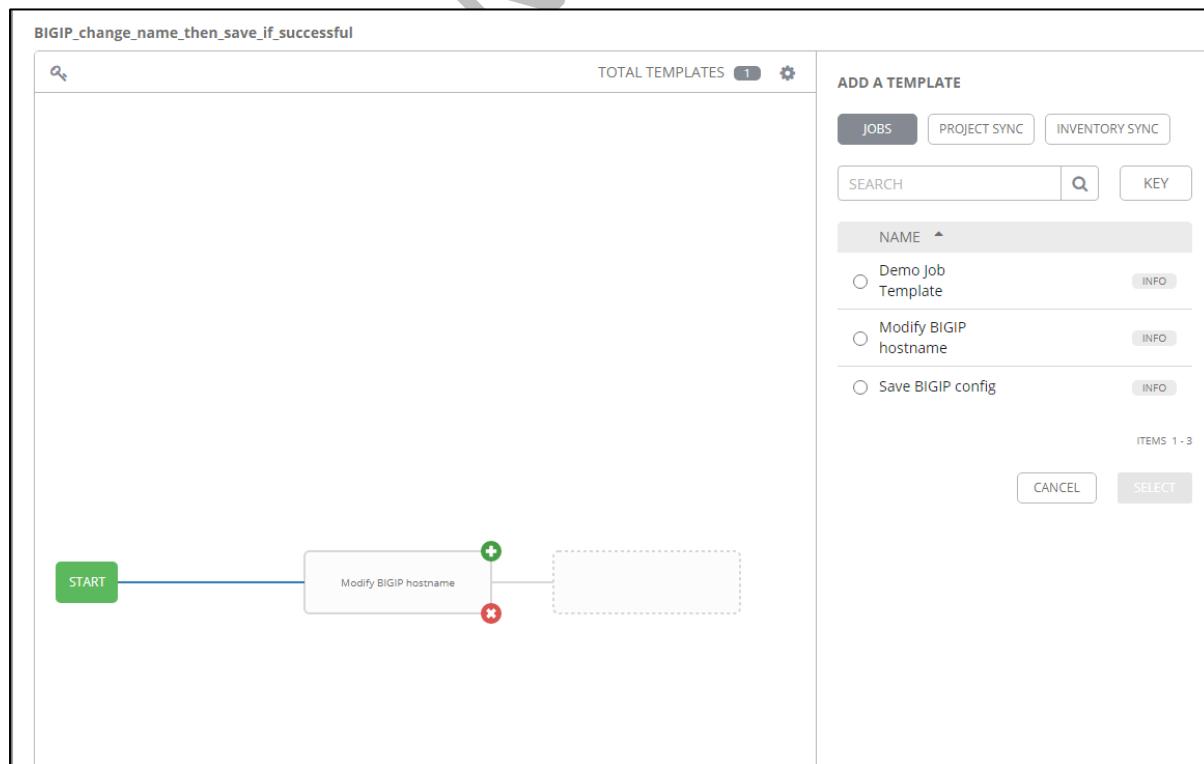




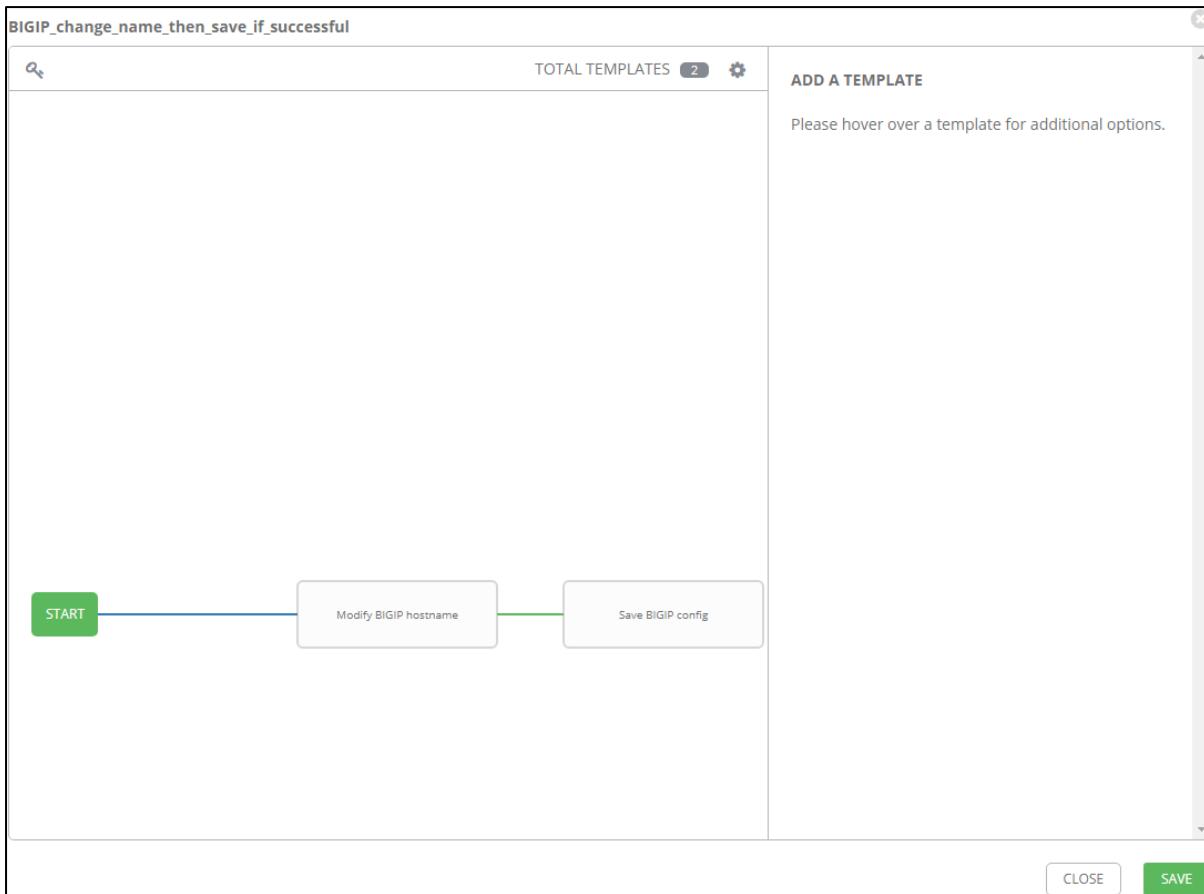
Workflows allow you to determine which Job Template to run based on the state of the previously executed Job. For the purpose of this lab task we will be chaining two Job Templates to be run one after another based on success of the initial Job Template. Select the “Modify BIGIP hostname” Job template (substitute with your Job Template if it is named differently).:



Click on the green + sign and select the second Job Template:



Once your Workflow Template looks like the following, click **SAVE**:



Select Notifications and enable the Slack notifier to notify on both success and failure of the job run:

The screenshot shows the "NOTIFICATIONS" tab for the workflow template "BIGIP_change_name_then_save_if_successful". The tab also includes "DETAILS" and "PERMISSIONS" tabs. The "NOTIFICATIONS" tab has a "SEARCH" field and a "KEY" button. To the right, there's a link "GO TO NOTIFICATIONS TO ADD A NEW TEMPLATE". Below these are two rows of notification entries. The first row is for "Slack notification" of type "Slack", with "SUCCESS" and "FAILURE" switches both set to "ON" (indicated by blue buttons). Two large green arrows point upwards towards these "ON" buttons. The second row is partially visible. At the bottom right, it says "ITEMS 1 - 1".

Edit the permissions of the Workflow Template to allow the Operator User/Team to execute the Workflow Template:

