



ANSIBLE
TOWER
by Red Hat®

Contents

Introduction to Ansible and Ansible Tower	3
Ansible Tower Dashboard	3
Enable Self-Service IT	4
Job Scheduling and Multi-Playbook Workflows.....	4
Manage and Track Your Entire Inventory	5
Integrate Tower Using the REST API and CLI Tool	6
About the Test Drive	7
Deployment Architecture	8
Getting Started.....	9
Lab 1: Getting Started with Ansible Tower.....	11
Access Ansible tower.....	11
Request for trial license	11
Assign License	14
Tower Dashboard	14
Add Credentials	15
Add Projects	17
Add Inventory and Test Connectivity.....	19
Lab 2: Deploy Web Servers on Linux VM using Ansible Playbook	26
Deploy Nginx Web Server	26
Remove Nginx Web Server	29
Deploy Apache Web Server	32
Lab 3: Manage Windows VM using Ansible Playbook	35
Create a Local User Account in Windows VM	35
Install a Package	40
Deploy IIS Web Server	44

Introduction to Ansible and Ansible Tower

Ansible, an open source community project sponsored by Red Hat, is a **simple automation language** that can perfectly describe an IT application infrastructure in the form of an Ansible Playbook. Ansible is also an **automation engine** that runs Ansible Playbooks.

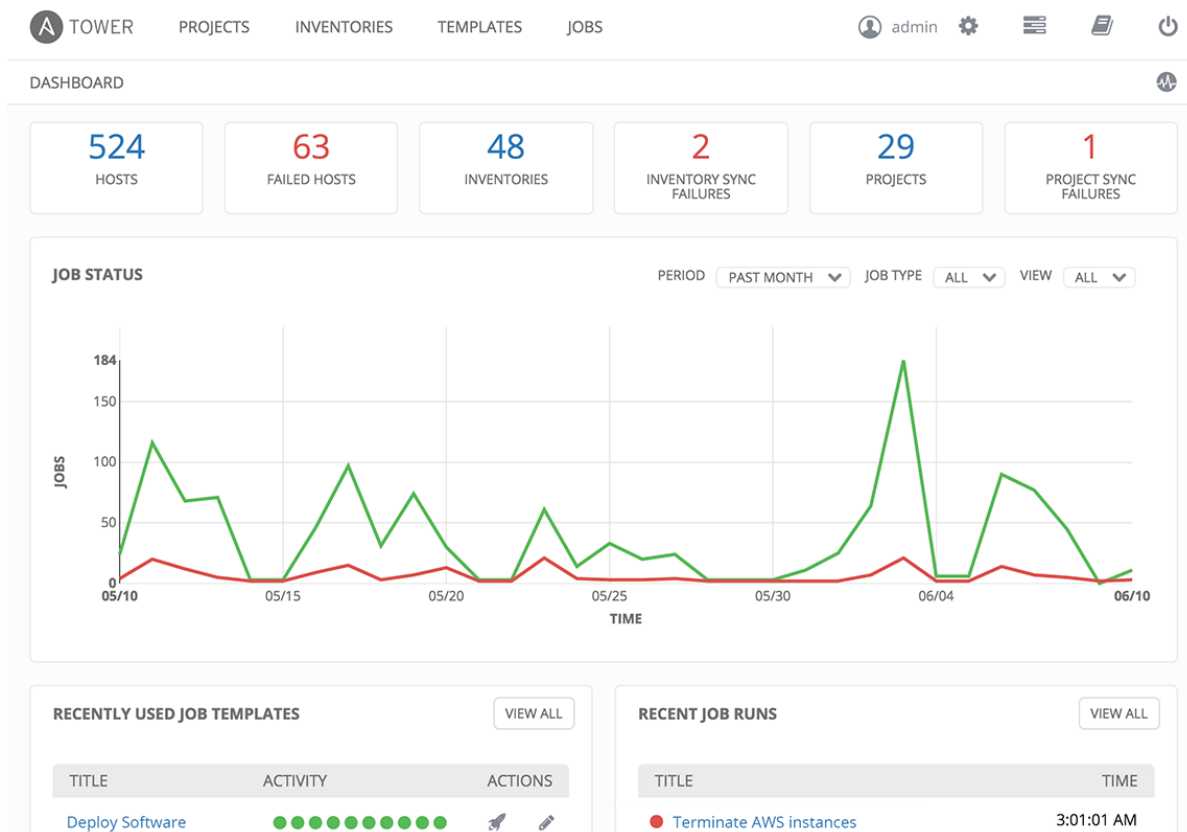
Ansible Tower by **Red Hat** is a commercial offering that helps teams control, secure and manage their Ansible automation. Tower builds on the underlying Ansible automation engine by adding a visual dashboard, role-based access control (RBAC), job scheduling and graphical inventory management.

Using Ansible Tower, all Ansible automations are **centrally logged**, ensuring **complete auditability and compliance**, across the organization.

Ansible Tower Dashboard

The Tower dashboard provides a heads-up NOC-style display for everything going on in your Ansible environment.

As soon as you log in, you'll see your host and inventory status, all the recent job activity and a snapshot of recent job runs. Adjust your job status settings to graph data from specific job and time ranges.



Enable Self-Service IT

Tower lets you launch Playbooks with just a single click. **Role-based access control** (RBAC) keeps environments secure, and teams efficient. Non-privileged users can **safely deploy** entire applications with **push-button deployment** access.

Tower's simplified portal mode and survey features allow IT administrators to delegate automation job runs to users across the organization - synchronized directly from corporate directories such as LDAP, Active Directory or delegated SAML authentication.

With Tower RBAC, developers or QA departments can provision their own development and test environments. Customer service agents can provision a new demo environment. Or junior admins can run simple jobs - like changing passwords - all at the press of a button.

LAUNCH JOB | DEPLOY SOFTWARE

✕

INVENTORY
CREDENTIAL
SURVEY

* ENTER NUMBER OF SERVICE INSTANCES.

2

* PLEASE SELECT THE SERVICE OWNER.

Alice ▼

* ENTER PASSWORD FOR DEPLOYED CERTIFICATE.

SHOW
.....

INVENTORY
 Cloud staging servers

CREDENTIAL
 Staging ssh key

CANCEL
LAUNCH

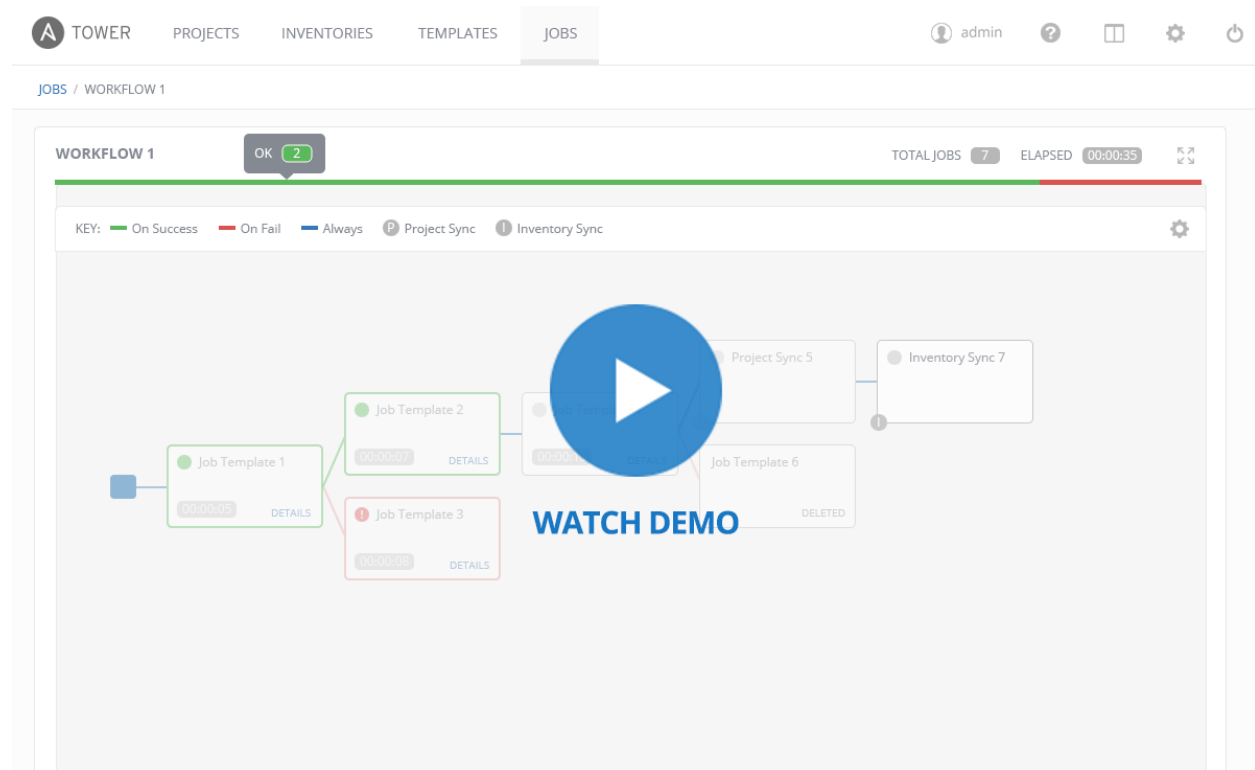
Job Scheduling and Multi-Playbook Workflows

Playbook runs, cloud inventory updates, and source control updates can be scheduled inside Tower - run now, run later, or run forever.

