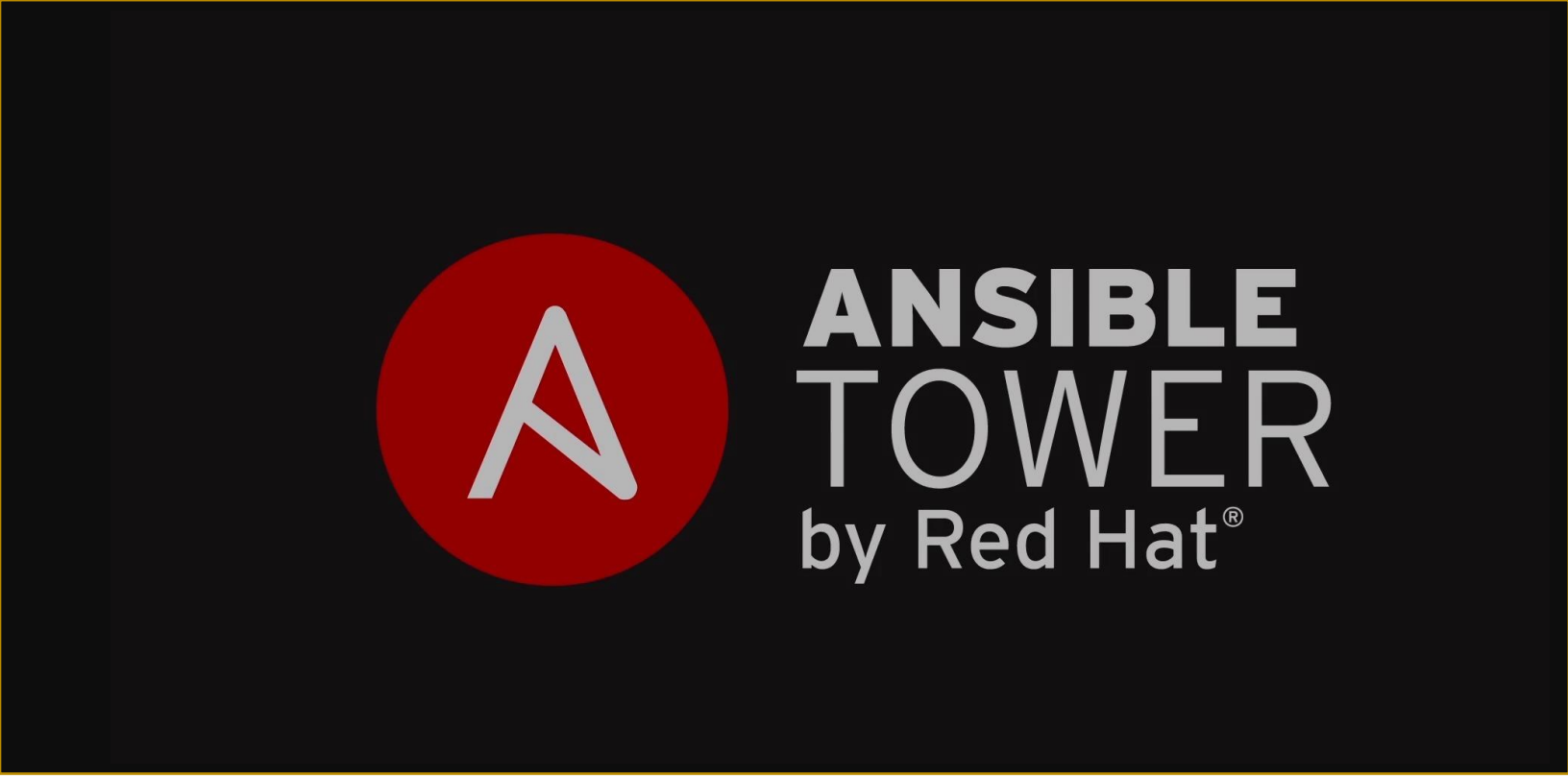

ANSIBLE
TOWER
by Red Hat®

Contents

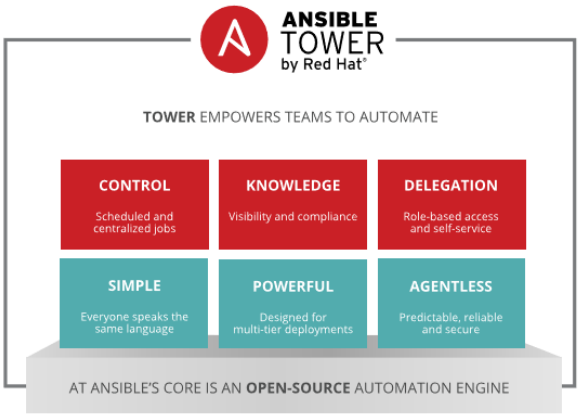
Introduction to Ansible Tower

Ansible is a **simple automation language** that can perfectly describe an IT application infrastructure in Ansible Playbooks. It's an **automation engine** that runs Ansible Playbooks.

Ansible Tower by **Red Hat** helps you scale IT automation, manage complex deployments and speed productivity. Centralize and control your IT infrastructure with a visual dashboard, role-based access control, job scheduling and graphical inventory management.

Ansible Tower is an **enterprise framework** for controlling, securing and managing your Ansible automation – with a **UI and RESTful API**

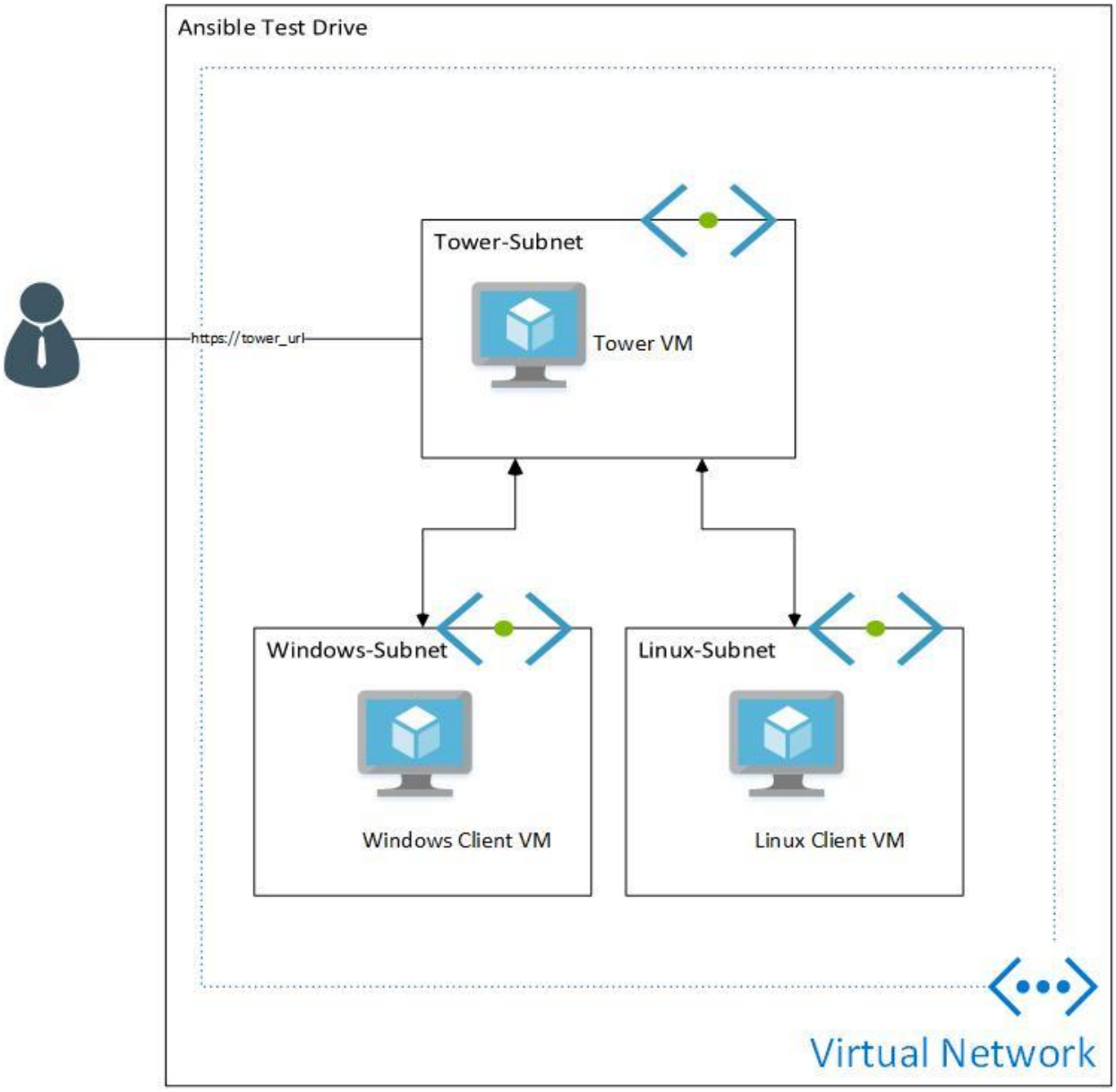
- **Role-based access control** keeps environments secure, and teams efficient
- Non-privileged users can **safely deploy** entire applications with **push-button deployment** access
- All Ansible automations are **centrally logged**, ensuring **complete auditability and compliance**



About the Test Drive

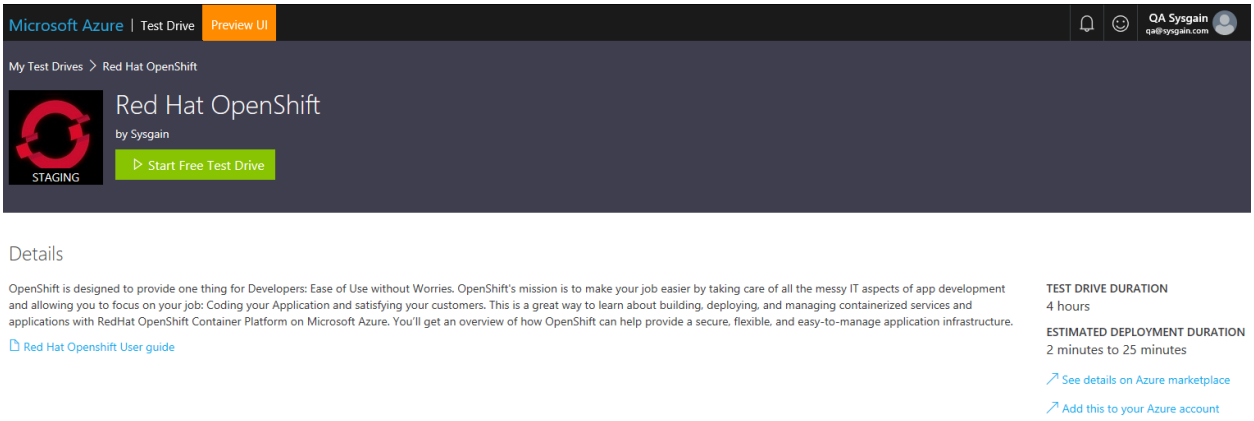
This test drive will help you experience the capabilities of Ansible Tower. Ansible Tower comes with a feature rich Dashboard to provide users with a nice interface to work with server management. We will be managing Windows as well as a Linux Virtual Machines using Ansible Tower and deploy web servers in that. Are you ready to take the Driver seat and experience Red Hat Ansible test drive……?

Deployment Architecture

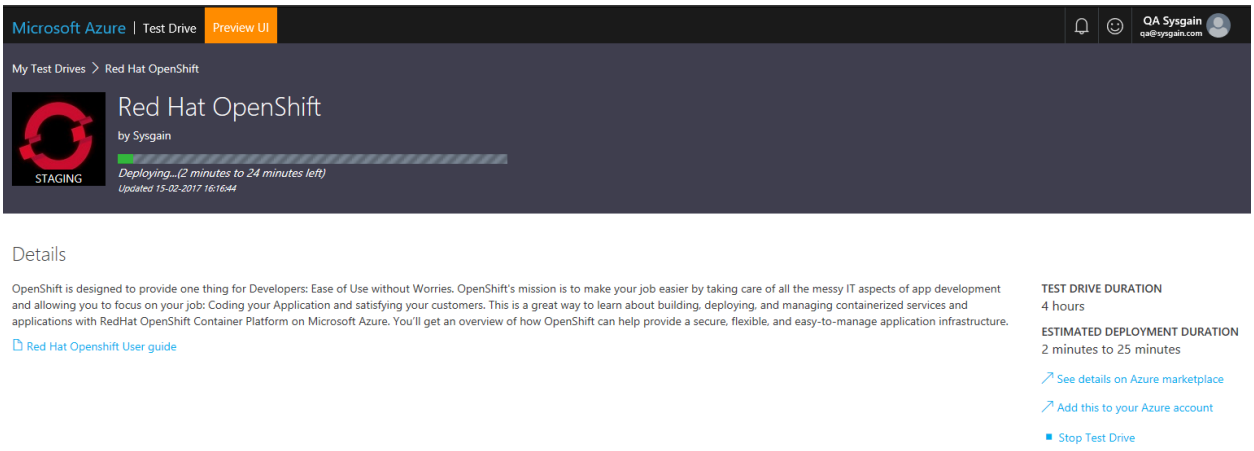


Getting Started

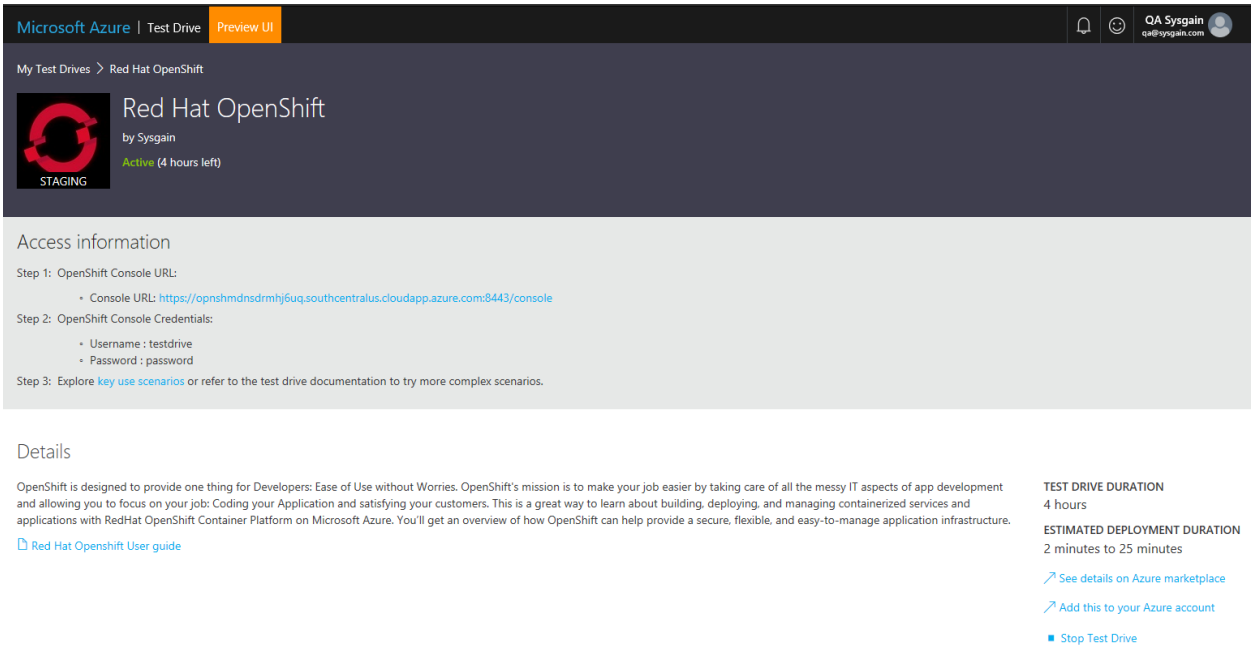
When you visit the Microsoft Azure Test Drive page for Red Hat Ansible Tower you see a web page as seen below.