The screenshot shows the A TOWER application interface. The top navigation bar includes 'PROJECTS', 'INVENTORIES', 'TEMPLATES', and 'JOBS'. On the right, there's a user icon labeled 'admin' and various settings icons. The main area displays a 'TEMPLATES / BIGIP_change_name_then_save_if_successful / PERMISSIONS' page. A modal window titled 'BIGIP_CHANGE_NAME_THEN_SAVE_IF_SUCCESSFUL | ADD USERS / TEAMS' is open. It contains two steps: 1. 'Please select Users / Teams from the lists below.' with tabs for 'USERS' (selected) and 'TEAMS'. A search bar and a 'KEY' button are present. A table shows one user: op1 (checked), Operator, One. 2. 'Please assign roles to the selected users/teams' with a table showing 'Operator One' assigned to 'USER'. Buttons for 'Execute', 'CANCEL', and 'SAVE' are at the bottom. The left sidebar shows 'TOWER' with 'DETAILS' and 'PERMISSIONS' buttons, and a 'TEMPLATES' section with '4' items, including 'BIGIP_change_name_then_save_if_successful' which is highlighted.

Logout then log in as **op1**. Run the workflow template. Was there any survey prompt?

3.21 Running dev F5 modules

Running F5 dev ansible modules is the same as how you would do for Ansible Engine/CLI. Create `ansible.cfg` within the specific project folder and specify the library first path of module discovery. Ensure that the dev module exists within the library.

3.22 Upgrading

Upgrading the Ansible Tower is as easy as installing it. Unpack the installation files and ensure that the ***inventory*** file is copied over from the previous installation and just execute the ***./setup.sh***. The installation process determines by default whether it is a fresh install or an upgrade.