Set up occasional tasks like nightly backups, periodic configuration remediation for compliance, or a full continuous delivery pipeline with just a few clicks.

Tower's multi-Playbook workflows chain any number of Playbooks, regardless of whether they use different inventories, run as different users, run at once or utilize different credentials.

Tower workflows allow for many complex operations. You can build a provisioning workflow that provisions machines, applies a base system configuration, and deploys an application, all with different Playbooks maintained by different teams. You can build a CI/CD testing workflow that builds an application, deploys it to a test environment, runs tests, and automatically promotes the application based on test results. Set up different Playbooks to run in case of success or failure of a prior workflow Playbook. Easily model complex processes with Tower's intuitive workflow editor.



Manage and Track Your Entire Inventory

Tower helps you manage your entire infrastructure. Easily pull your inventory from public cloud providers such as Amazon Web Services, Microsoft Azure, and more. Synchronize from your local OpenStack cloud or VMware environment. Connect your inventory directly to your Red Hat Satellite or Red Hat CloudForms environment. Or connect Tower directly to your custom CMDB.

Tower can keep your cloud inventory in sync, and Tower's powerful provisioning callbacks allow nodes to request configuration on demand, enabling autoscaling.

The screenshot shows the 'Cloud Servers' configuration page in Red Hat Ansible Tower. The page has a top navigation bar with 'TOWER', 'PROJECTS', 'INVENTORIES', 'TEMPLATES', and 'JOBS'. The user is logged in as 'admin'. The breadcrumb trail is 'INVENTORIES / MANAGE CLOUD STAGING SERVERS / EDIT'.

The main form is titled 'CLOUD SERVERS' and has two tabs: 'DETAILS' (selected) and 'NOTIFICATIONS'. The form contains the following fields and options:

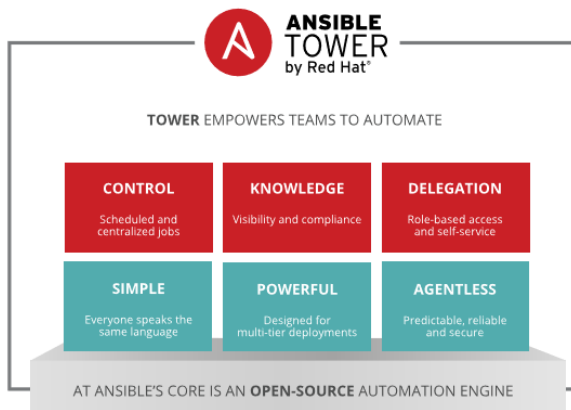
- *NAME:** A text input field containing 'Cloud servers'.
- DESCRIPTION:** An empty text input field.
- SOURCE:** A dropdown menu showing 'Amazon EC2'.
- CLOUD CREDENTIAL:** A search input field containing 'Amazon keys'.
- REGIONS:** A multi-select dropdown showing 'US East (Northern Virginia)'.
- INSTANCE FILTERS:** A text input field containing 'tag:Name=*staging*'.
- ONLY GROUP BY:** An empty text input field.
- UPDATE OPTIONS:** A group of checkboxes:
 - ☒ Overwrite
 - ☒ Overwrite Variables
 - ☐ Update on Launch
- VARIABLES:** Radio buttons for 'YAML' (selected) and 'JSON'.

At the bottom, there is a table with one row and one column, containing the number '1'.

Integrate Tower Using the REST API and CLI Tool

Every feature of Tower is available via Tower's REST API, providing the ideal API for a systems management infrastructure to build against. Call Tower jobs from your existing build tools, show Tower information in your custom dashboards and more. Get API usage information and best practices with built-in documentation.

If it's easier for you to wrap a command line interface than write REST code, Tower's CLI tool is available for launching jobs from CI systems such as Jenkins, or when you need to integrate with other command line tools.



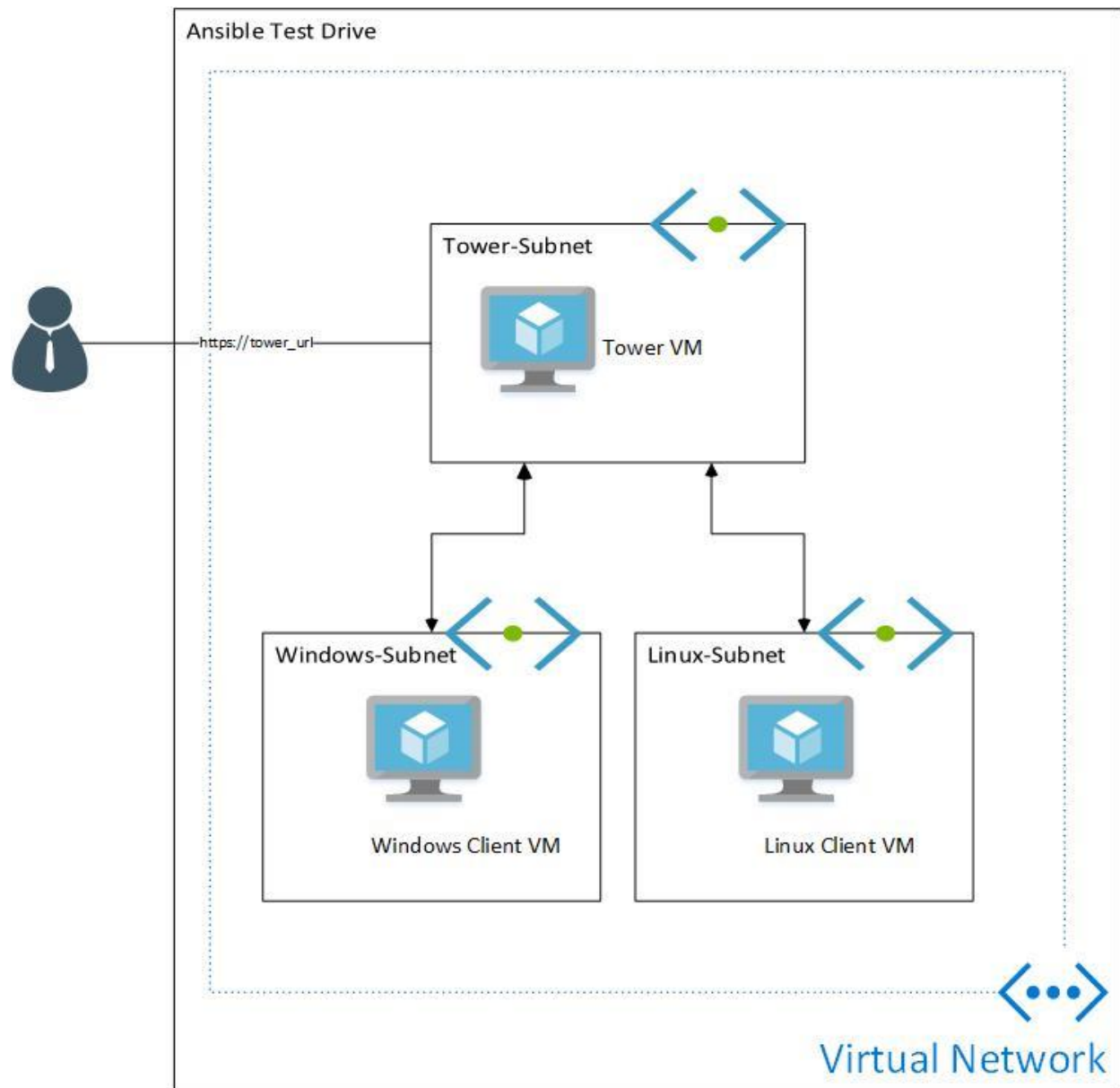
About the Test Drive

This Test Drive will help you experience the capabilities of Ansible Tower. As described in previous sections, Ansible Tower builds upon the Ansible Engine. To get the most out of this Test Drive, the user should be familiar with Ansible and how Ansible Playbooks work. Most users find Ansible incredibly easy to learn and are writing playbooks the same day.

The Ansible website hosts many resources for those wishing to learn more about Ansible (<https://www.ansible.com/get-started>).

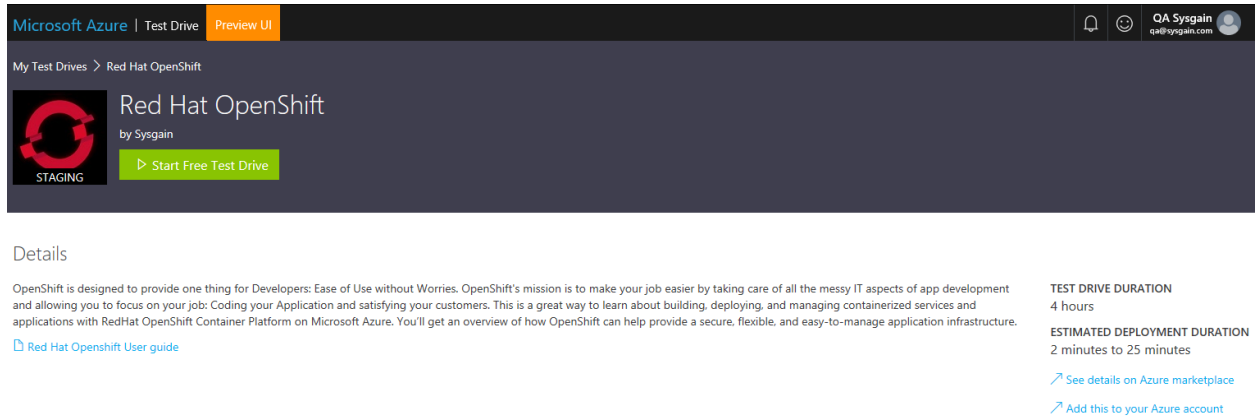
Ansible Tower comes with a feature rich Dashboard to provide users with a nice interface to work with server management. We will be using Ansible Tower to deploy and manage web servers in Windows and Linux environments. Are you ready to take the Driver seat and experience the Red Hat Ansible Tower Test Drive.....?