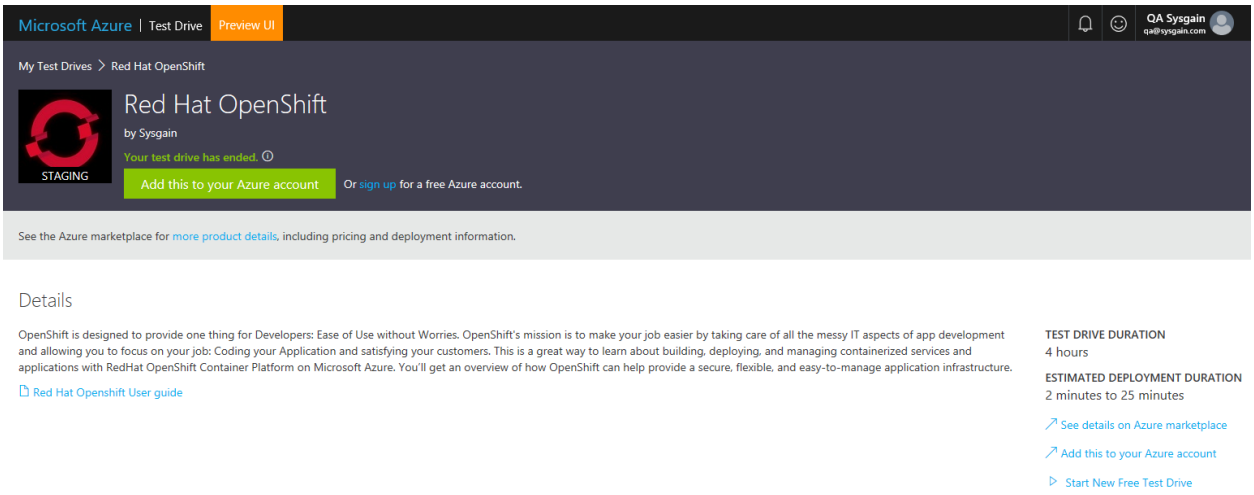
Once you click on ***Start Free Test Drive,*** the test drive starts deploying. It will take 2-30 minutes to successfully launch the test drive.



After the test drive provisioning is complete, login credentials, and required details are provided in the access information as well as by email.



Once your test drive gets expired you see this following page.

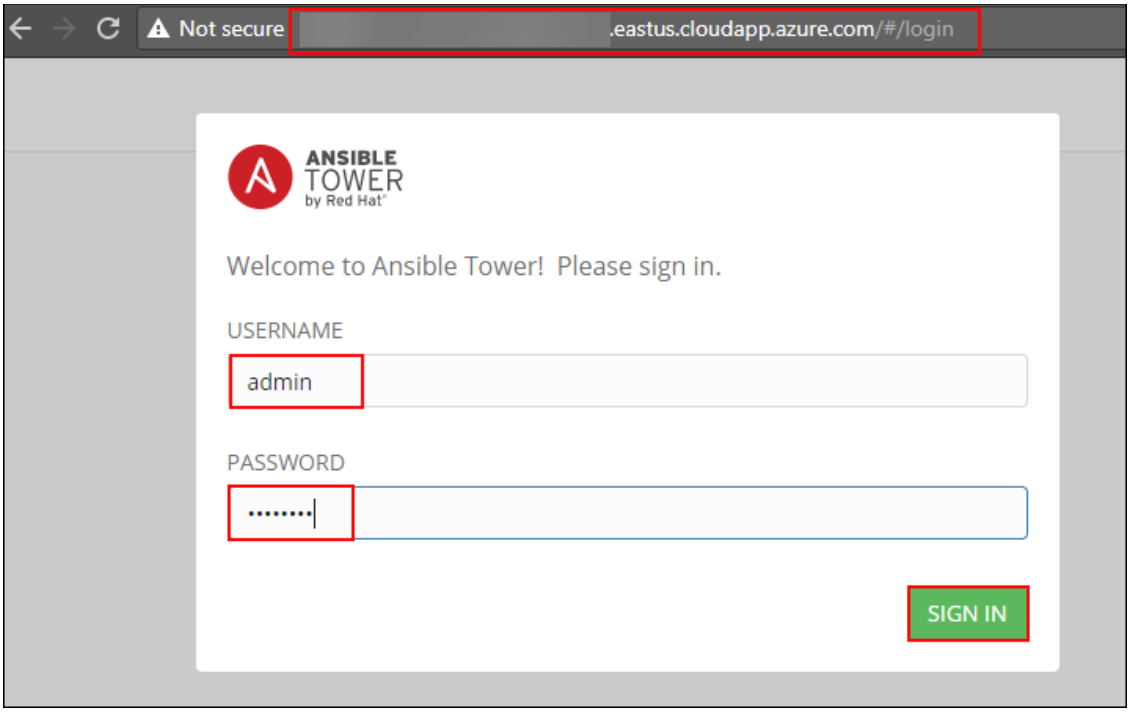


Lab 1: Getting Started with Ansible Tower

Access ansible tower

During this lab, we are going to log in to Tower using the Ansible Tower DNS Name and other login details received via email or provided in the access information, browse to the Tower interface at: http://<TowerDNSName>

Log in using the Tower username and password details as provided. The default username set during installation is 'admin'. Password will be provided in the email.



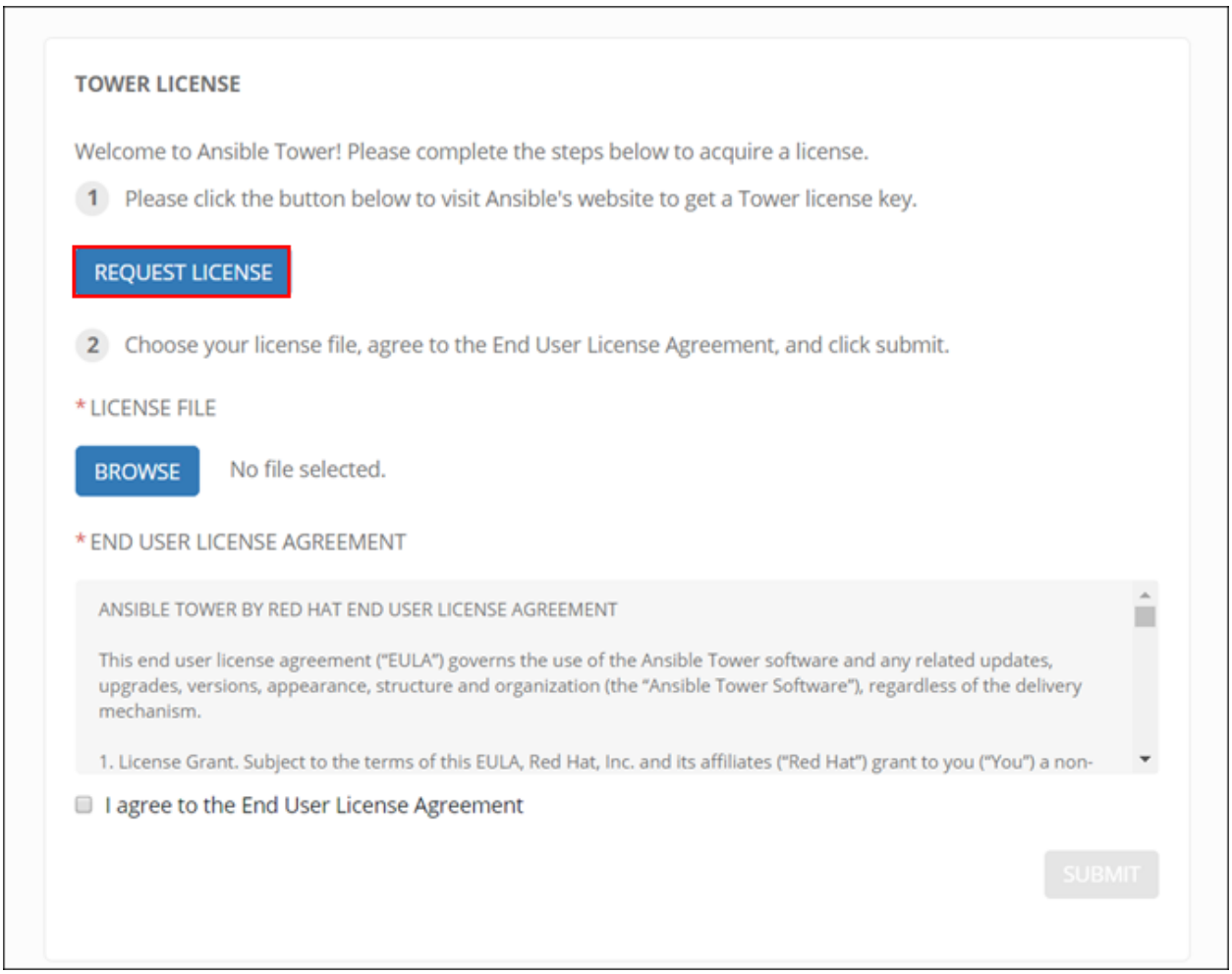
Once we sign in, we will be directed to the Tower License page.

Request for trial license

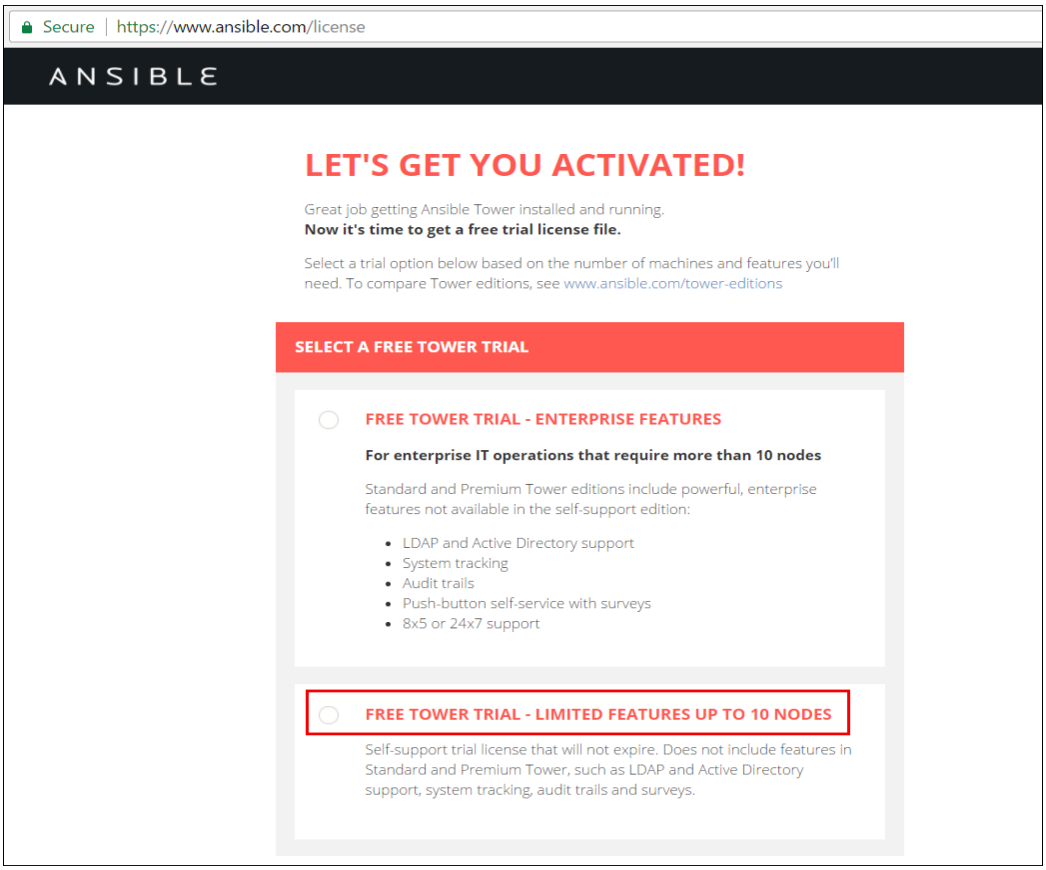
During this lab, we will request for Ansible Tower Trial License. A license is required for Ansible Tower to run. If you already have a license, you can skip this exercise. Ansible Tower by Red Hat ("**Ansible Tower**") is a proprietary software product provided via an annual subscription entered between you and Red Hat, Inc. ("**Red Hat**").

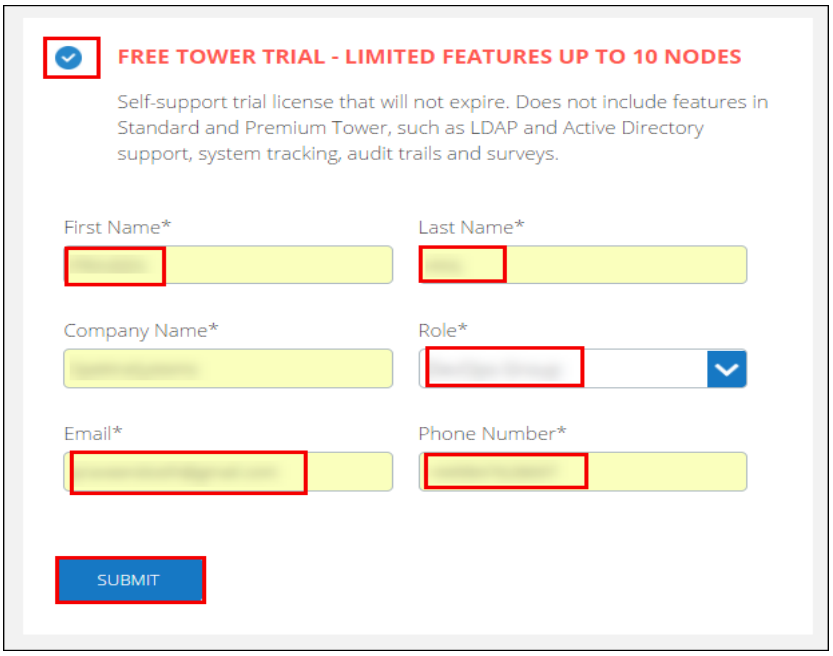
While a license is required for Ansible Tower to run, there is no fee for managing up to 10 hosts. Additionally, trial licenses are available for exploring Ansible Tower with a larger number of hosts.

- Trial licenses for Ansible Tower are also available at: http://ansible.com/license



We can select Free Tower Trial – Limited Features up to 10 Nodes for this test drive and then complete the steps for license request.





We must make sure, we provide a valid email id. The license file will be mailed to that email address in 2 to 10 mins.

Assign License

Once we receive the trial license via email, we will upload the license file.

To add the license:

1. Save the license (or save the license contents to a text file locally, if needed).
2. Click the **Browse** button and navigate to the location where the license file is saved to upload it. The uploaded license may be a plain text file or a JSON file, and must include properly formatted JSON code.
3. Once uploaded, check to agree to the End User License Agreement and click **Submit**.

Once the license has been accepted, Tower navigates to the main Ansible interface for the Dashboard (which we can access by clicking on the Ansible Tower logo at the top left of the screen as well).

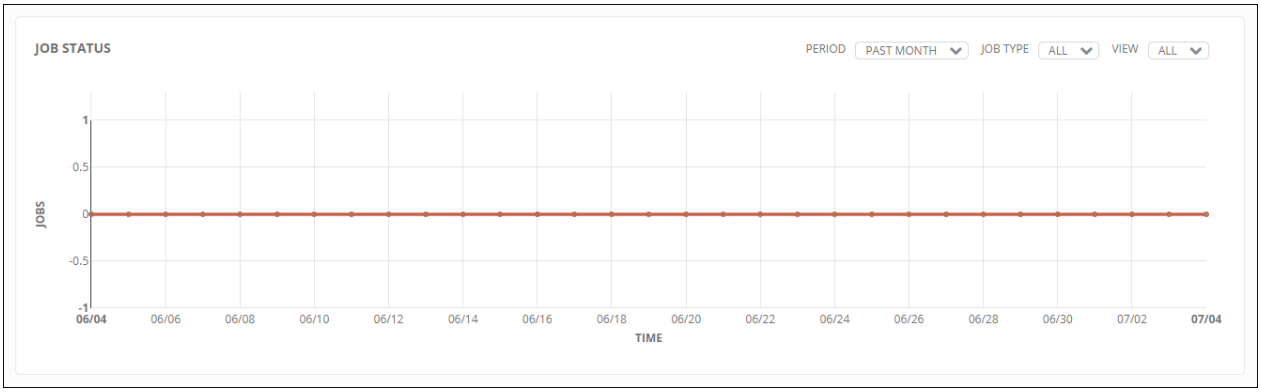
Tower Dashboard

The Tower Dashboard offers a friendly graphical framework for your IT orchestration needs. Across the top-left side of the Tower Dashboard, administrators can quickly navigate to their **Projects**, **Inventories**, **Job Templates**, and **Jobs**.

Across the top-right side of this interface, administrators can access the tools they need to configure organizations, users, groups, and permissions as well as view related documentation, access portal mode, and log out.