Deployment Architecture

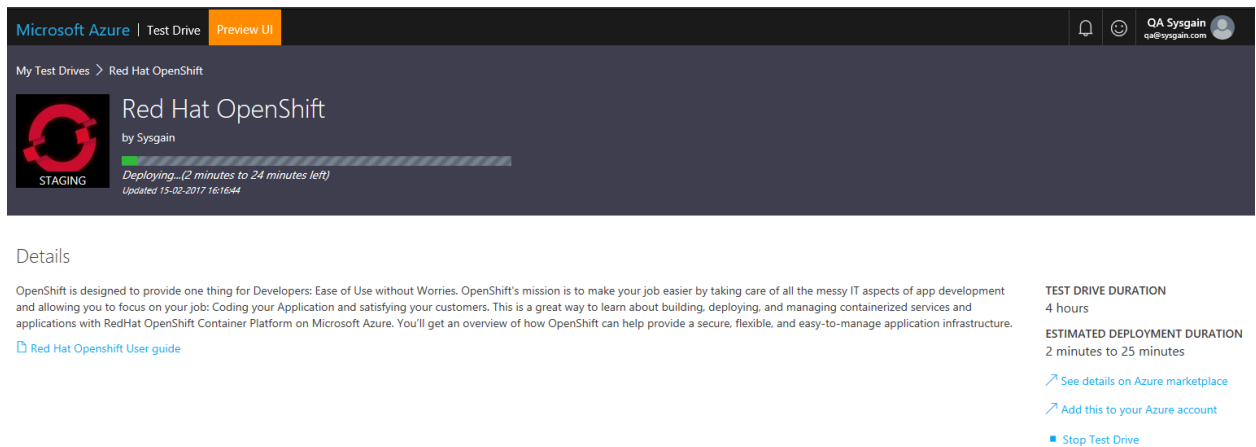


Getting Started

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.

Microsoft Azure | Test Drive Preview UI

My Test Drives > Red Hat OpenShift

Red Hat OpenShift
by Sysgain
Active (4 hours left)

Access information

Step 1: OpenShift Console URL:

- Console URL: <https://opnshmdnsdrmh6uq.southcentralus.cloudapp.azure.com:8443/console>

Step 2: OpenShift Console Credentials:

- Username : testdrive
- Password : password

Step 3: Explore [key use scenarios](#) or refer to the test drive documentation to try more complex scenarios.

Details

OpenShift is designed to provide one thing for Developers: Ease of Use without Worries. OpenShift's mission is to make your job easier by taking care of all the messy IT aspects of app development and allowing you to focus on your job: Coding your Application and satisfying your customers. This is a great way to learn about building, deploying, and managing containerized services and applications with RedHat OpenShift Container Platform on Microsoft Azure. You'll get an overview of how OpenShift can help provide a secure, flexible, and easy-to-manage application infrastructure.

[Red Hat OpenShift User guide](#)

TEST DRIVE DURATION
4 hours

ESTIMATED DEPLOYMENT DURATION
2 minutes to 25 minutes

[See details on Azure marketplace](#)

[Add this to your Azure account](#)

[Stop Test Drive](#)

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

Microsoft Azure | Test Drive Preview UI

My Test Drives > Red Hat OpenShift

Red Hat OpenShift
by Sysgain
Your test drive has ended. ⓘ

[Add this to your Azure account](#) Or [sign up for a free Azure account.](#)

See the Azure marketplace for [more product details](#), including pricing and deployment information.

Details

OpenShift is designed to provide one thing for Developers: Ease of Use without Worries. OpenShift's mission is to make your job easier by taking care of all the messy IT aspects of app development and allowing you to focus on your job: Coding your Application and satisfying your customers. This is a great way to learn about building, deploying, and managing containerized services and applications with RedHat OpenShift Container Platform on Microsoft Azure. You'll get an overview of how OpenShift can help provide a secure, flexible, and easy-to-manage application infrastructure.

[Red Hat OpenShift User guide](#)

TEST DRIVE DURATION
4 hours

ESTIMATED DEPLOYMENT DURATION
2 minutes to 25 minutes

[See details on Azure marketplace](#)

[Add this to your Azure account](#)

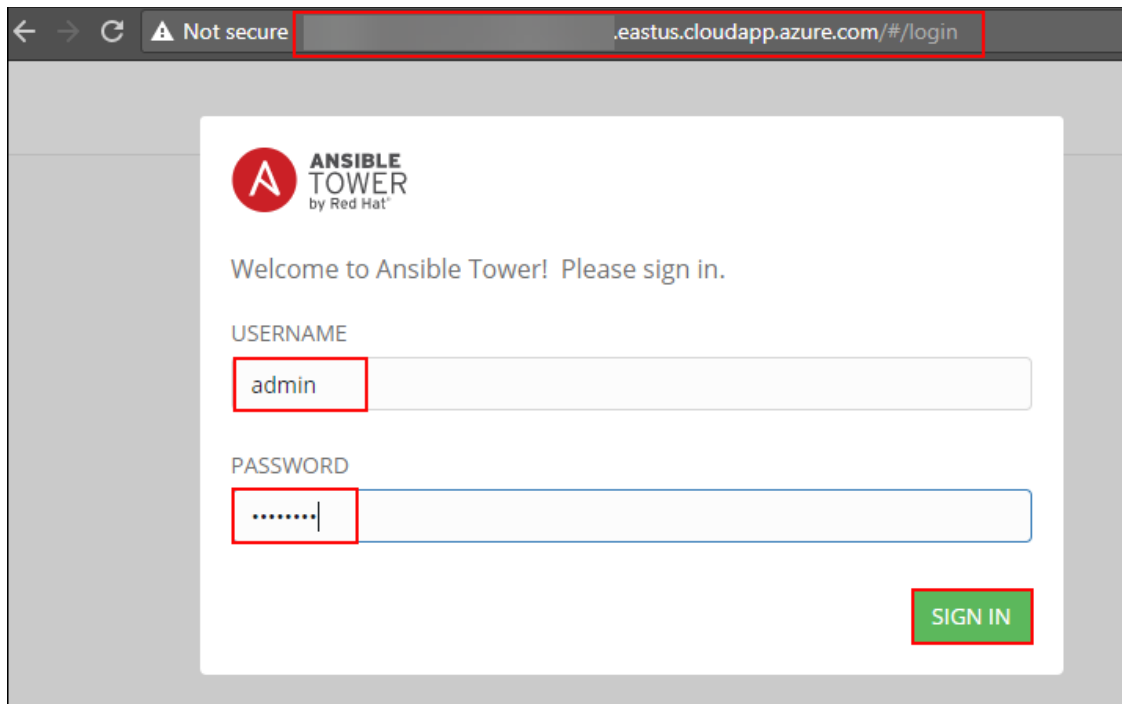
[Start New Free Test Drive](#)

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.



The screenshot shows a web browser window with the address bar displaying `eastus.cloudapp.azure.com/#/login`. The page content features the Ansible Tower logo and the text "Welcome to Ansible Tower! Please sign in." Below this, there are two input fields: "USERNAME" with the value "admin" and "PASSWORD" with masked characters ".....". A green "SIGN IN" button is located at the bottom right of the form.

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>

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-

☐ I agree to the End User License Agreement

SUBMIT

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

Secure | <https://www.ansible.com/license>

ANSIBLE

LET'S GET YOU ACTIVATED!

Great job getting Ansible Tower installed and running.
Now it's time to get a free trial license file.

Select a trial option below based on the number of machines and features you'll need. To compare Tower editions, see www.ansible.com/tower-editions

SELECT A FREE TOWER TRIAL

☐ **FREE TOWER TRIAL - ENTERPRISE FEATURES**
For enterprise IT operations that require more than 10 nodes
Standard and Premium Tower editions include powerful, enterprise features not available in the self-support edition:

- LDAP and Active Directory support
- System tracking
- Audit trails
- Push-button self-service with surveys
- 8x5 or 24x7 support

☒ **FREE TOWER TRIAL - LIMITED FEATURES UP TO 10 NODES**
Self-support trial license that will not expire. Does not include features in Standard and Premium Tower, such as LDAP and Active Directory support, system tracking, audit trails and surveys.

☒ **FREE TOWER TRIAL - LIMITED FEATURES UP TO 10 NODES**

Self-support trial license that will not expire. Does not include features in Standard and Premium Tower, such as LDAP and Active Directory support, system tracking, audit trails and surveys.

First Name*

Last Name*

Company Name*

Role*

▼

Email*

Phone Number*

SUBMIT

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..
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).