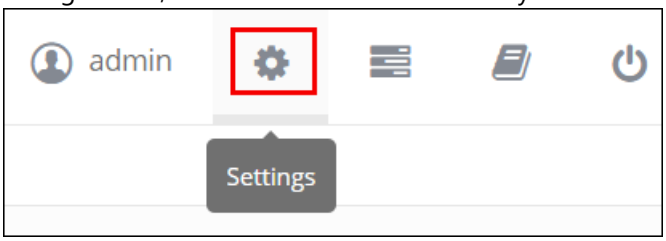
At the top of the Dashboard is a summary of your hosts, inventories, and projects. Each of these is linked to the corresponding object in Tower, for easy access.

On the main Tower Dashboard screen, a summary appears listing your current **Job Status**. Also, available for review are summaries of **Recently Used Job Templates** and **Recently Run Jobs**.



Add Credentials

To enter the Settings Menu screen for Ansible Tower, click the button. This screen allows you to create your organizations, add credentials, add users and teams, schedule management jobs, modify your Tower's configuration, and more. You can also view your license from the Settings Menu's 'View Your License' link.



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.

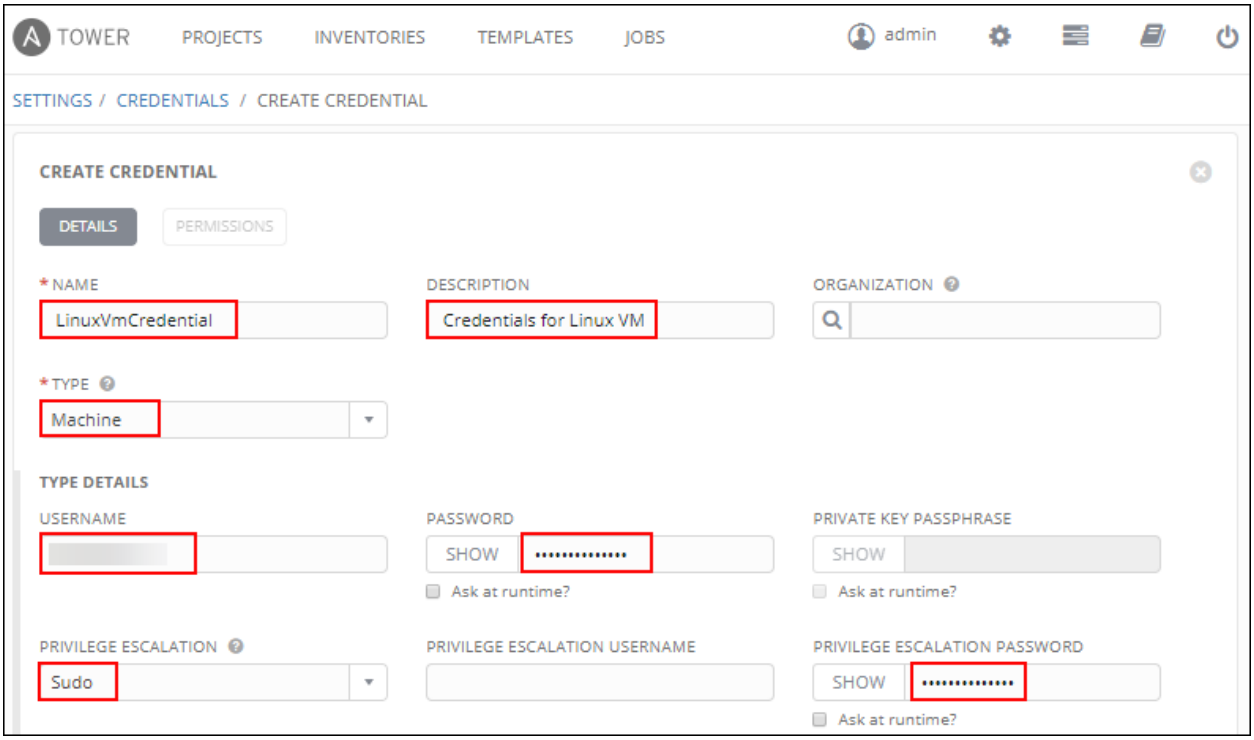
Tower credentials are imported and stored encrypted in Tower, and are not retrievable in plain text on the command line by any user. Once a password or key has been entered into the Tower interface, it is encrypted and inserted into the Tower database, and cannot be retrieved from Tower. We can grant users and teams the ability to use these credentials, without exposing the credential to the user. If we have a user move to a different team or leave the organization, you don't have to re-key all our systems just because that credential was available in Tower.

Now, we will navigate to Credentials, by clicking on the Setting ( ) button on the top right of the dashboard menu.

We will add new credentials in Ansible Tower for the Linux Virtual Machine as well as Windows Virtual Machine:

1. Click the + ADD button located in the upper right corner of the **Credentials** screen.
2. Enter the following details:
   - *Name*: **LinuxVmCredential**
   - *Description*: **Credentials for Linux VM**
   - *Organization*: **default**

- *Type*: **Machine**
- *Type details*:
    a. *Username:* (Enter the **vmUsername** for **VM** received via **email**)
    b. *Password:* (Enter the **vmPassword** for **VM** received via **email)**
    c. *Privilege escalation*: **SUDO**
    d. *Privilege escalation Password*: **(Provide the vmPassword for VM you received via email)**

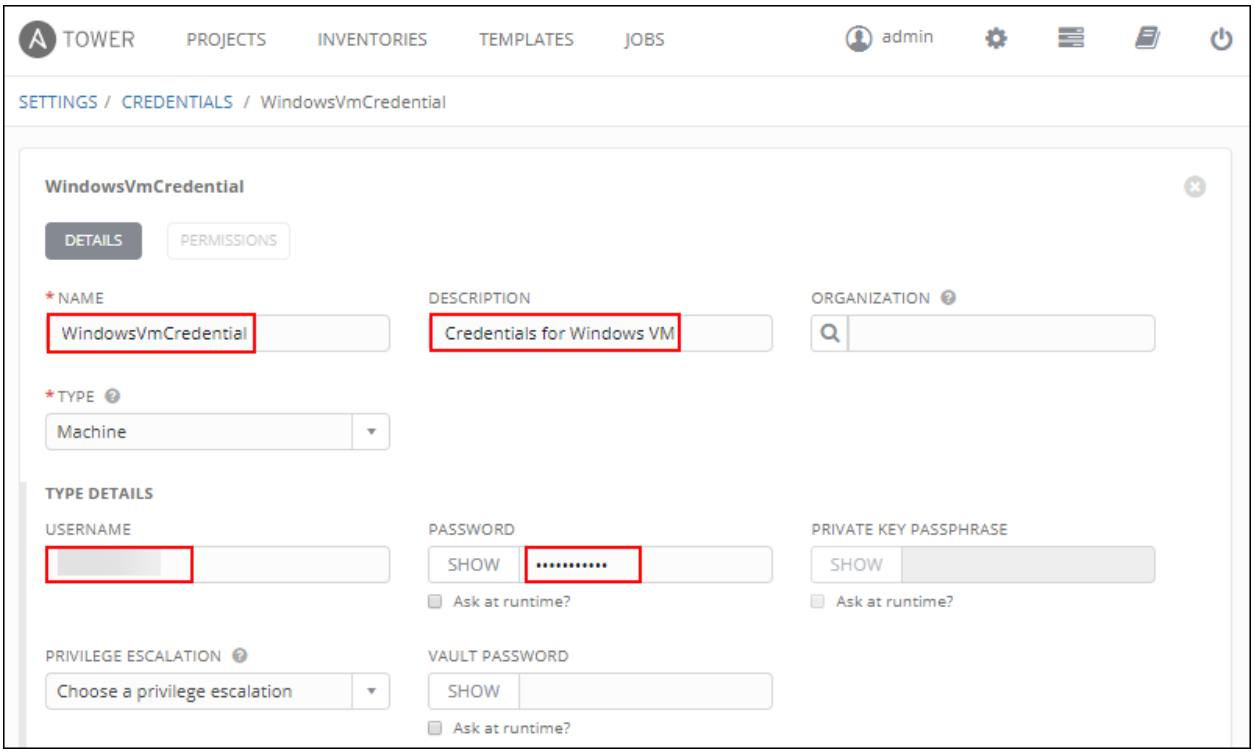


3. Click **Save** when done.

4. Again, click the + ADD button located in the upper right corner of the **Credentials** screen.

5. Enter the given details into the following fields:

- *Name*: **WindowsVmCredential**
- *Description*: **Credentials for Windows VM**
- *Organization*: **default**
- *Type*: **Machine**
- *Type details*:
    a. *Username:* (Enter the **vmUsername** for **VM** received via **email**)
    b. *Password:* (Enter the **vmPassword** for **VM** received via **email)**



6. Click **Save** when done.

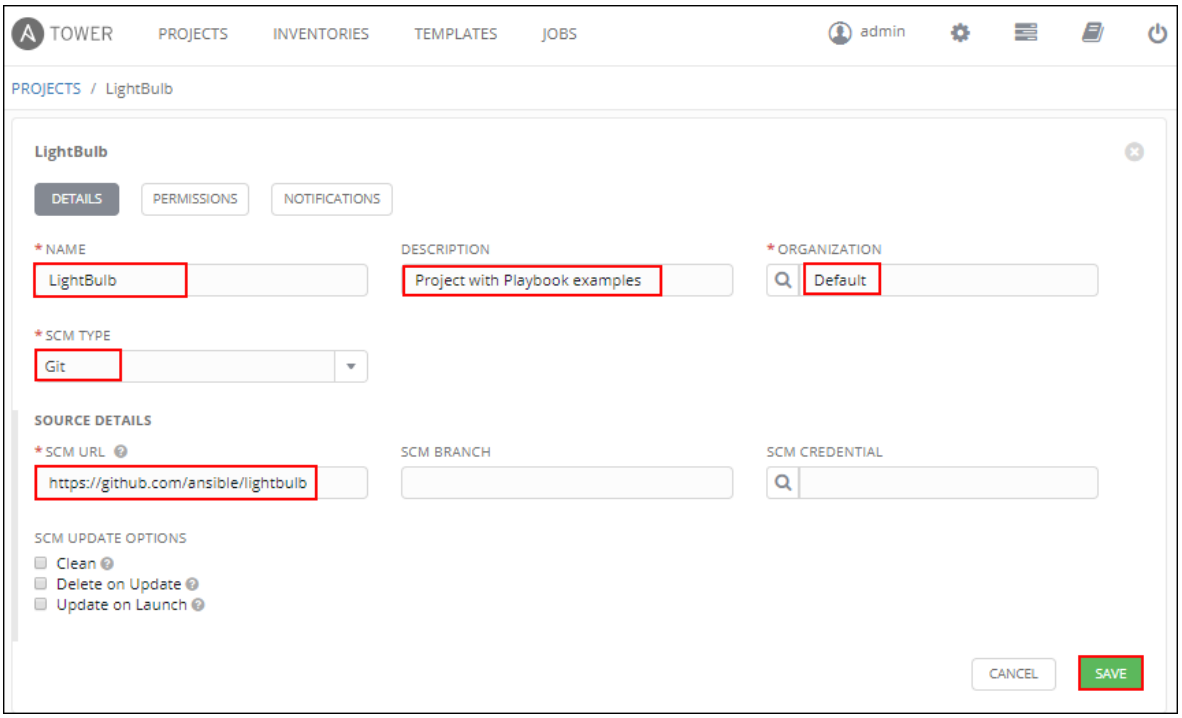
Add Projects

A Project is a logical collection of Ansible playbooks, represented in Tower.

We can manage playbooks and playbook directories by either placing them manually under the Project Base Path on your Tower server, or by placing your playbooks into a source code management (SCM) system supported by Tower, including Git, Subversion, and Mercurial.

Now you will create a new project for Linux Machines by clicking on PROJECTS on top of the dashboard menu:

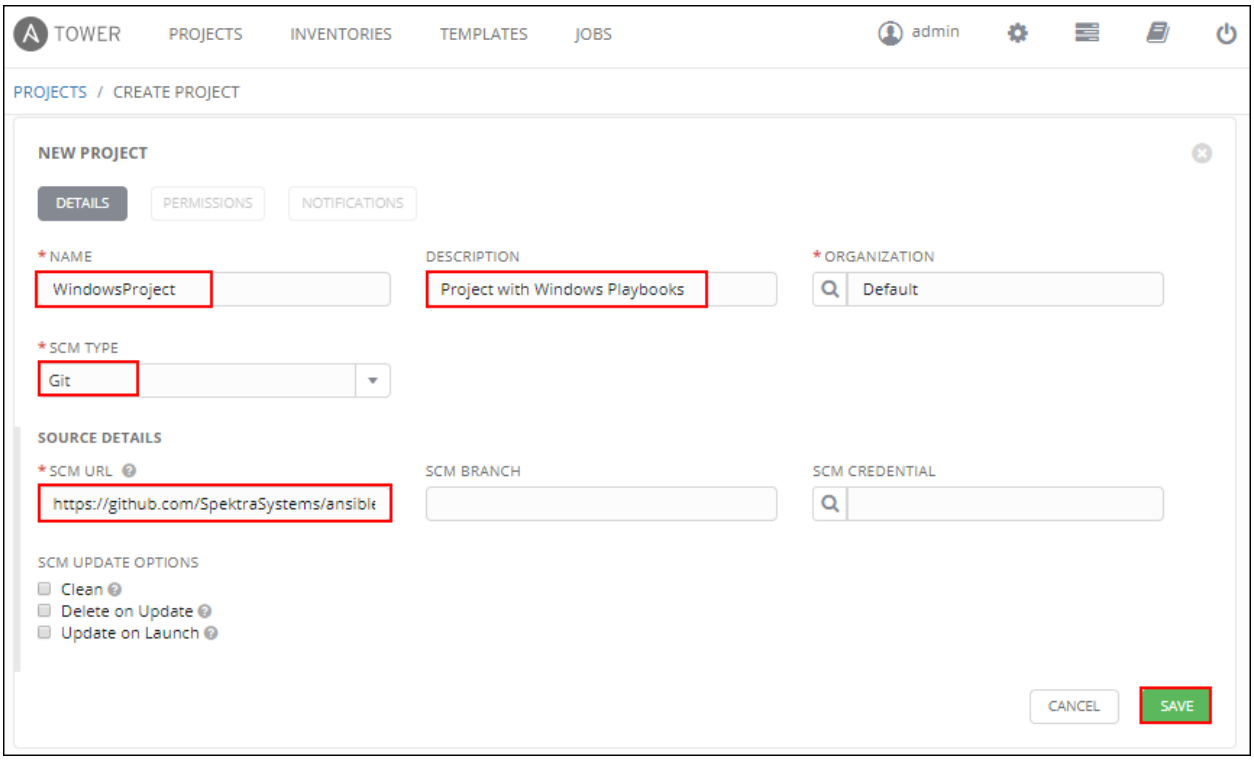
1. Then, click the + ADD button, which launches the **Create Project** dialog.
2. Enter the given details into the following fields:
   - *Name*: **LightBulb**
   - *Description*: **Project with Playbook examples**
   - *Organization*: **default**
   - *SCM Type*: **Git**
   - *Source details*:
     a. *SCM URL:* https://github.com/ansible/lightbulb



3. Click **Save** when done.

Now you will create another new project for Windows Machines by clicking on PROJECTS on top of the dashboard menu:

4. Again, click the + ADD button, which launches the **Create Project** dialog.
5. Enter the given details into the following fields:
    - *Name*: **WindowsProject**
    - *Description*: **Project with Windows Playbooks**
    - *Organization*: **default**
    - *SCM Type*: **Git**
    - *Source details*:
        a. *SCM URL:* https://github.com/SpektraSystems/ansible-testdrive



6. Click **Save** when done.

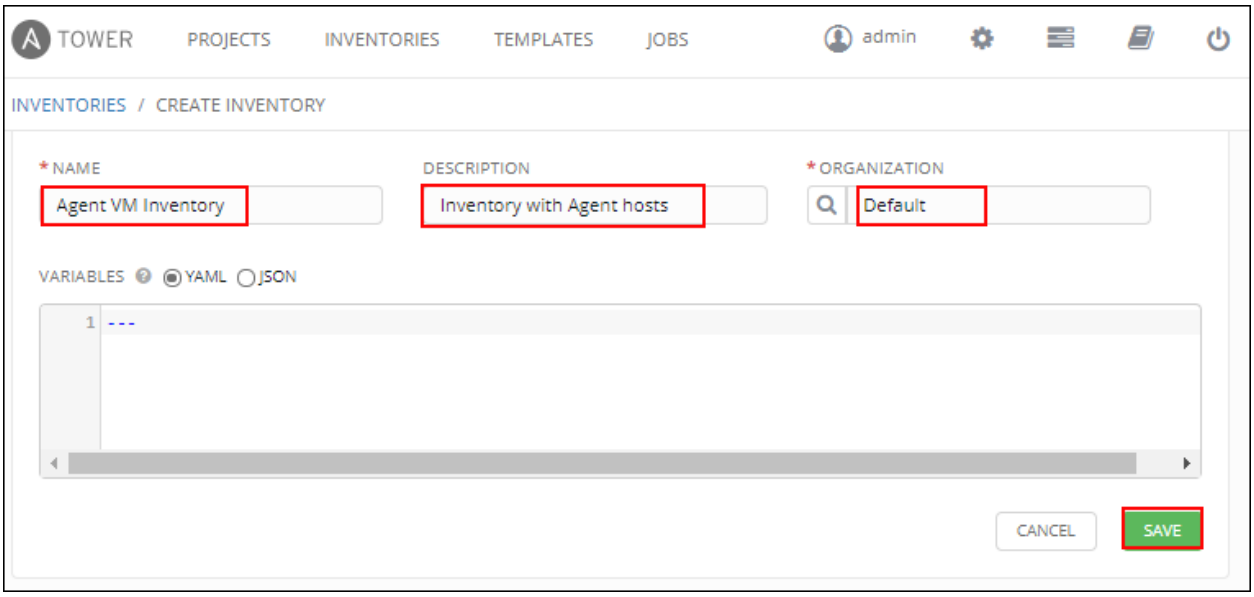
Add Inventory and Test Connectivity

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. Groups may be sourced manually, by entering host names into Tower, or from one of Ansible Tower's supported cloud providers.

Now, you will create a new Inventory and add Linux and Windows Machines as hosts by first clicking on INVENTORIES on top of the dashboard menu:

1. Then, click the + ADD button, which launches the **Create Inventory** dialog.
2. Enter the given details into the following fields:

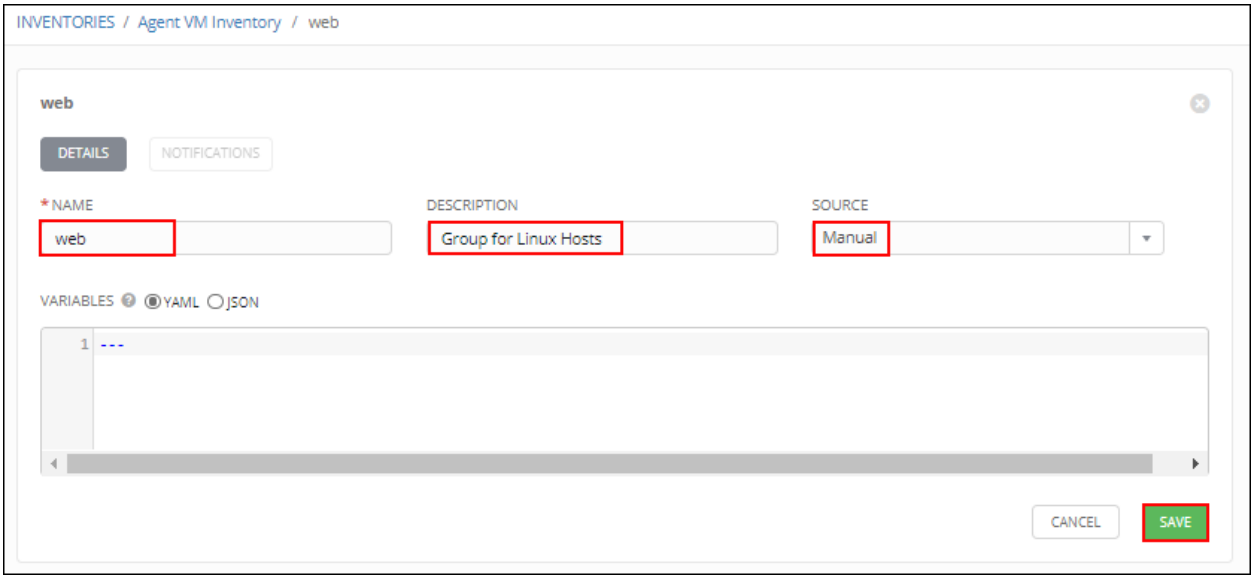
   - *Name*: **Agent VM Inventory**
   - *Description*: **Inventory with Agent hosts**
   - *Organization*: **Default**



3. Click on **Save** when done.

4. Now, click the + ADD button in GROUPS, which launches the **Create Group** dialog.
5. Enter the given details into the following fields:

   - *Name*: **web**
   - *Description*: **Group for Linux Hosts**
   - *Source*: **Manual**



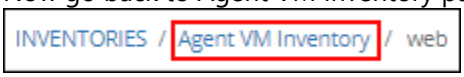
6. Click on **Save** when done.

7. Now, click on the created group **web**

8. Then, click the + ADD button in HOSTS, which launches the **Create Host** dialog.
9. Enter the given details into the following fields:

- *Host Name*: (Enter the **DNS Name** for **Linux VM** received via **email**)
- *Description*: **Linux Host**

10. Click on **Save** when done.
11. Now go back to Agent VM Inventory page by clicking on Agent VM Inventory

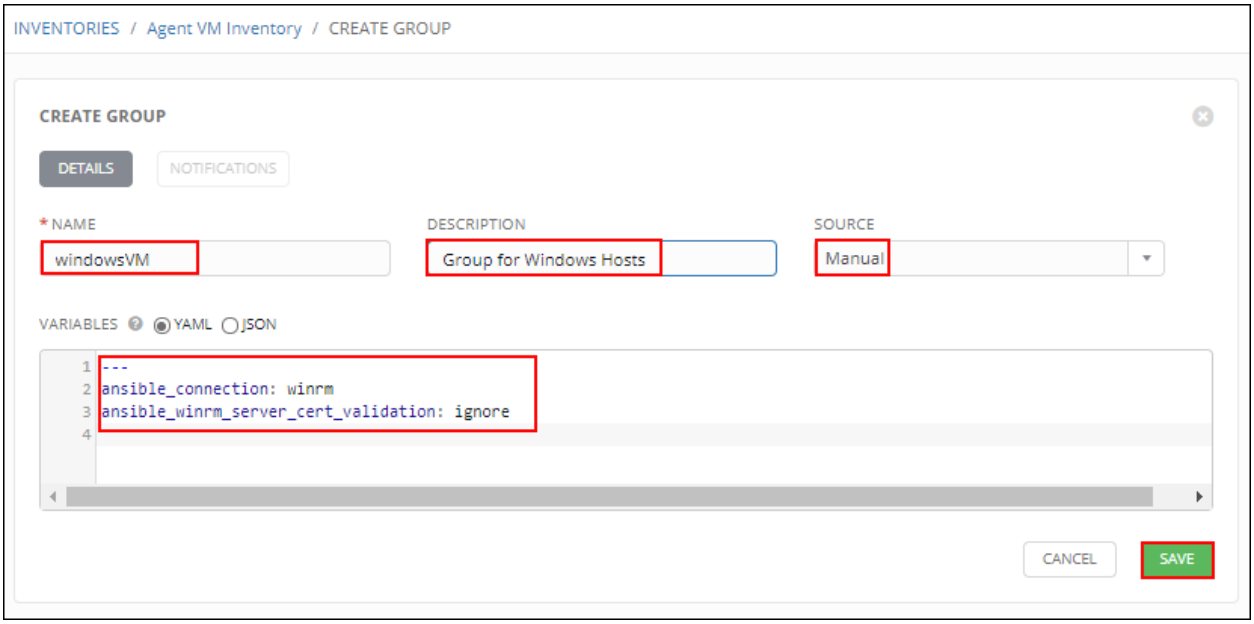


12. Now, click the 

button in GROUPS, which launches the **Create Group** dialog.
13. Enter the given details into the following fields:

- *Name*: **windowsVM**
- *Description*: **Group for Windows Hosts**
- *Source*: **Manual**
- *Variables:* Add the following lines below ---

  ansible_connection: winrm
  ansible_winrm_server_cert_validation: ignore

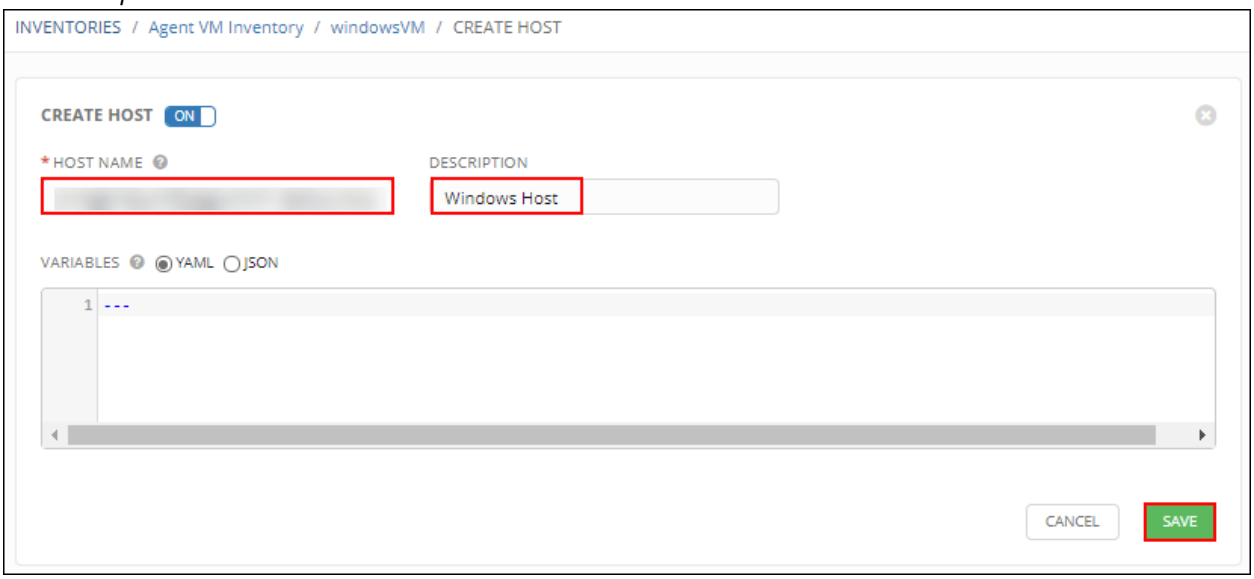


14. Click on **Save** when done.

15. Now, click on the created group **windowsVM**

16. Then, click the + ADD button in HOSTS, which launches the **Create Host** dialog.
17. Enter the given details into the following fields:

    - *Host Name*: (Enter the **DNS Name** for **Windows VM** received via **email**)
    - *Description*: **Windows Host**

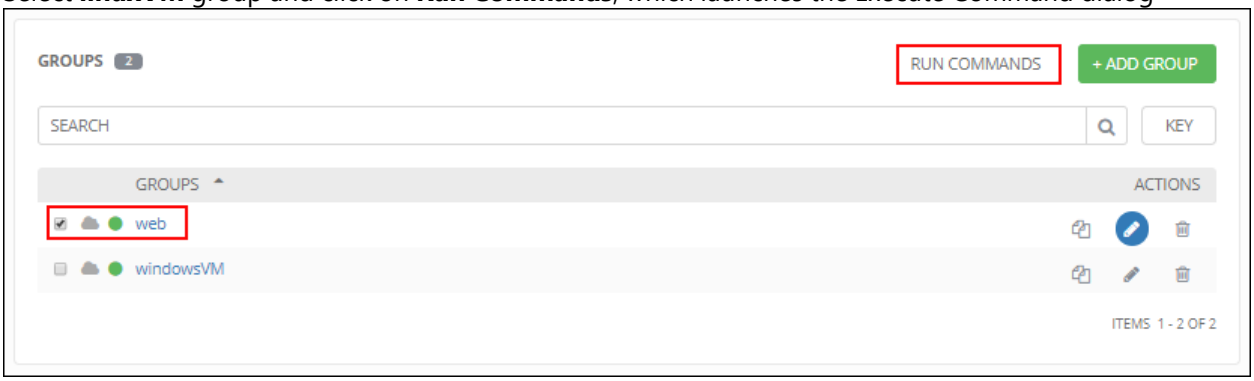


18. Click on **Save** when done.

Now we will test the connectivity to the added hosts.

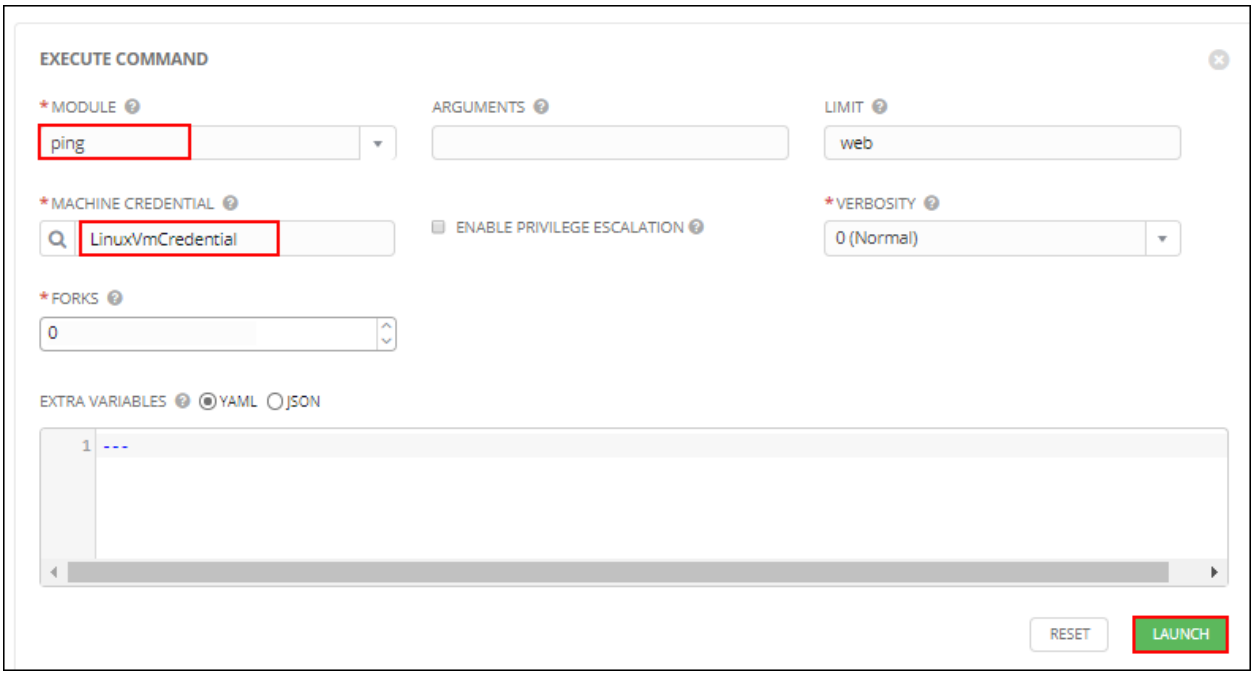
19. Now go back to Agent VM Inventory page by clicking on Agent VM Inventory