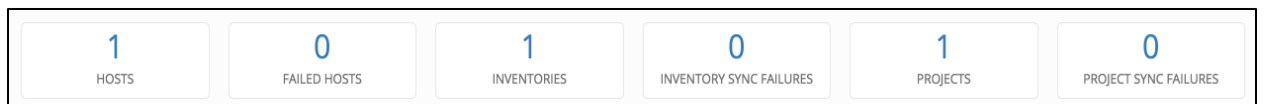
Tower Dashboard

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

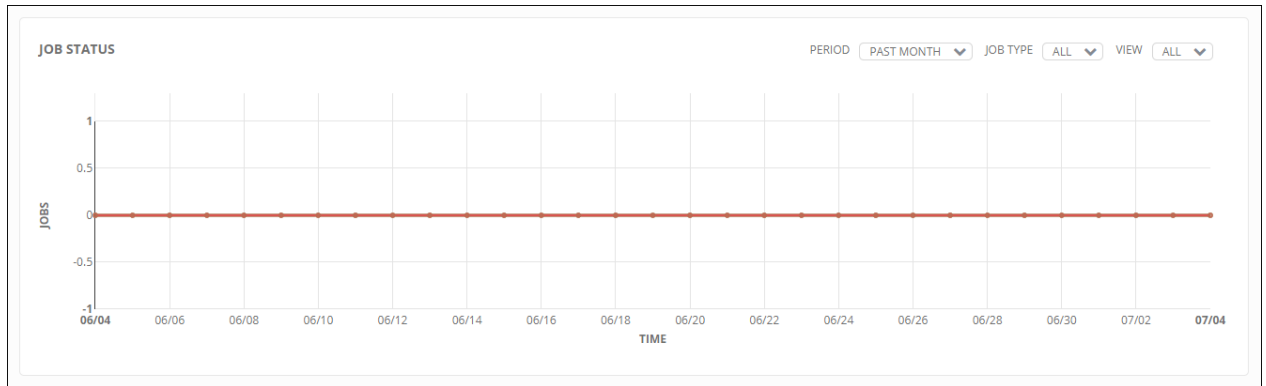
Across the top-right side of this interface, administrators can access the tools they need to configure organizations, users, groups, and permissions as well as view related documentation, access portal mode, and log out.




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.

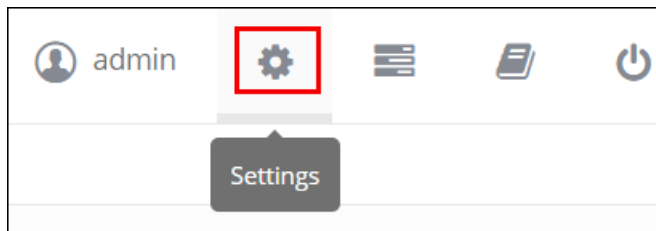


On the main Tower Dashboard screen, a summary appears listing your current **Job Status**. Also, available for review are summaries of **Recently Used Job Templates** and **Recently Run Jobs**.




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  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:**
 - Username:** (Enter the **vmUsername** for **VM** received via **email**)
 - Password:** (Enter the **vmPassword** for **VM** received via **email**)
 - Privilege escalation: SUDO**
 - Privilege escalation Password:** (Provide the **vmPassword** for **VM** you received via **email**)

The screenshot shows the 'CREATE CREDENTIAL' form in Red Hat Ansible Tower. The form is titled 'CREATE CREDENTIAL' and has two tabs: 'DETAILS' and 'PERMISSIONS'. The 'DETAILS' tab is active. The form contains several fields: 'NAME' (LinuxVmCredential), 'DESCRIPTION' (Credentials for Linux VM), 'ORGANIZATION' (default), 'TYPE' (Machine), 'USERNAME' (redacted), 'PASSWORD' (SHOW, redacted), 'PRIVATE KEY PASSPHRASE' (SHOW, redacted), 'PRIVILEGE ESCALATION' (Sudo), 'PRIVILEGE ESCALATION USERNAME' (empty), and 'PRIVILEGE ESCALATION PASSWORD' (SHOW, redacted). Each field is highlighted with a red box. The form also includes checkboxes for 'Ask at runtime?' for the password and private key passphrase fields.

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:**
 - Username:** (Enter the **vmUsername** for **VM** received via **email**)
 - Password:** (Enter the **vmPassword** for **VM** received via **email**)

The screenshot shows the 'WindowsVmCredential' configuration page in Red Hat Ansible Tower. The page has a top navigation bar with 'TOWER', 'PROJECTS', 'INVENTORIES', 'TEMPLATES', and 'JOBS'. The user is logged in as 'admin'. The breadcrumb trail is 'SETTINGS / CREDENTIALS / WindowsVmCredential'. The 'DETAILS' tab is selected. The form contains the following fields:

- NAME:** WindowsVmCredential (highlighted with a red box)
- DESCRIPTION:** Credentials for Windows VM (highlighted with a red box)
- ORGANIZATION:** Searchable field
- TYPE:** Machine (dropdown menu)
- TYPE DETAILS:**
 - USERNAME:** (highlighted with a red box)
 - PASSWORD:** SHOW button, followed by a field containing '*****' (highlighted with a red box). Below it is a checkbox for 'Ask at runtime?'.
 - PRIVATE KEY PASSPHRASE:** SHOW button, followed by a field. Below it is a checkbox for 'Ask at runtime?'.
- PRIVILEGE ESCALATION:** Choose a privilege escalation (dropdown menu)
- VAULT PASSWORD:** SHOW button, followed by a field. Below it is a checkbox for 'Ask at runtime?'.

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>

LightBulb

DETAILS PERMISSIONS NOTIFICATIONS

* NAME: LightBulb DESCRIPTION: Project with Playbook examples * ORGANIZATION: Default

* SCM TYPE: Git

SOURCE DETAILS

* SCM URL: https://github.com/ansible/lightbulb SCM BRANCH: SCM CREDENTIAL:

SCM UPDATE OPTIONS

☐ Clean ☐ Delete on Update ☐ Update on Launch

CANCEL SAVE

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>

The screenshot shows the 'NEW PROJECT' form in Red Hat Ansible Tower. The form is titled 'NEW PROJECT' and has three tabs: 'DETAILS', 'PERMISSIONS', and 'NOTIFICATIONS'. The 'DETAILS' tab is active. The form contains several fields: 'NAME' (WindowsProject), 'DESCRIPTION' (Project with Windows Playbooks), 'ORGANIZATION' (Default), 'SCM TYPE' (Git), 'SCM URL' (https://github.com/SpetraSystems/ansible), 'SCM BRANCH' (empty), 'SCM CREDENTIAL' (empty), and 'SCM UPDATE OPTIONS' (Clean, Delete on Update, Update on Launch). The 'SAVE' button is highlighted in green.

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**

INVENTORIES / CREATE INVENTORY

*NAME: Agent VM Inventory

DESCRIPTION: Inventory with Agent hosts

*ORGANIZATION: Default

VARIABLES: ☒ YAML ☐ JSON

1 ---

CANCEL SAVE

- Click on **Save** when done.
- Now, click the **+ ADD** button in GROUPS, which launches the **Create Group** dialog.
- Enter the given details into the following fields:

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

INVENTORIES / Agent VM Inventory / web

web

DETAILS NOTIFICATIONS

*NAME: web

DESCRIPTION: Group for Linux Hosts

SOURCE: Manual

VARIABLES: ☒ YAML ☐ JSON

1 ---

CANCEL SAVE

- Click on **Save** when done.
- Now, click on the created group **web**
- Then, click the **+ ADD** button in HOSTS, which launches the **Create Host** dialog.
- 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

[INVENTORIES](#) / [Agent VM Inventory](#) / web

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

INVENTORIES / Agent VM Inventory / CREATE GROUP

CREATE GROUP

DETAILS

NOTIFICATIONS

* NAME

windowsVM

DESCRIPTION

Group for Windows Hosts

SOURCE

Manual

VARIABLES

YAML

JSON

1


2ansible_connection: winrm

3ansible_winrm_server_cert_validation: ignore

4


CANCEL


SAVE

14. Click on **Save** when done.
15. Now, click on the created group **windowsVM**
16. Then, click the  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**

INVENTORIES / Agent VM Inventory / windowsVM / CREATE HOST

CREATE HOST ☒

*HOST NAME  DESCRIPTION

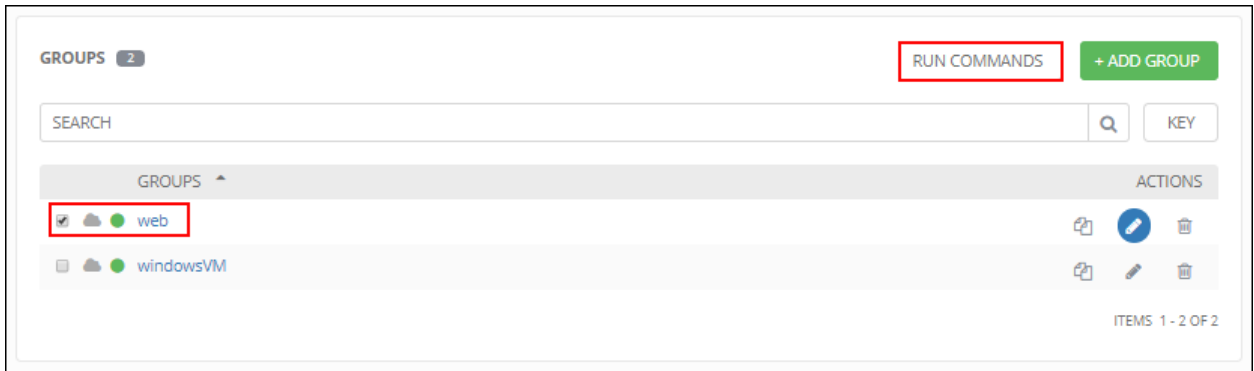
VARIABLES  ☒ YAML ☐ JSON

1

CANCEL

SAVE

- Click on **Save** when done. Now we will test the connectivity to the added hosts.
- Now go back to Agent VM Inventory page by clicking on Agent VM Inventory
- 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.