20. Select **linuxVM** group and click on **Run Commands**, which launches the Execute Command dialog

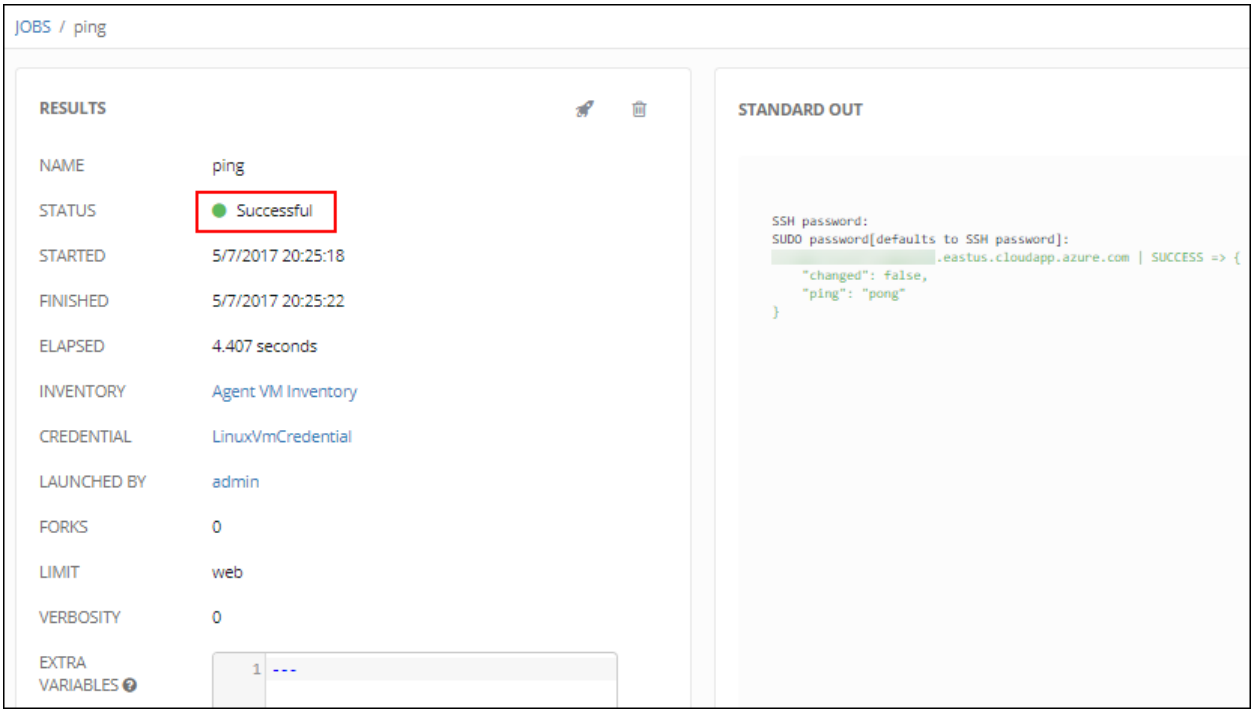


21. Enter the given details into the following fields:

- Module: **ping**
- *Machine Credential*: **LinuxVmCredential**

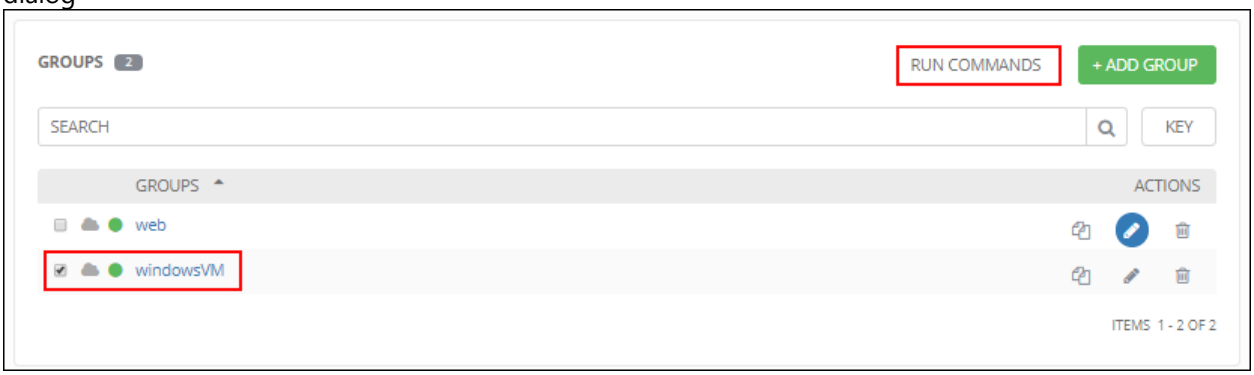


22. Click on **Launch** when done, which launches the Job Results page.



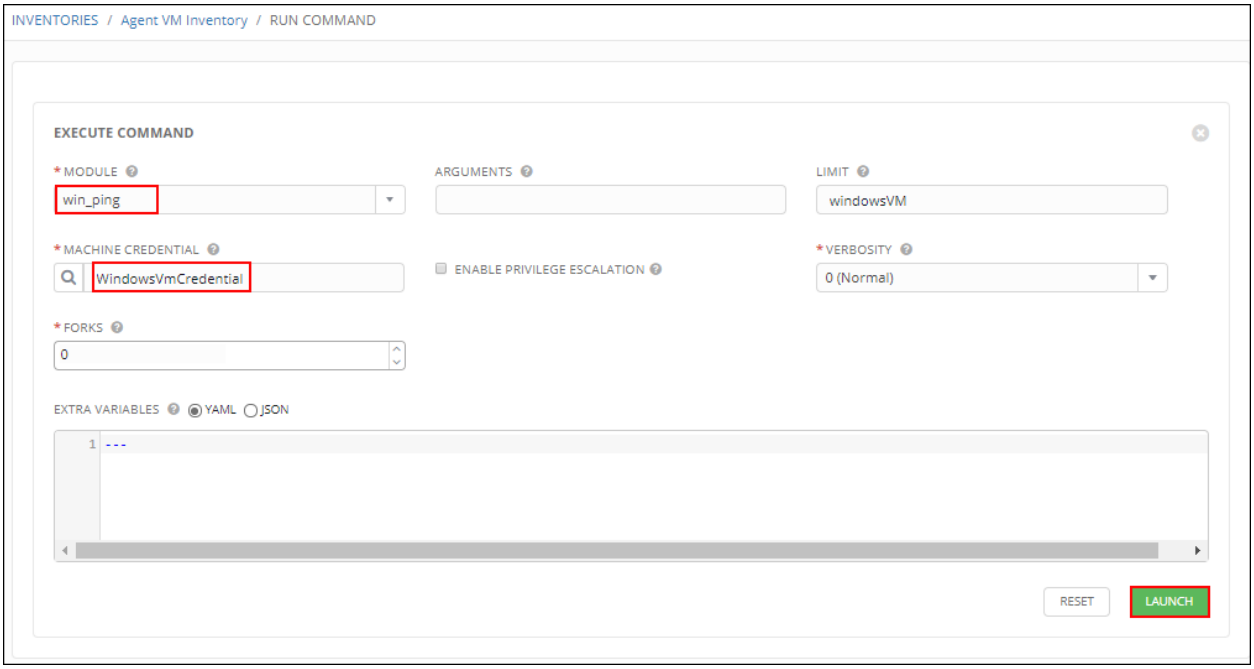
We can see status of the Job in that page, and if that job was successful, status will be **Successful** with a green icon next to it.

23. Now go back to Agent VM Inventory page by clicking on INVENTORIES from the top menu and then click on Agent VM Inventory.
24. Select **windowsVM** group and click on **Run Commands**, which launches the Execute Command dialog

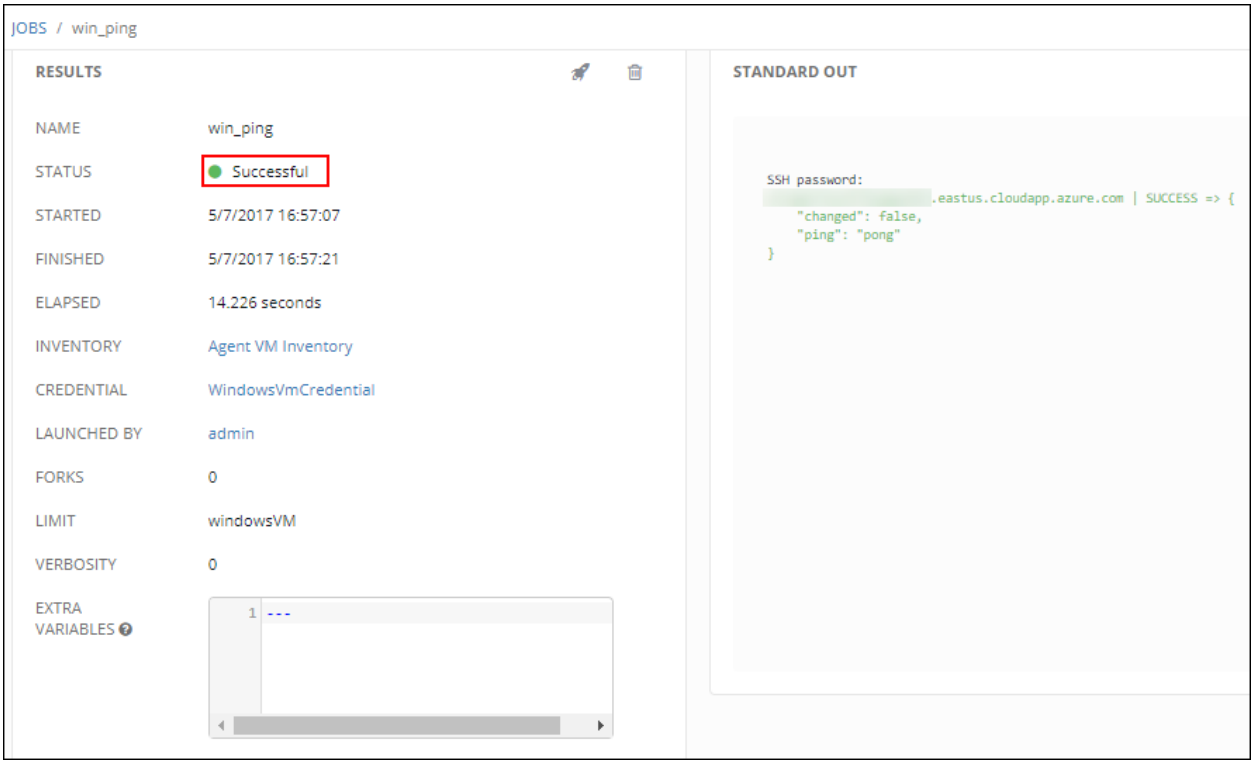


25. Enter the given details into the following fields:

- Module: Select **win_ping**
- *Machine Credential*: **WindowsVmCredential**



26. Click on **Launch** when done, which launches the Job Results page.



We can see status of the Job in that page, and if that job was successful, status will be **Successful** with a green icon next to it.

END OF LAB

Lab 2: Deploy Web Servers on Linux VM using Ansible Playbook

Deploy Nginx Web Server

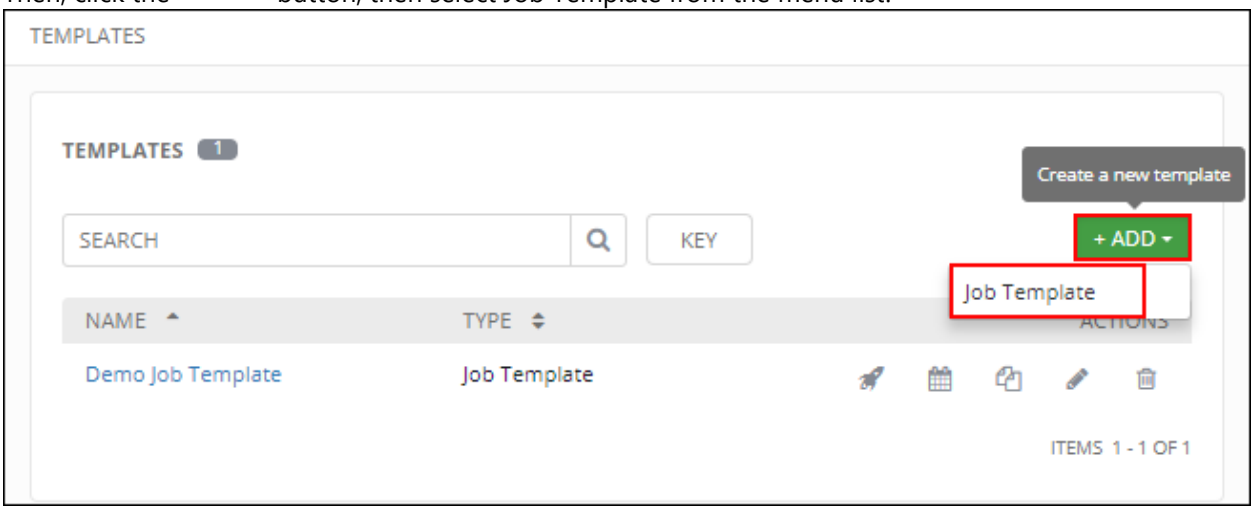
A job template is a definition and set of parameters for running an Ansible job. Job templates are useful to execute the same job many times. Job templates also encourage the reuse of Ansible playbook content and collaboration between teams. While the REST API allows for the execution of jobs directly, Tower requires that you first create a job template.

We can access Job Template page by clicking on TEMPLATES on the menu on top of the dashboard.

This menu opens a list of the job templates that are currently available. The job template list may be sorted and searched by **Name** or **Description**. The **Templates** tab also enables the user to launch, schedule, modify, and remove a job template.

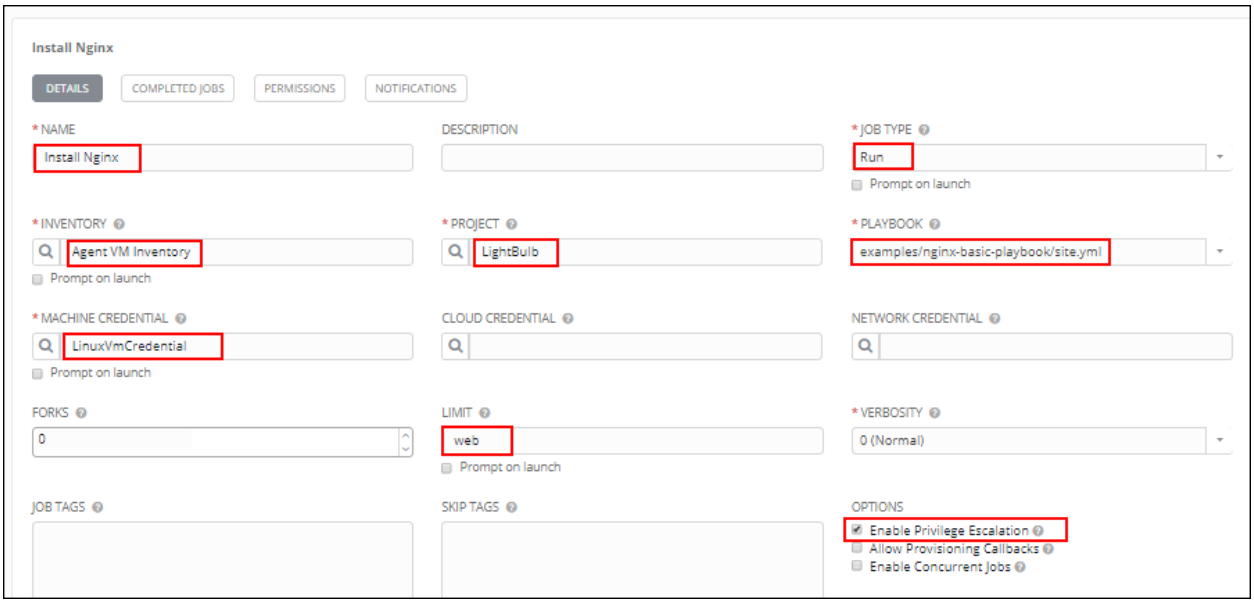
Now, you will create a new template for deploying Nginx Web Server by first clicking on TEMPLATES on top of the dashboard menu:

1. Then, click the + ADD button, then select Job Template from the menu list.



2. Enter the given details into the following fields:

- *NAME:* **Install Nginx**
- *JOB TYPE:* **Run**
- *INVENTORY:* **Agent VM Inventory**
- *PROJECT:* **LighBulb**
- *PLAYBOOK:* **examples/nginx-basic-playbook/site.yml**
- *MACHINE CREDENTIAL:* **LinuxVmCredential**
- *LIMIT:* **web**
- *Enable Privilege Escalation:* **Tick**