JOBS / ping

RESULTS

NAME	ping
STATUS	● Successful
STARTED	5/7/2017 20:25:18
FINISHED	5/7/2017 20:25:22
ELAPSED	4.407 seconds
INVENTORY	Agent VM Inventory
CREDENTIAL	LinuxVmCredential
LAUNCHED BY	admin
FORKS	0
LIMIT	web
VERBOSITY	0
EXTRA VARIABLES	1 ---

STANDARD OUT

```
SSH password:
SUDO password[defaults to SSH password]:
.eastus.cloudapp.azure.com | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
```

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

GROUPS 2

RUN COMMANDS + ADD GROUP

SEARCH Q KEY

GROUPS	ACTIONS
<input type="checkbox"/> ● web	Copy Edit Delete
<input checked="" type="checkbox"/> ● windowsVM	Copy Edit Delete

ITEMS 1 - 2 OF 2

25. Enter the given details into the following fields:
 - Module: Select **win_ping**
 - Machine Credential: **WindowsVmCredential**

INVENTORIES / Agent VM Inventory / RUN COMMAND

EXECUTE COMMAND

*MODULE

win_ping

ARGUMENTS

LIMIT

windowsVM

*MACHINE CREDENTIAL

WindowsVmCredential

ENABLE PRIVILEGE ESCALATION

*VERBOSITY

0 (Normal)

*FORKS

0

EXTRA VARIABLES

```
1 ---
```

RESET

LAUNCH

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

JOBS / win_ping

RESULTS

NAME	win_ping
STATUS	Successful
STARTED	5/7/2017 16:57:07
FINISHED	5/7/2017 16:57:21
ELAPSED	14.226 seconds
INVENTORY	Agent VM Inventory
CREDENTIAL	WindowsVmCredential
LAUNCHED BY	admin
FORKS	0
LIMIT	windowsVM
VERBOSITY	0
EXTRA VARIABLES	<pre>1 ---</pre>

STANDARD OUT

```
SSH password: .eastus.cloudapp.azure.com | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
```

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.

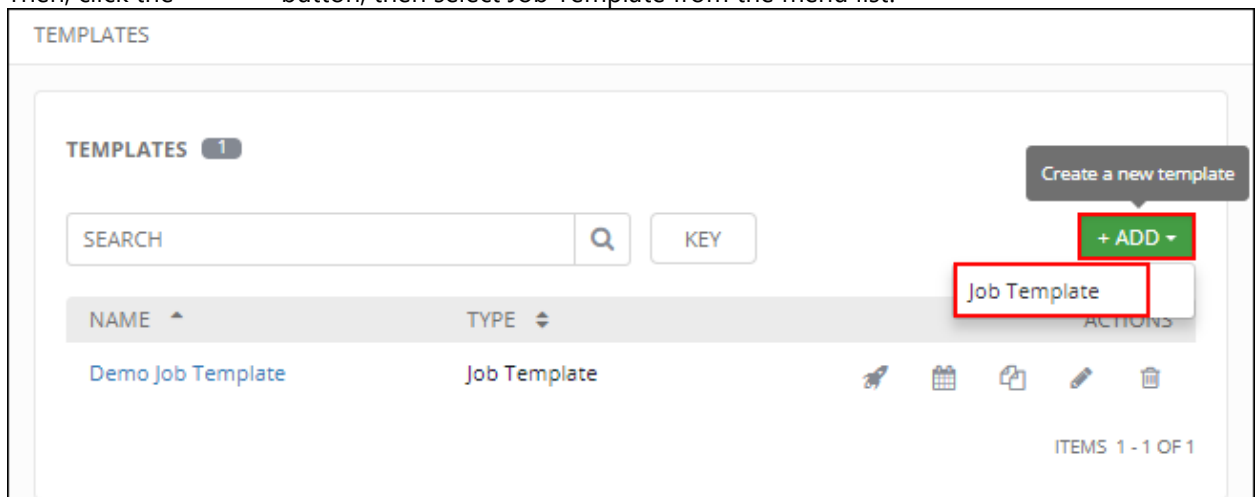
We can access Job Template page by clicking on TEMPLATES on the menu on top of the dashboard.



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  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*

The screenshot shows the 'Install Nginx' job template configuration page. The fields are organized into sections: NAME, DESCRIPTION, JOB TYPE, INVENTORY, PROJECT, PLAYBOOK, MACHINE CREDENTIAL, CLOUD CREDENTIAL, NETWORK CREDENTIAL, FORKS, LIMIT, VERBOSITY, JOB TAGS, SKIP TAGS, and OPTIONS. Red boxes highlight the following values: 'Install Nginx' in the NAME field, 'Run' in the JOB TYPE dropdown, 'Agent VM Inventory' in the INVENTORY dropdown, 'LightBulb' in the PROJECT dropdown, 'examples/nginx-basic-playbook/site.yml' in the PLAYBOOK dropdown, 'LinuxVmCredential' in the MACHINE CREDENTIAL dropdown, 'web' in the LIMIT dropdown, and the 'Enable Privilege Escalation' checkbox in the OPTIONS section.

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.

NAME	TYPE	ACTIVITY	LABELS	ACTIONS
Demo Job Template	Job Template			
Install Nginx	Job Template			

ITEMS 1 - 2 OF 2

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.

The screenshot displays the Red Hat Ansible Tower interface. On the left, the 'DETAILS' panel shows the job status as 'Successful' (highlighted with a red box). The job was started on 5/7/2017 at 22:34:13 and finished at 22:34:38. The template used is 'Install Nginx', and the job type is 'Run'. The job was launched by 'admin' and is associated with the 'Agent VM Inventory'. The project is 'LightBulb', and the revision is 'd319ec4103c2816309879fd0a9f3e2080a555f4'. The playbook is 'examples/nginx-basic-playbook/site.yml', and the machine credential is 'LinuxVmCredential'. The job has 0 forks, a limit of 'web', and a verbosity of 0 (Normal). The 'EXTRA VARIABLES' section is empty.

On the right, the 'Install Nginx' job execution details are shown. The job is categorized under 'PLAYS' (1) and 'TASKS' (8). The execution log shows the following steps:

- 1 SSH password:
- 2 SUDO password[defaults to SSH password]:
- 3
- 4 PLAY [install and start nginx with wsgi] *****
- 5
- 6 TASK [Gathering Facts] *****
- 7 Ok: [.eastus.cloudapp.azure.com]
- 8
- 9 TASK [nginx packages are present] *****
- 10 Ok: [.eastus.cloudapp.azure.com] => (11 u'python-devel', u'gcc')]
- 11
- 12 TASK [uwsgi package is present] *****
- 13 Ok: [.eastus.cloudapp.azure.com]
- 14
- 15 TASK [latest default.conf is present] *****
- 16 Ok: [.eastus.cloudapp.azure.com]
- 17
- 18 TASK [latest index.html is present] *****
- 19 Ok: [.eastus.cloudapp.azure.com]
- 20
- 21 TASK [nginx service is started and enabled] *****
- 22 Ok: [linagenteyomfjqagpwm1.eastus.cloudapp.azure.com]
- 23


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

Now to verify that Nginx 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  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

NEW JOB TEMPLATE

DETAILS | COMPLETED JOBS | PERMISSIONS | NOTIFICATIONS

* NAME: Remove Nginx

DESCRIPTION:

* JOB TYPE: Run

Prompt on launch

* INVENTORY: Agent VM Inventory

* PROJECT: LightBulb

* PLAYBOOK: examples/nginx-remove-playbook/site.yml

Prompt on launch

* MACHINE CREDENTIAL: LinuxVmCredential

CLOUD CREDENTIAL:

NETWORK CREDENTIAL:

Prompt on launch

FORKS: 0

LIMIT: web

* VERBOSITY: 0 (Normal)

Prompt on launch

JOB TAGS:

SKIP TAGS:

Prompt on launch

OPTIONS:

- ☒ Enable Privilege Escalation
- ☐ Allow Provisioning Callbacks
- ☒ Enable Concurrent Jobs

- 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.

TEMPLATES 3

SEARCH

NAME	TYPE
Demo Job Template	Job Template
Install Nginx	Job Template
Remove Nginx	Job Template

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.

The screenshot displays the Red Hat Ansible Tower interface. On the left, the 'DETAILS' panel shows the job status as 'Successful' (highlighted with a red box). The job is titled 'Remove Nginx' and was launched by 'admin' on '5/7/2017 22:34:43'. The inventory used is 'Agent VM Inventory' and the project is 'LightBulb'. The playbook is 'examples/nginx-remove-playbook/site.yml' and the machine credential is 'LinuxVmCredential'. The job type is 'Run' and the limit is 'web'. The verbosity is '0 (Normal)'. The 'EXTRA VARIABLES' section is empty.

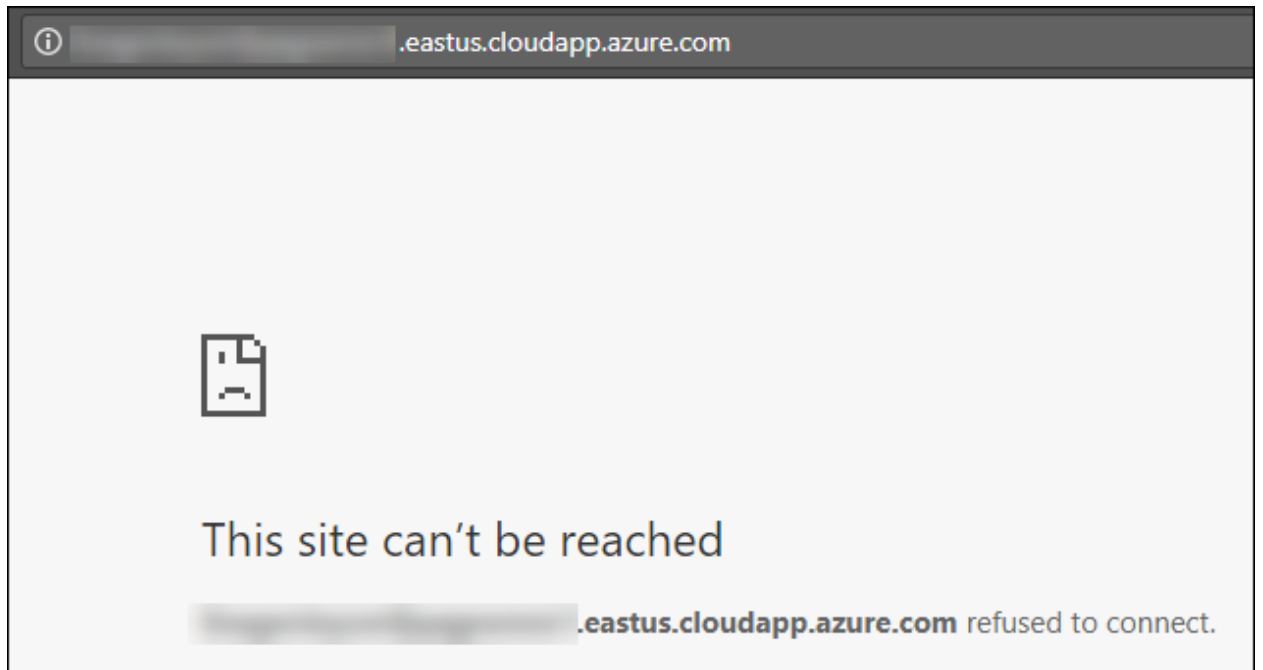
On the right, the 'Remove Nginx' job execution details are shown. The job is completed with 1 play, 5 tasks, and 1 host. The tasks executed are:

- PLAY [removes nginx with wsgi]
- TASK [Gathering Facts]
- TASK [nginx service is stopped]
- TASK [nginx package is absent]
- TASK [uwsgi package is absent]
- TASK [files created by nginx-simple are absent]

The job recap shows 5 OK, 3 changed, and 0 failed.

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

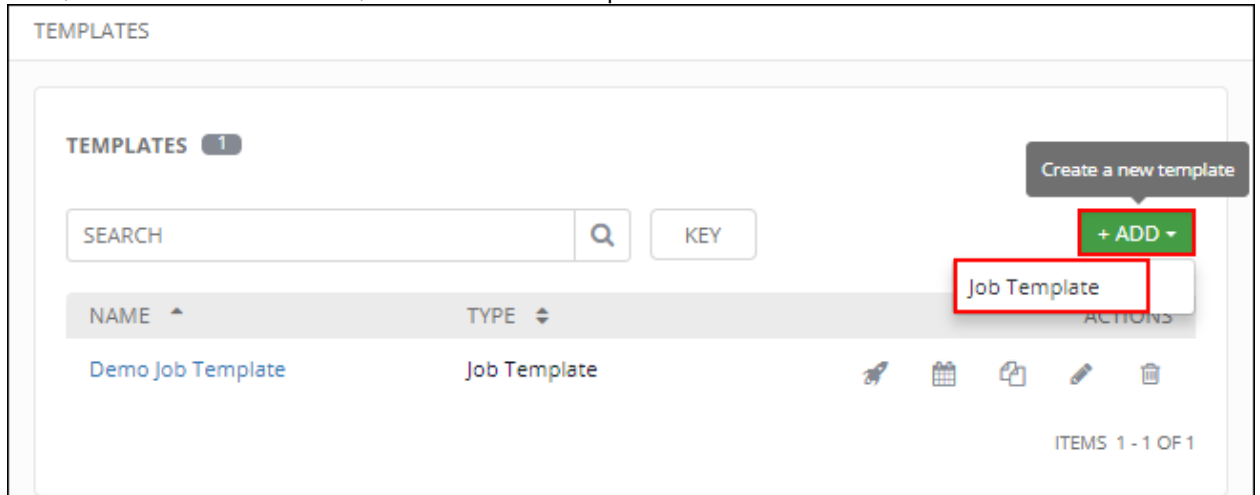
Now to verify that Nginx Webserver is removed from the Linux Virtual Machine and is not accessible through Public DNS Name, we will open a browser and navigate to the Linux VM DNS name and check if the page is unreachable



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

Install Apache

DETAILS | COMPLETED JOBS | PERMISSIONS | NOTIFICATIONS

* NAME: DESCRIPTION:

* JOB TYPE:
 ☐ Prompt on launch

* INVENTORY:
 ☐ Prompt on launch

* PROJECT:
 ☐ Prompt on launch

* PLAYBOOK:

* MACHINE CREDENTIAL:
 ☐ Prompt on launch

CLOUD CREDENTIAL:

NETWORK CREDENTIAL:

FORKS:
 ☐ Prompt on launch

LIMIT:
 ☐ Prompt on launch

* VERBOSITY:

JOB TAGS:

SKIP TAGS:

OPTIONS:
 ☒ Enable Privilege Escalation
 ☐ Allow Provisioning Callbacks
 ☐ Enable Concurrent Jobs

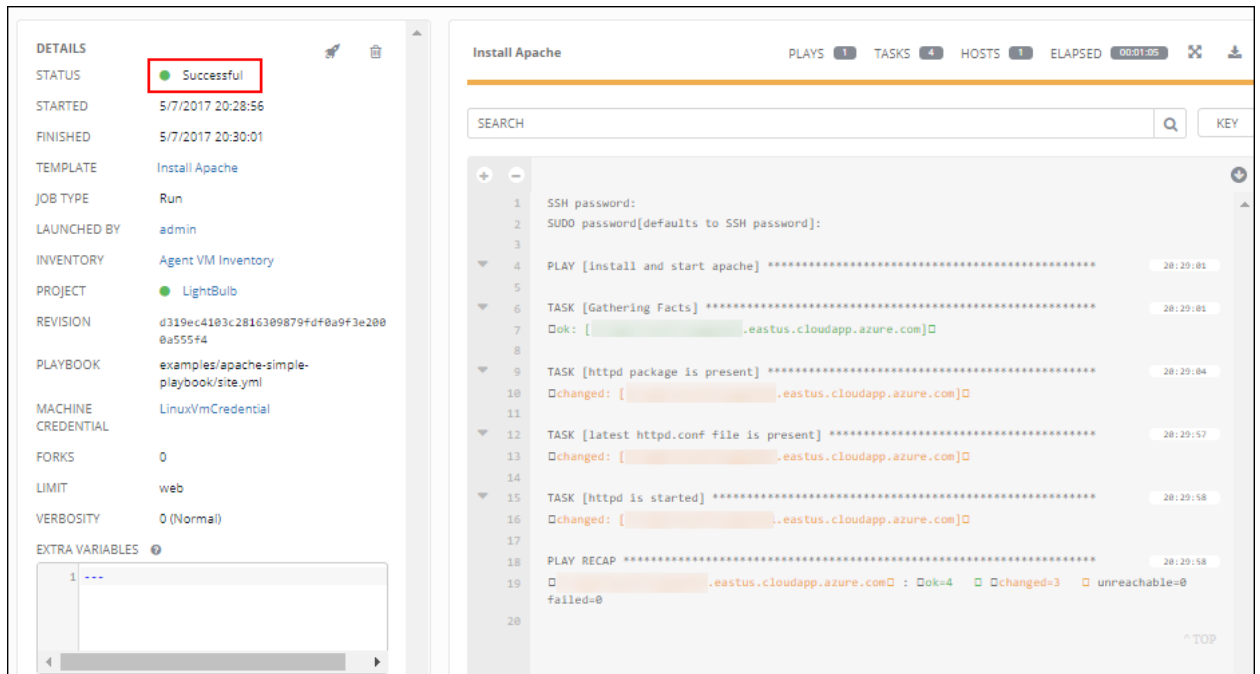
3. Click on **Save** when done.

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.