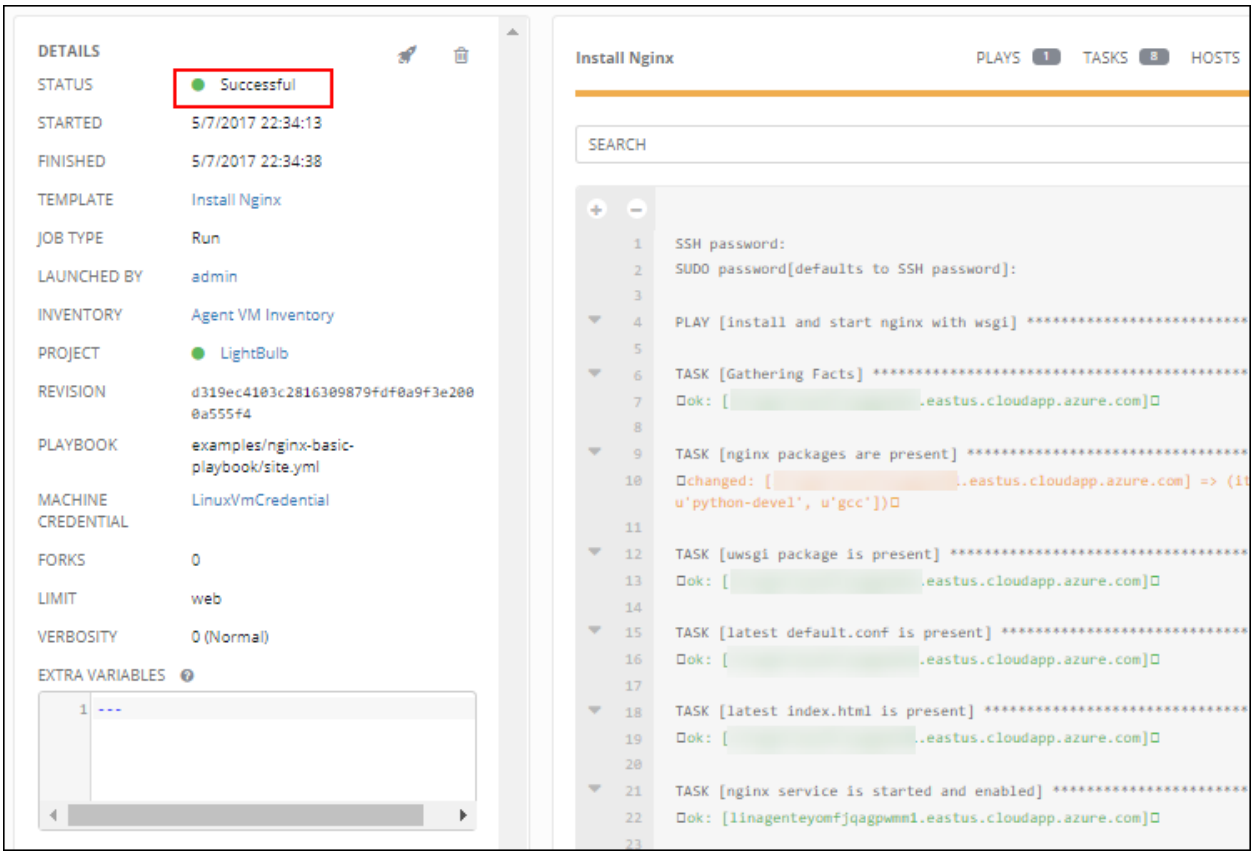


3. Click on **Save** when done.

Saving the template does not exit the job template page but remains on the Job Template Details view for further editing, if necessary. The **Details** tab of a saved job allows you to review, edit, and add a survey (if the job type is not a scan).

You can verify the template is saved when the newly created template appears on the list of templates at the bottom of the screen.

Now, we will launch a Job Template by clicking the launch icon. We will be directed to the Job Results Page. The Jobs page shows details of all the tasks and events for that playbook run.



We can see the status as successful for Job to deploy Nginx Web Server in Linux VM.

Now to verify that apache is installed on Linux Virtual Machine and is accessible through Public DNS Name, we will open a browser and navigate to the Linux VM DNS name and check if the Ansible Tower is coming on the page as shown below.

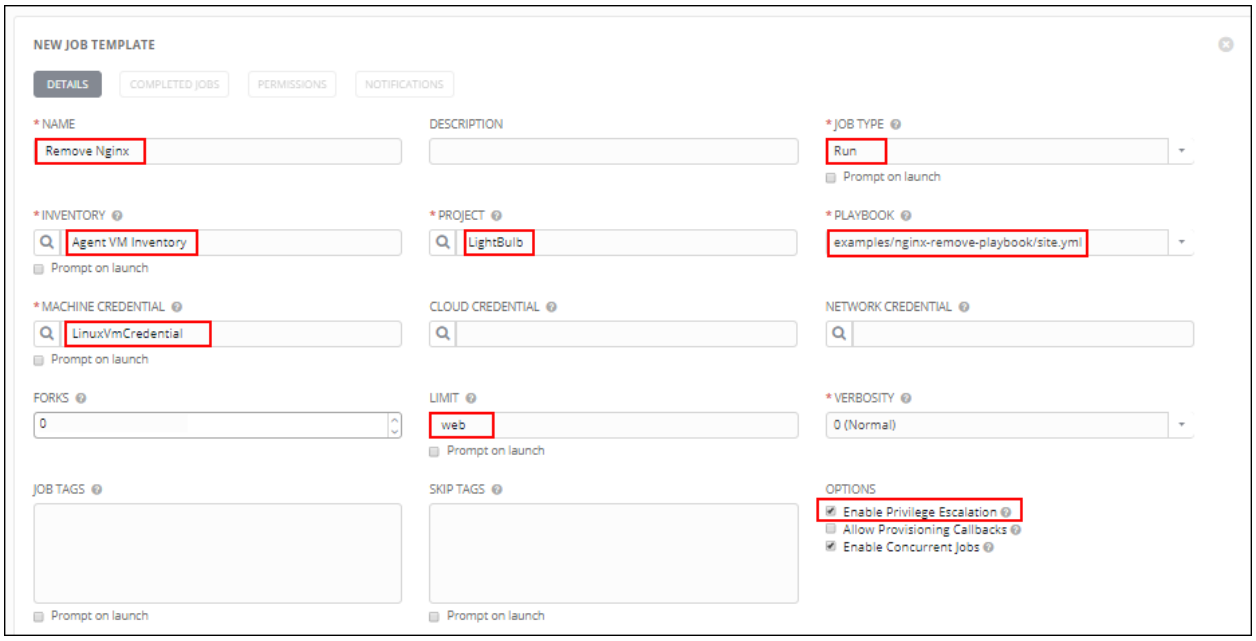
Remove Nginx Web Server

Now, you will create a new template for removing Nginx Web Server by first clicking on TEMPLATES on top of the dashboard menu:

1. Then, click the + ADD button, then select Job Template from the menu list.

2. Enter the given details into the following fields:

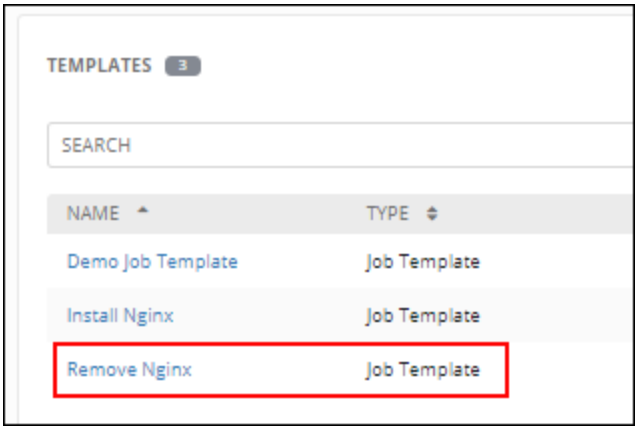
   - *NAME:* **Remove Nginx**
   - *JOB TYPE:* **Run**
   - *INVENTORY:* **Agent VM Inventory**
   - *PROJECT:* **LighBulb**
   - *PLAYBOOK:* **examples/nginx-remove-playbook/site.yml**
   - *MACHINE CREDENTIAL:* **LinuxVmCredential**
   - *LIMIT:* **web**
   - *Enable Privilege Escalation:* **Tick**



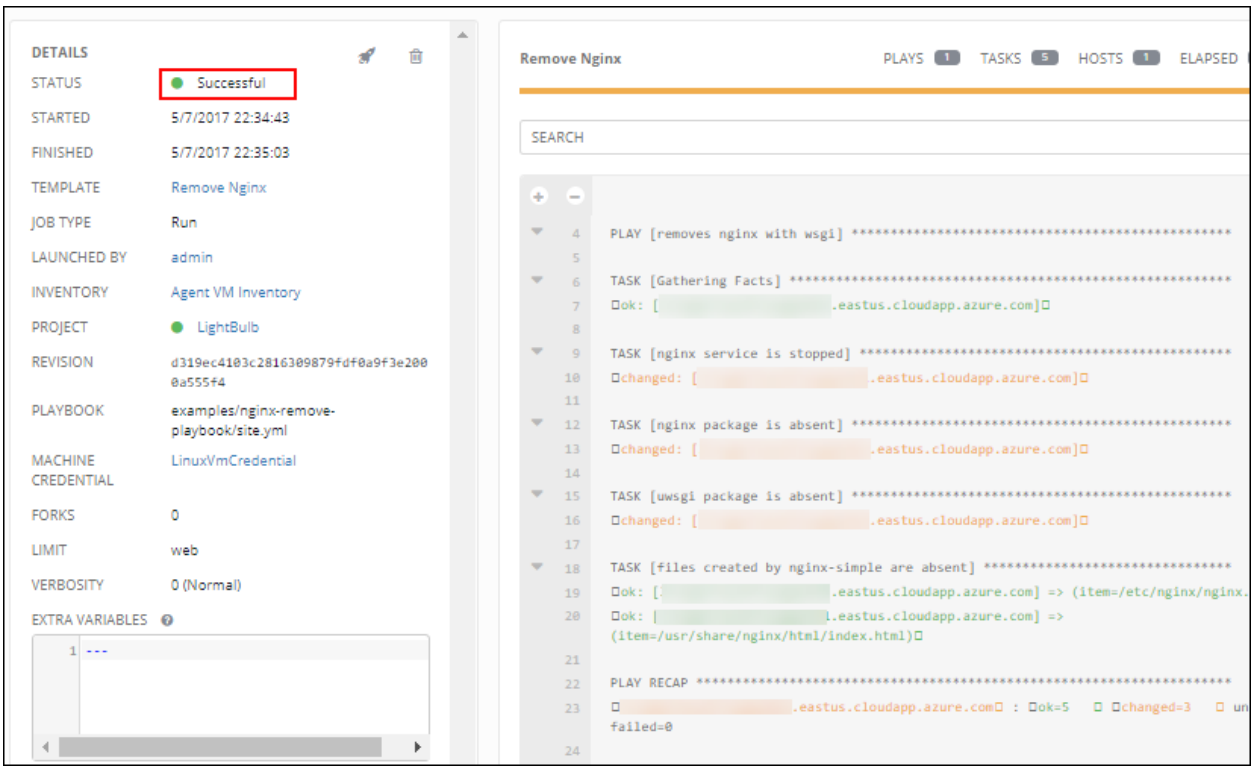
3. Click on **Save** when done.

Saving the template does not exit the job template page but remains on the Job Template Details view for further editing, if necessary. The **Details** tab of a saved job allows you to review, edit, and add a survey (if the job type is not a scan).

You can verify the template is saved when the newly created template appears on the list of templates at the bottom of the screen.

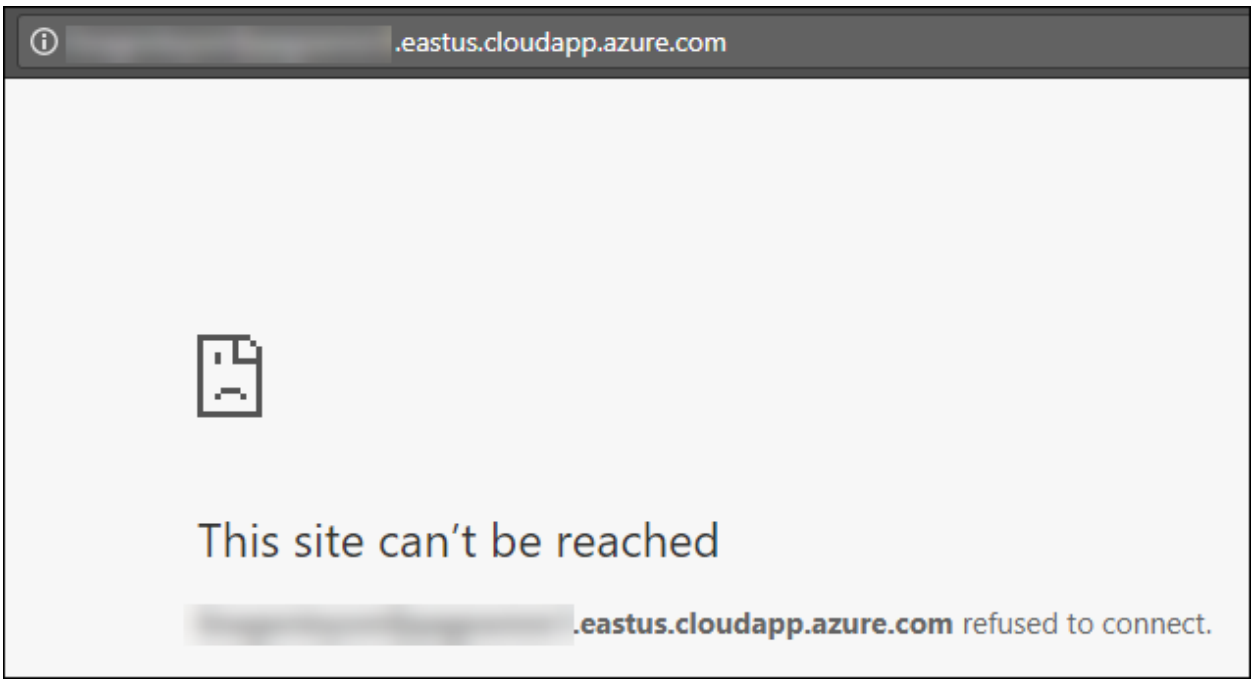


Now, we will launch a Job Template by clicking the launch icon. We will be directed to the Job Results Page. The Jobs page shows details of all the tasks and events for that playbook run.



We can see the status as successful for Job to deploy Nginx Web Server in Linux VM.

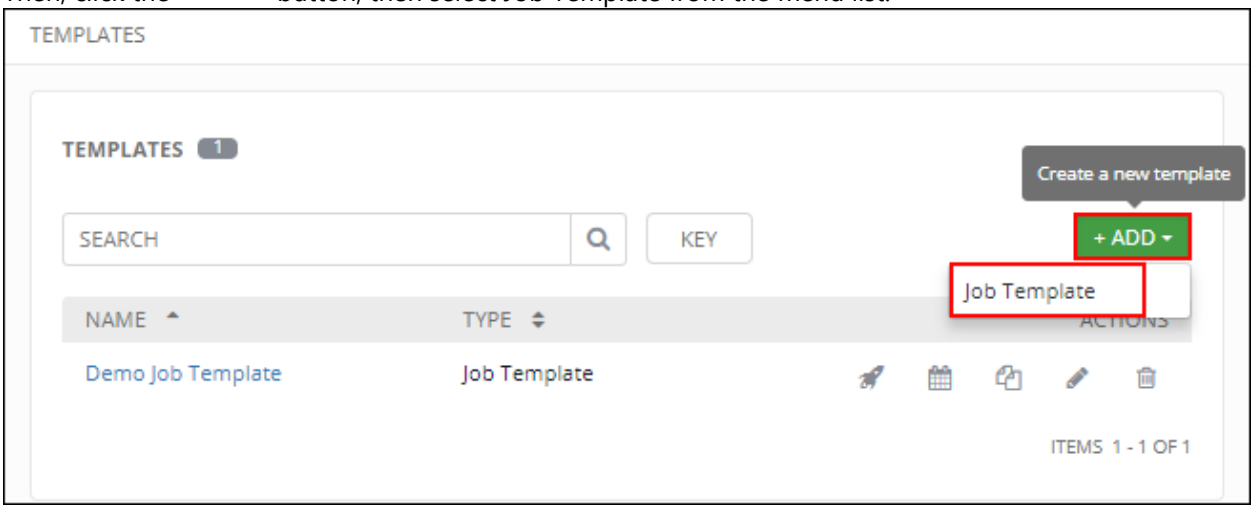
Now to verify that Nginx Webserver is removed from the Linux Virtual Machine and is not accessible through Public DNS Name, we will open a browser and navigate to the Linux VM DNS name and check if the page is unreachable



Deploy Apache Web Server

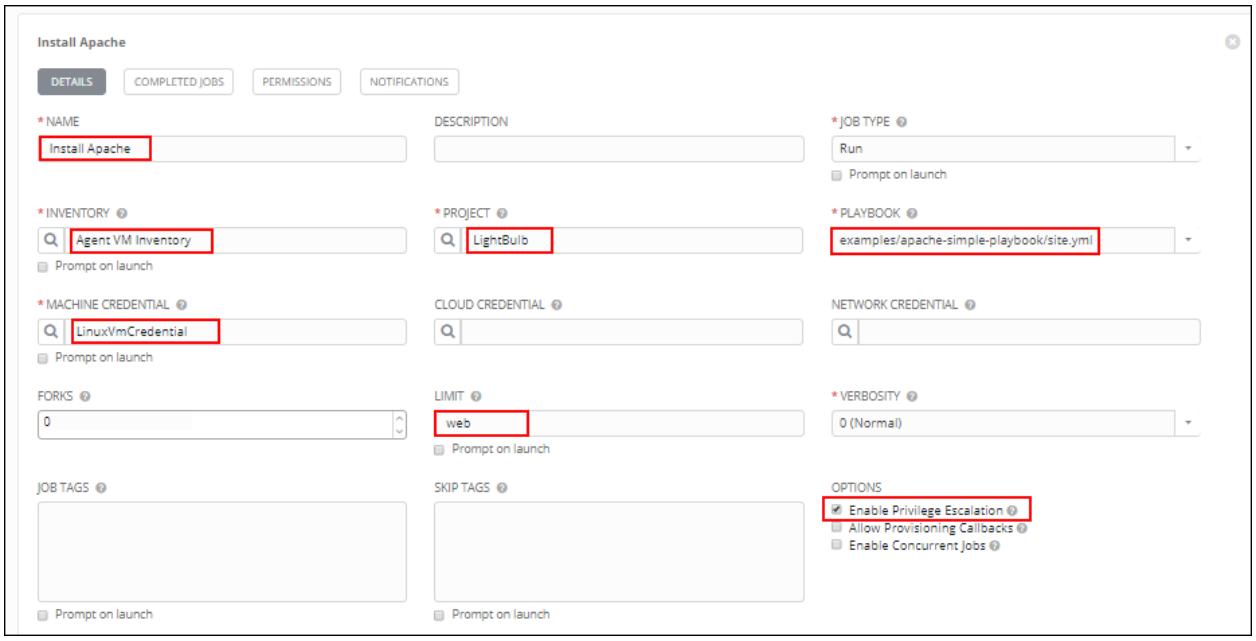
Now, you will create a new template for deploying Apache Web Server by first clicking on TEMPLATES on top of the dashboard menu:

1. Then, click the + ADD button, then select Job Template from the menu list.



2. Enter the given details into the following fields:

- *NAME:* **Install Apache**
- *JOB TYPE:* **Run**
- *INVENTORY:* **Agent VM Inventory**
- *PROJECT:* **LighBulb**
- *PLAYBOOK:* **examples/apache-simple-playbook/site.yml**
- *MACHINE CREDENTIAL:* **LinuxVmCredential**
- *LIMIT:* **web**
- *Enable Privilege Escalation:* **Tick**

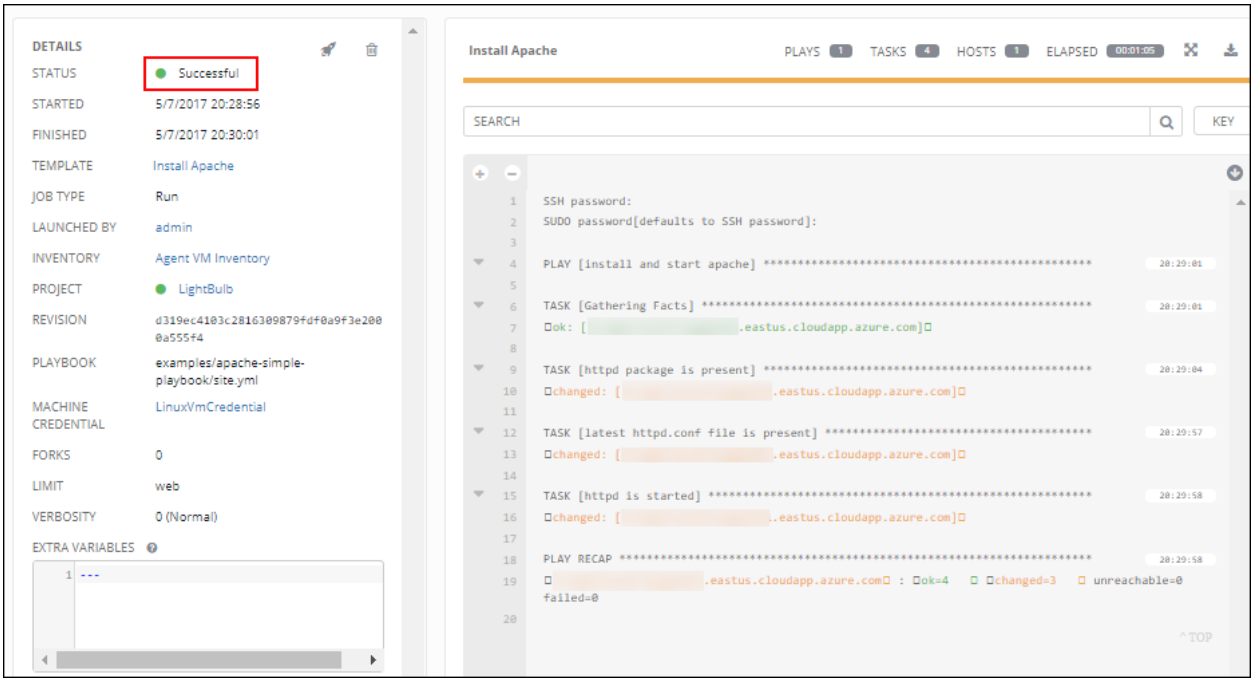


3. Click on **Save** when done.

Saving the template does not exit the job template page but remains on the Job Template Details view for further editing, if necessary. The **Details** tab of a saved job allows you to review, edit, and add a survey (if the job type is not a scan).

You can verify the template is saved when the newly created template appears on the list of templates at the bottom of the screen.

Now, we will launch a Job Template by clicking the launch icon. We will be directed to the Job Results Page. The Jobs page shows details of all the tasks and events for that playbook run.



We can see the status as successful for Job to deploy Nginx Web Server in Linux VM.

Now to verify that apache is installed on Linux Virtual Machine and is accessible through Public DNS Name, we will open a browser and navigate to the Linux VM DNS name and check if the Ansible Tower is coming on the page as shown below.

END OF LAB

Lab 3: Manage Windows VM using Ansible Playbook

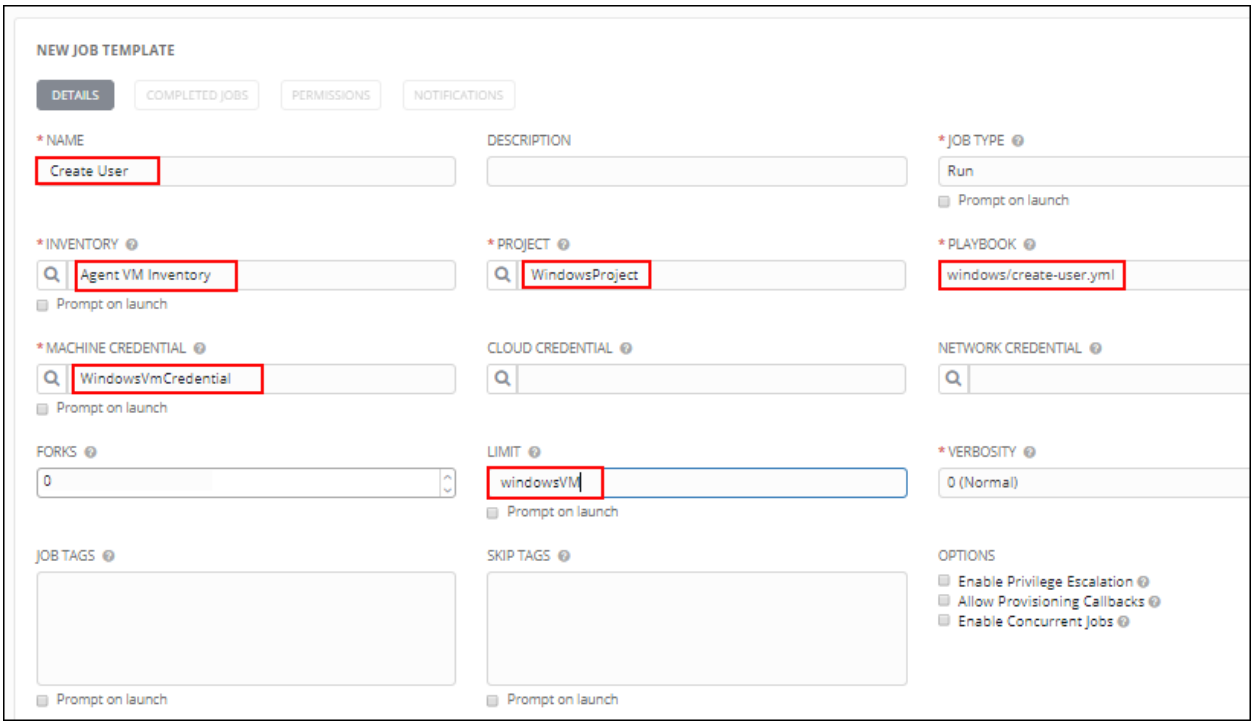
Create a Local User Account in Windows VM

Now, you will create a new template for creating a local User account in Windows VM by first clicking on TEMPLATES on top of the dashboard menu:

1. Then, click the + ADD button, then select Job Template from the menu list.

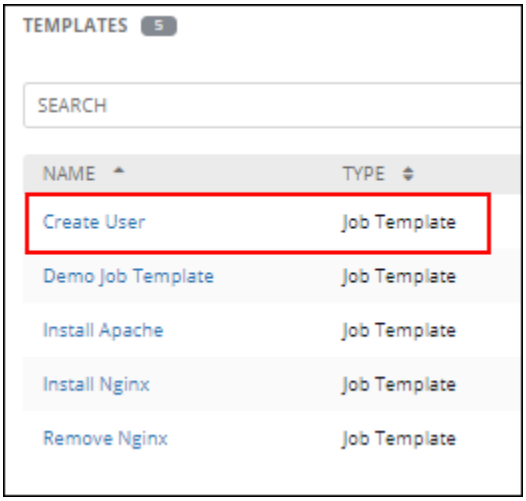
2. Enter the given details into the following fields:

   - *NAME:* **Create User**
   - *JOB TYPE:* **Run**
   - *INVENTORY:* **Agent VM Inventory**
   - *PROJECT:* **WindowsProject**
   - *PLAYBOOK:* **windows/create-user.yml**
   - *MACHINE CREDENTIAL:* **WindowsVmCredential**
   - *LIMIT:* **windowsVM**

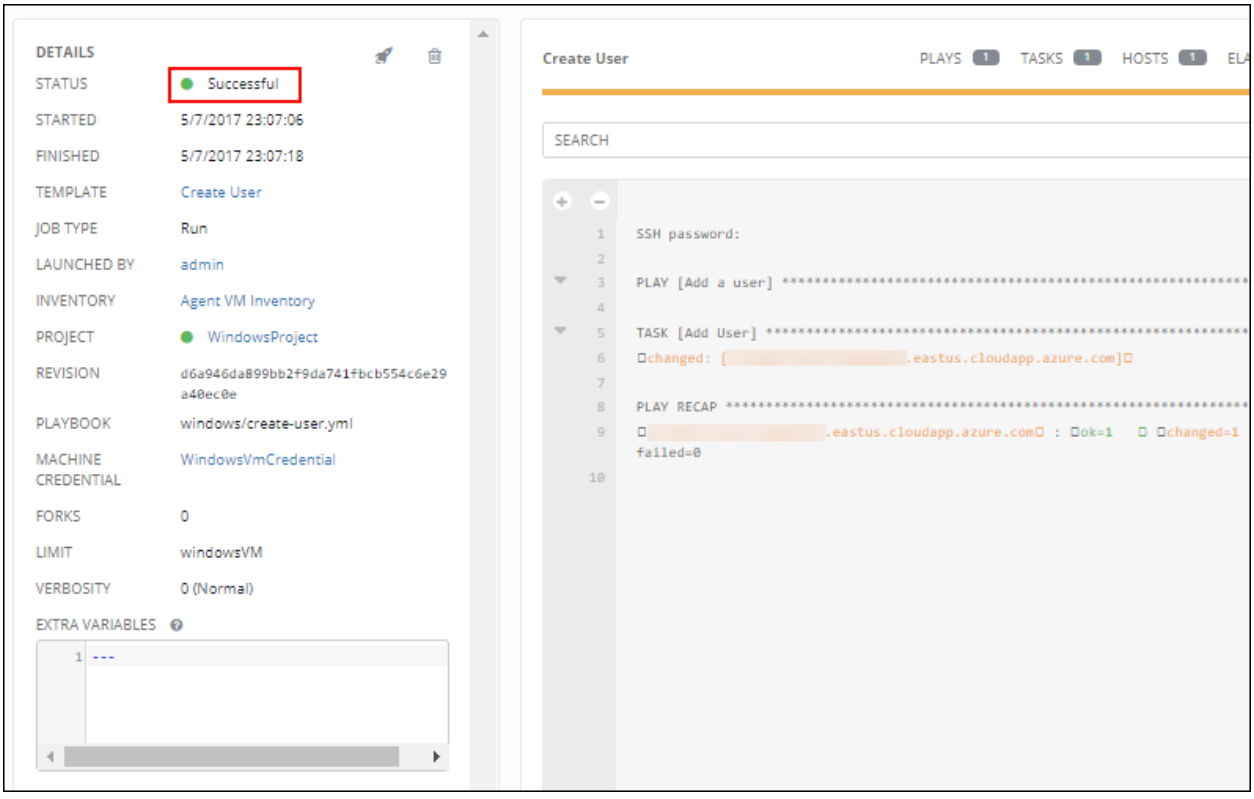


3. Click on **Save** when done.

You can verify the template is saved when the newly created template appears on the list of templates at the bottom of the screen.



Now, we will launch a Job Template by clicking the launch icon. We will be directed to the Job Results Page. The Jobs page shows details of all the tasks and events for that playbook run.



We can see the status as successful for Job to create a local user account in Windows VM.

Now to verify that user account is created on Windows Virtual Machine, we will access the Virtual Machine and check.

<TO DO>

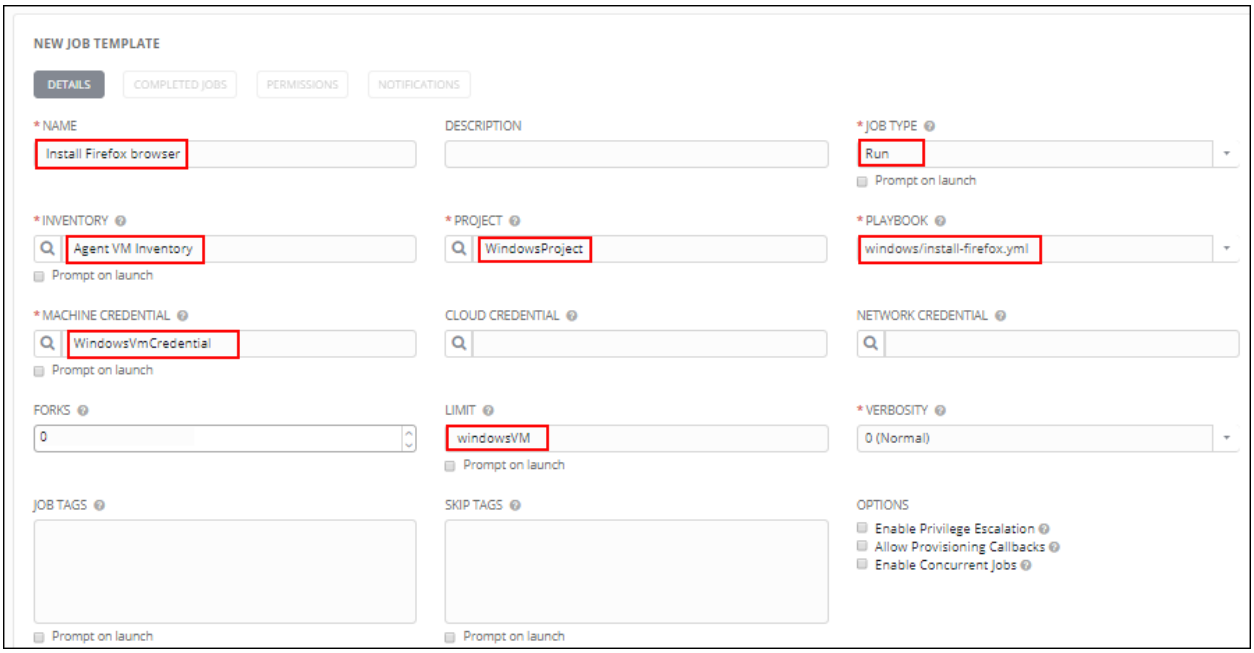
Install a Package

Now, you will create a new template for installing Firefox Browser by first clicking on TEMPLATES on top of the dashboard menu:

1. Then, click the + ADD button, then select Job Template from the menu list.

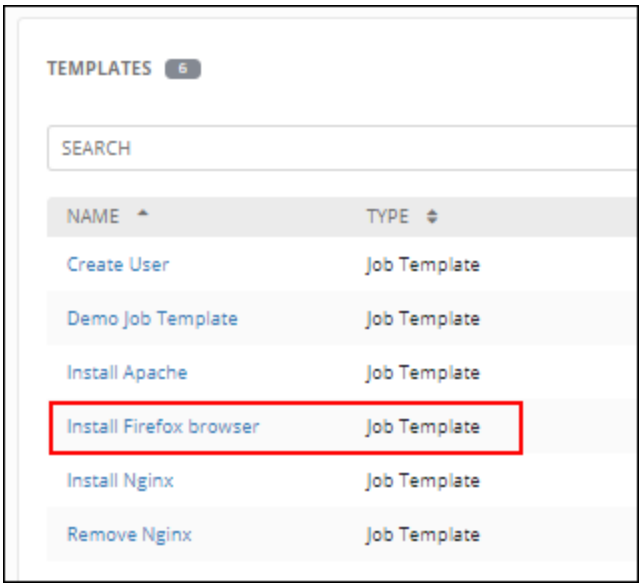
2. Enter the given details into the following fields:

   - *NAME:* **Install Firefox browser**
   - *JOB TYPE:* **Run**
   - *INVENTORY:* **Agent VM Inventory**
   - *PROJECT:* **WindowsProject**
   - *PLAYBOOK:* **windows/install-firefox.yml**
   - *MACHINE CREDENTIAL:* **WindowsVmCredential**
   - *LIMIT:* **windowsVM**

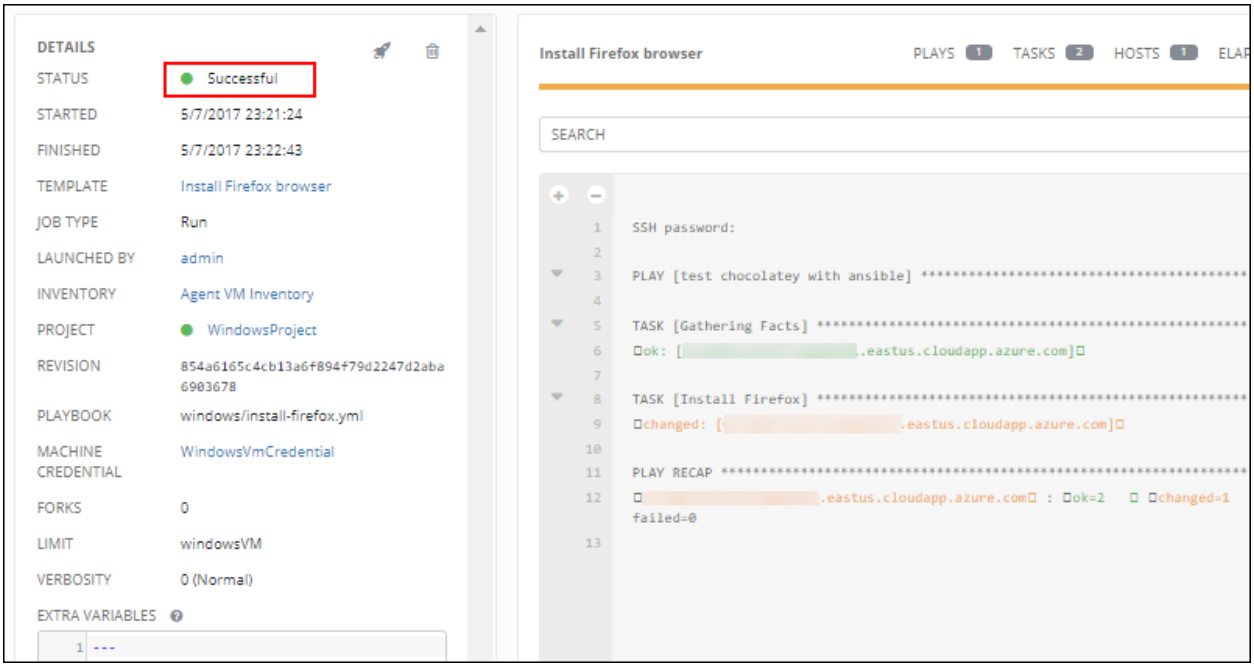


3. Click on **Save** when done.

You can verify the template is saved when the newly created template appears on the list of templates at the bottom of the screen.



Now, we will launch a Job Template by clicking the launch icon. We will be directed to the Job Results Page. The Jobs page shows details of all the tasks and events for that playbook run.



We can see the status as successful for Job to install Firefox Browser on the windows virtual machine.

Now to verify that Firefox is installed on the Windows Virtual Machine, we will log in to the VM.

<TO DO>

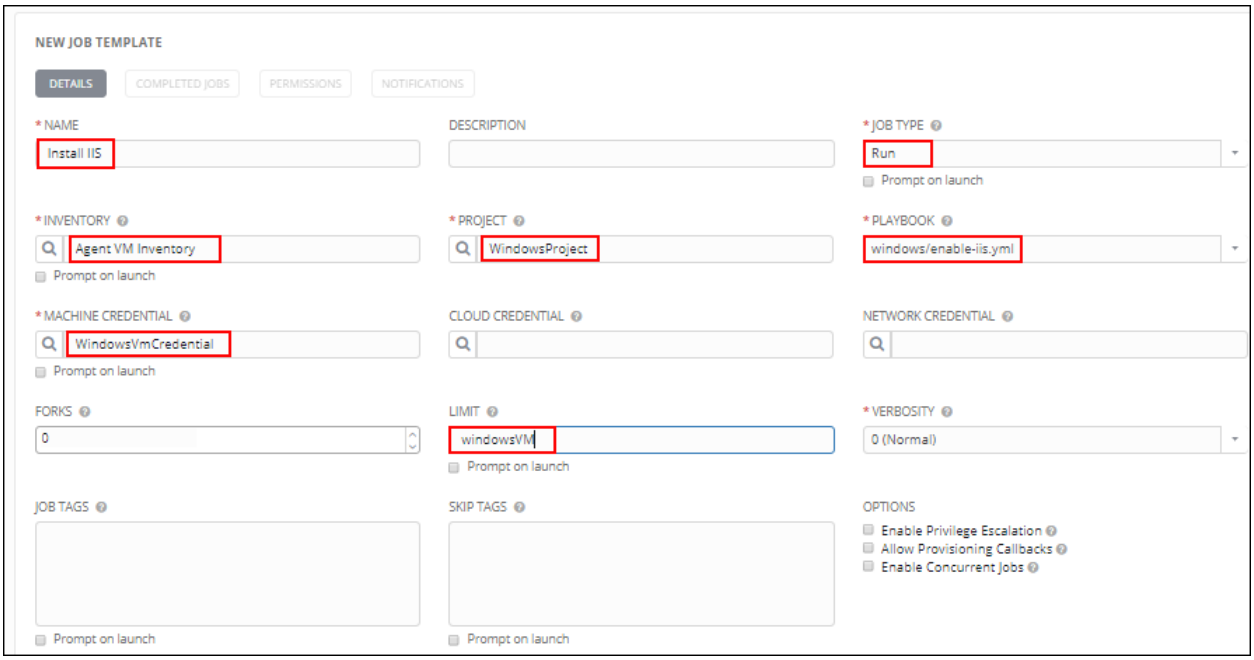
Deploy IIS Web Server

Now, we will create a new template for deploying IIS Web Server by first clicking on TEMPLATES on top of the dashboard menu:

1. Then, click the + ADD button, then select Job Template from the menu list.

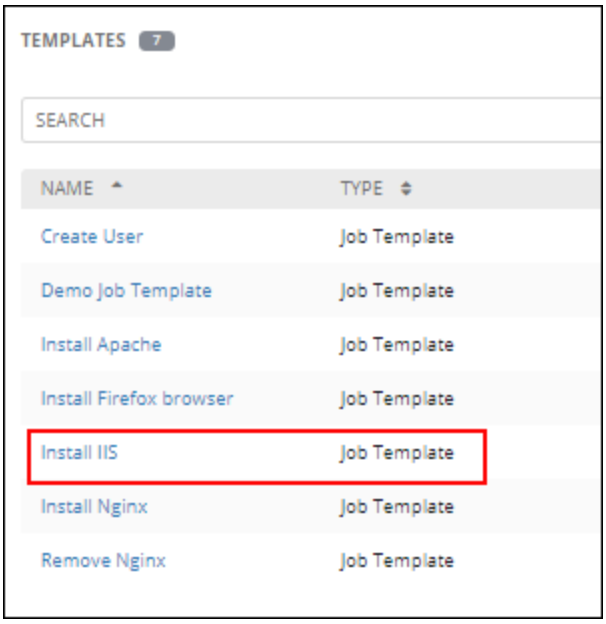
2. Enter the given details into the following fields:

   - *NAME:* **Install IIS**
   - *JOB TYPE:* **Run**
   - *INVENTORY:* **Agent VM Inventory**
   - *PROJECT:* **WindowsProject**
   - *PLAYBOOK:* **windows/enable -iis.yml**
   - *MACHINE CREDENTIAL:* **WindowsVmCredential**
   - *LIMIT:* **windowsVM**

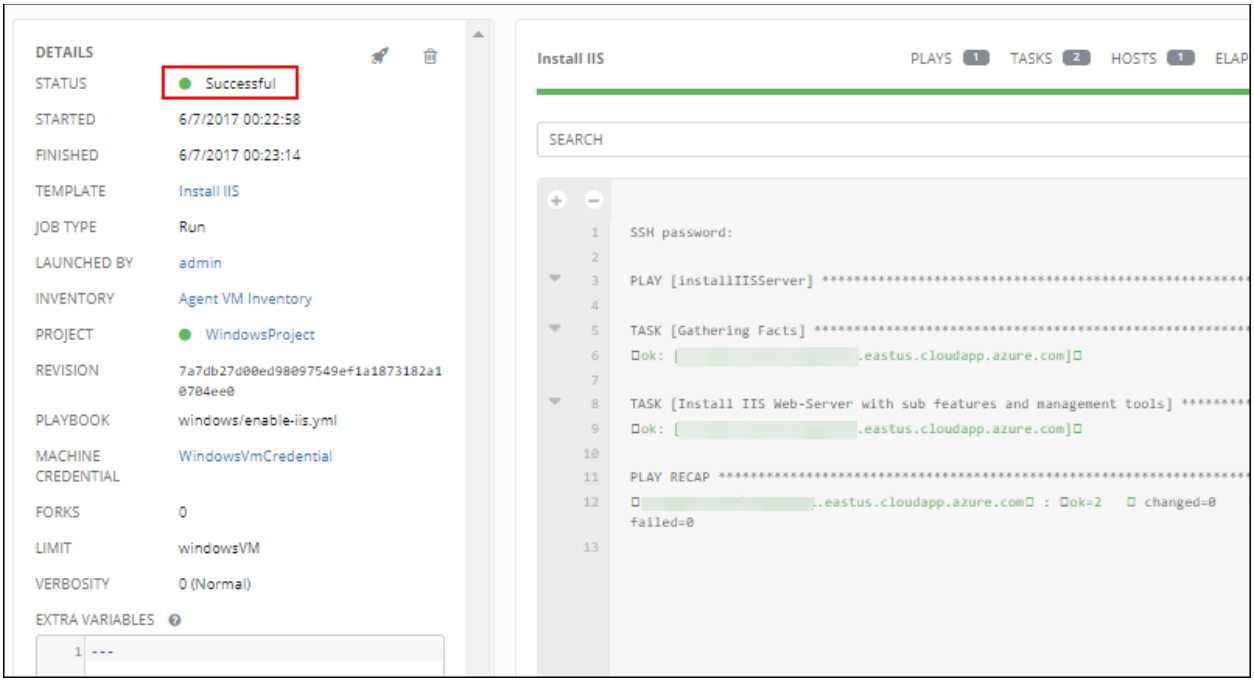


3. Click on **Save** when done.

You can verify the template is saved when the newly created template appears on the list of templates at the bottom of the screen.

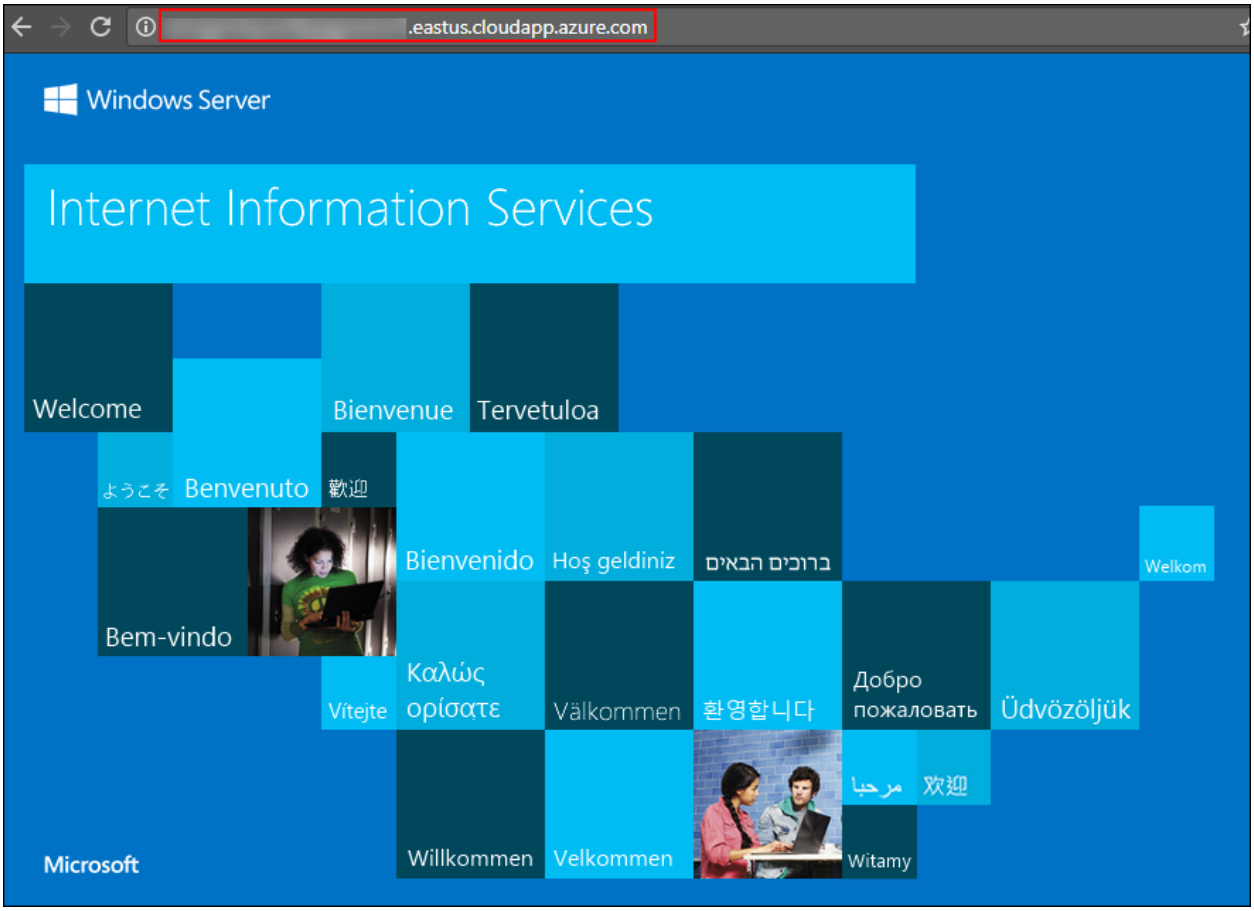


Now, we will launch a Job Template by clicking the launch icon. We will be directed to the Job Results Page. The Jobs page shows details of all the tasks and events for that playbook run.



We can see the status as successful for Job to deploy IIS Web Server in Linux VM.

Now to verify that apache is installed on Windows Virtual Machine and is accessible through Public DNS Name, we will open a browser and navigate to the Windows VM DNS name and check if the IIS Server default page is coming up as shown below.



END OF LAB

Thank You for following the test drive.

*****