TEMPLATES 4	
SEARCH	
NAME	TYPE
Demo Job Template	Job Template
Install Apache	Job Template
Install Nginx	Job Template
Remove Nginx	Job Template

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 apache 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

NEW JOB TEMPLATE

DETAILS | COMPLETED JOBS | PERMISSIONS | NOTIFICATIONS

* NAME: **Create User**

DESCRIPTION:

* JOB TYPE: Run

☐ Prompt on launch

* INVENTORY: **Agent VM Inventory**

* PROJECT: **WindowsProject**

* PLAYBOOK: **windows/create-user.yml**

☐ Prompt on launch

* MACHINE CREDENTIAL: **WindowsVmCredential**

CLOUD CREDENTIAL:

NETWORK CREDENTIAL:

* FORKS: 0

* LIMIT: **windowsVM**

☐ Prompt on launch

* VERBOSITY: 0 (Normal)

JOB TAGS:

SKIP TAGS:

☐ Prompt on launch

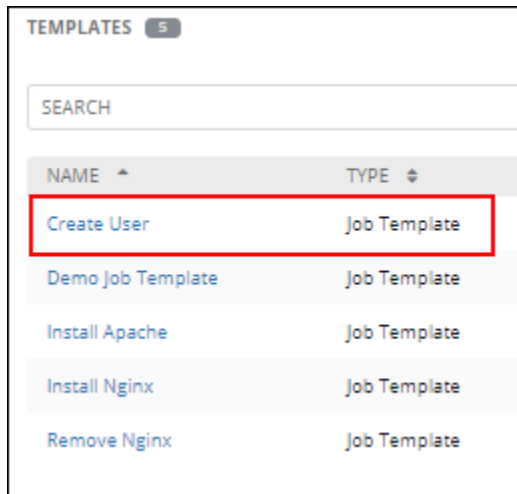
OPTIONS

- ☐ Enable Privilege Escalation
- ☐ Allow Provisioning Callbacks
- ☐ Enable Concurrent Jobs

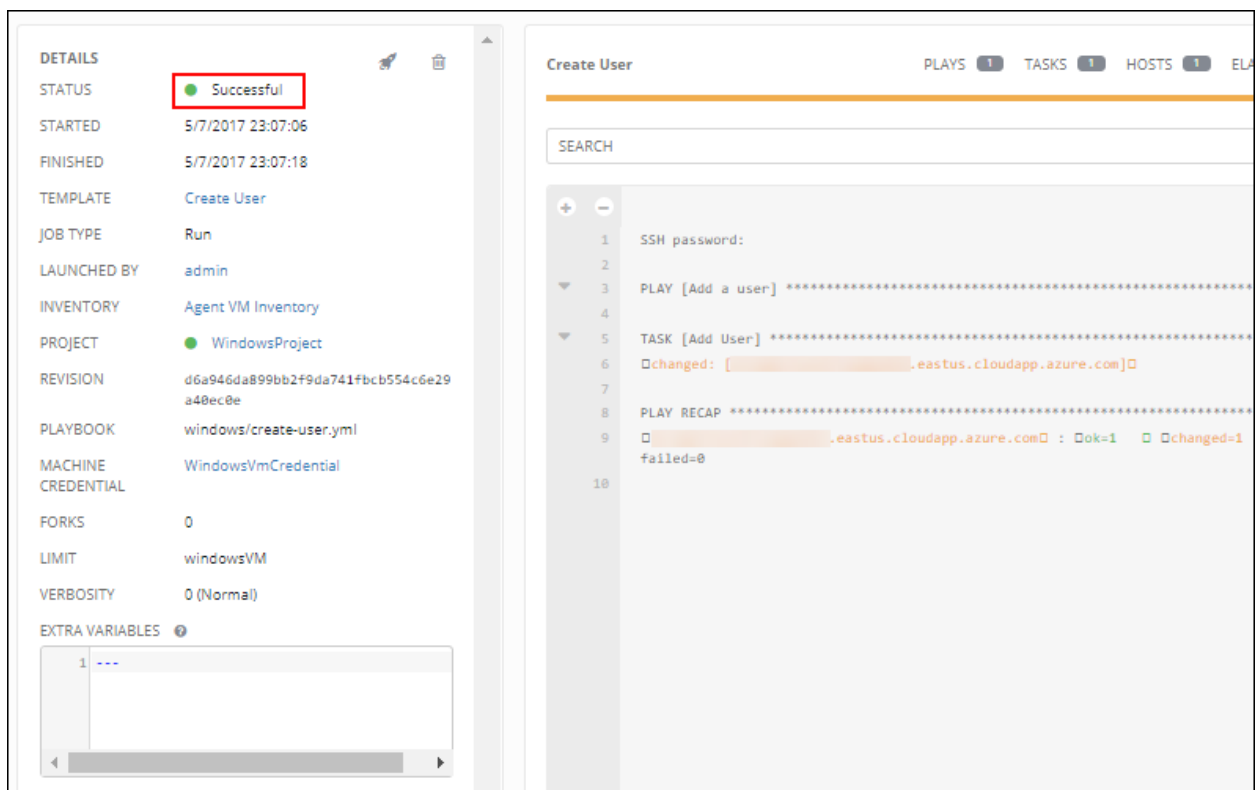
3. Click on **Save** when done.

The play create-user.yml has commands for creating a local user named 'ansible' with password 'ans1bl3@P4ssw0rd'

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.

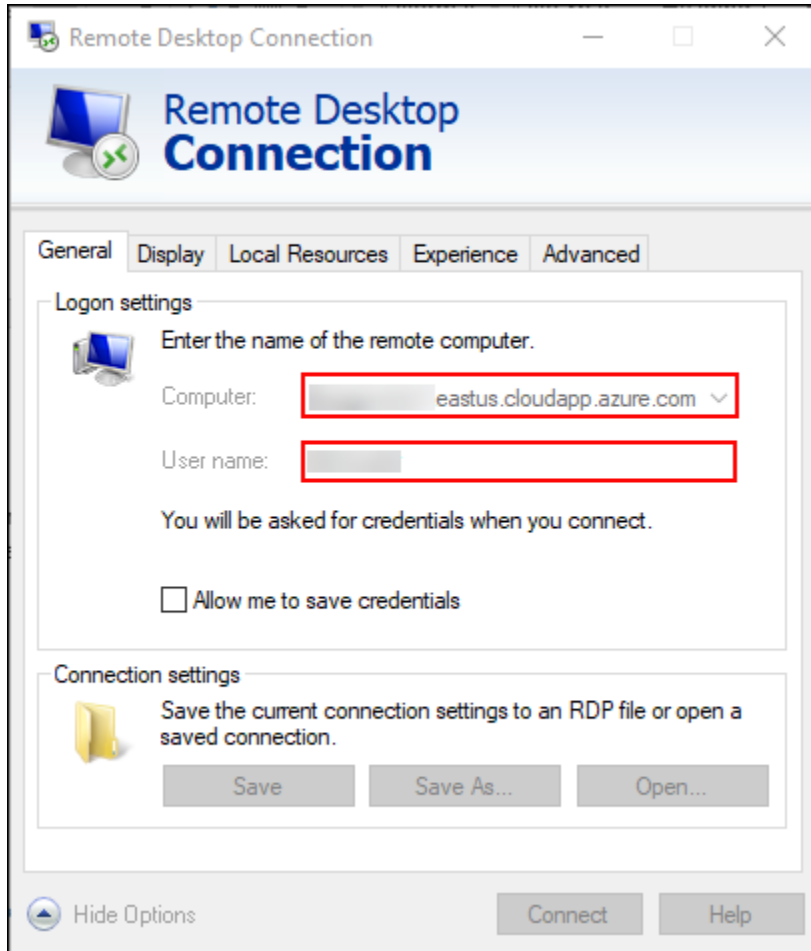


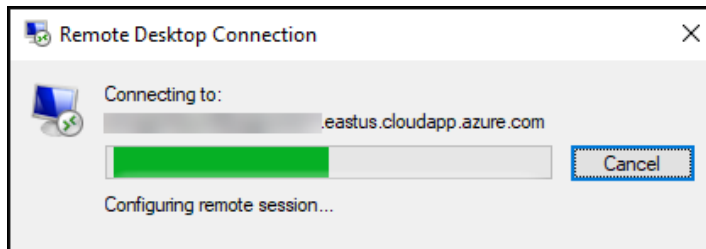
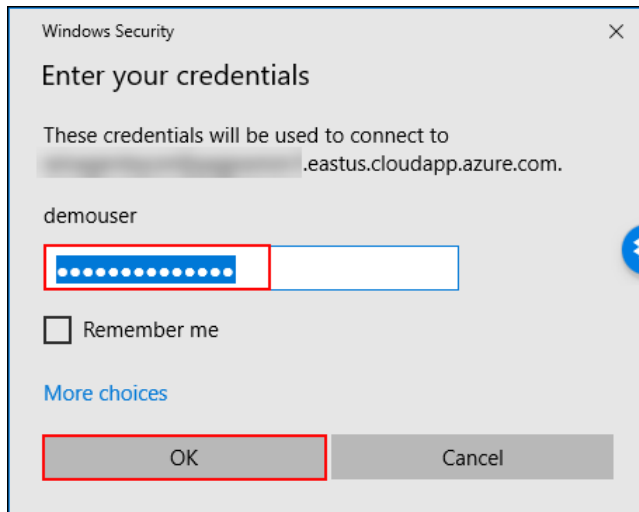
We can see the status as successful for Job to create a local user account in Windows VM.

Now to verify that a user account with username 'ansible' is created on Windows Virtual Machine, we will access the Virtual Machine using a RDP client and check.

Using RDP Client, DNS Name of Windows VM, username and password of VM we received via email, log in to the Windows VM.

From Windows, using the built-in RDP Client, we can connect like given below:



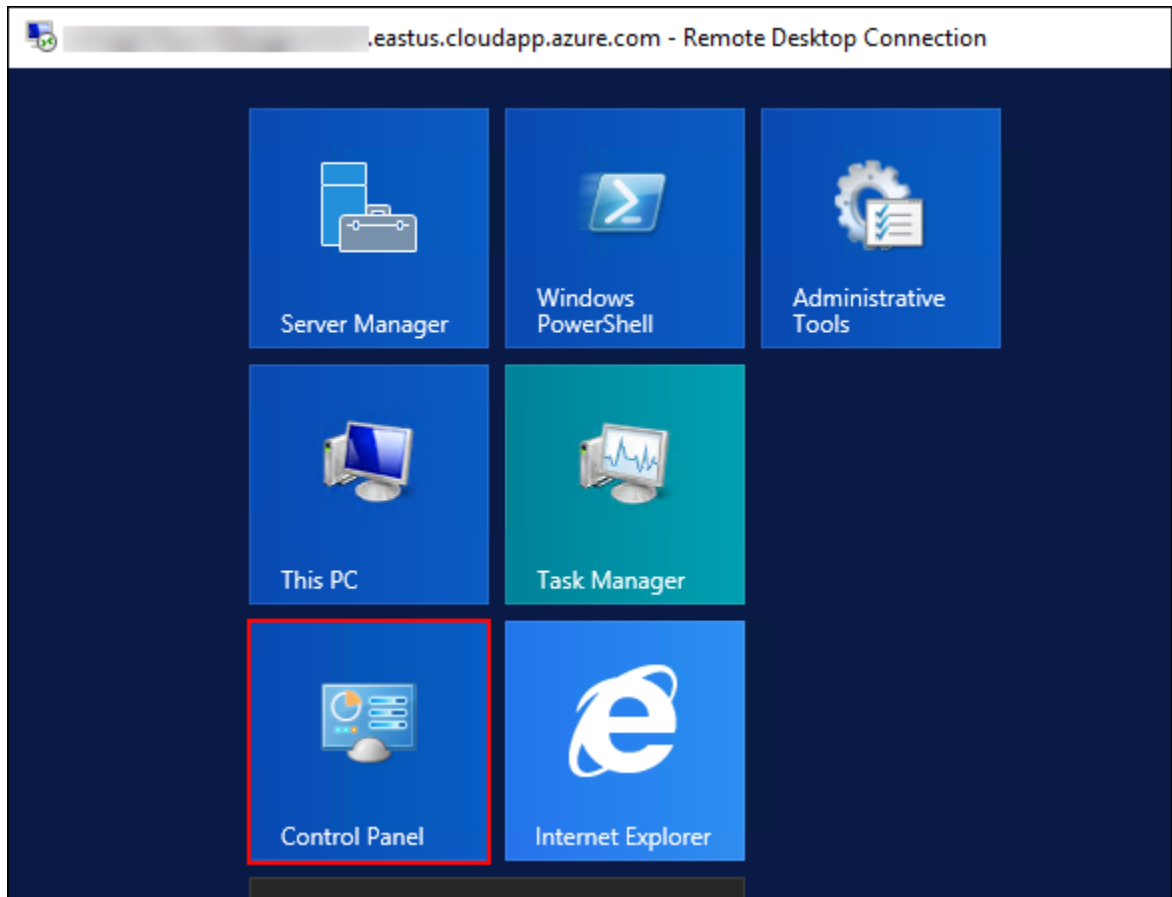


Once we are logged in to the virtual machine, we should go to the User accounts inside Control Panel.

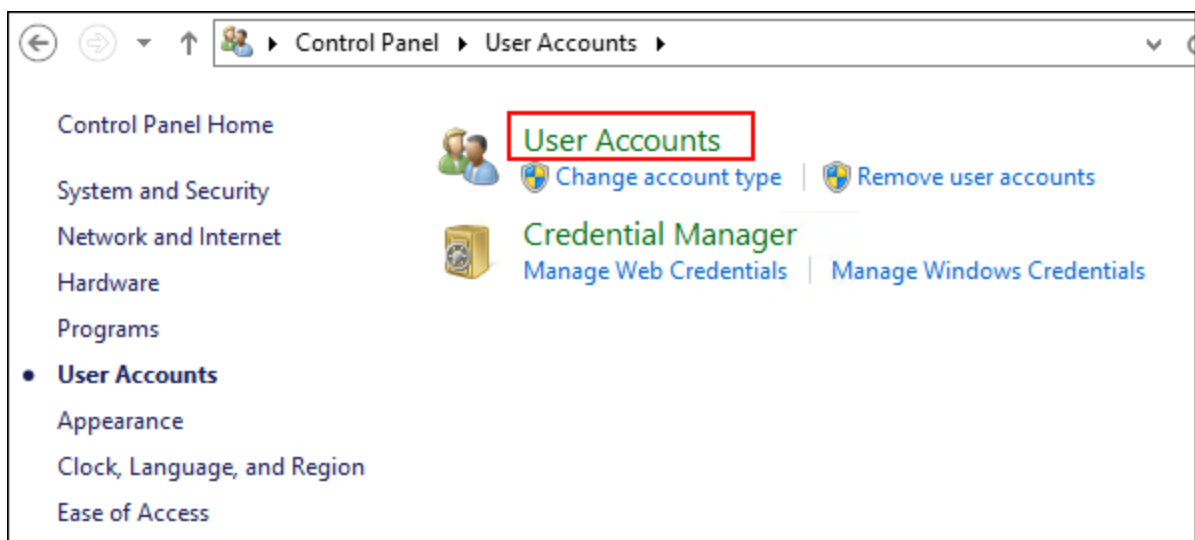
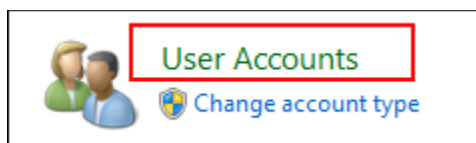
For that, we will click on Windows button the bottom left of the windows vm.

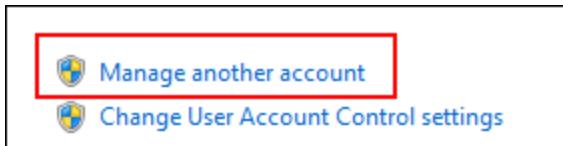


Then Select Control Panel from the tiles listed.

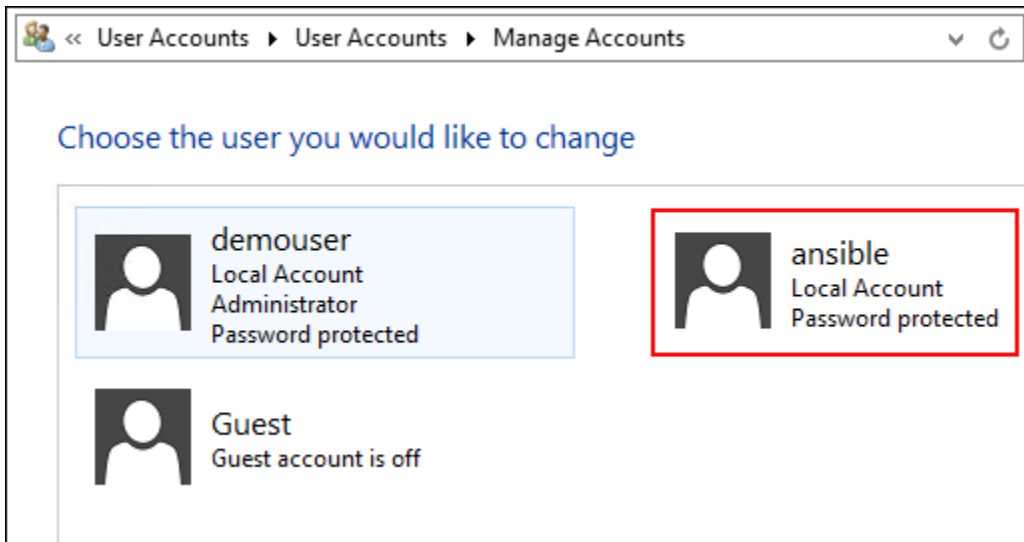


From the options listed, we should go to User accounts and again select User accounts and then select Manage another account to view the other User accounts present in the Virtual Machine.






In the Manage Accounts page, we can see that the user ansible has been created.



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  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

NEW JOB TEMPLATE

DETAILS COMPLETED JOBS PERMISSIONS NOTIFICATIONS

* NAME DESCRIPTION

* JOB TYPE
 ☐ Prompt on launch

* INVENTORY * PROJECT * PLAYBOOK
 ☐ Prompt on launch

* MACHINE CREDENTIAL CLOUD CREDENTIAL NETWORK CREDENTIAL
 ☐ Prompt on launch

FORKS LIMIT * VERBOSITY
 ☐ Prompt on launch

JOB TAGS SKIP TAGS
 ☐ Prompt on launch

OPTIONS
 ☐ Enable Privilege Escalation
 ☐ Allow Provisioning Callbacks
 ☐ Enable Concurrent Jobs

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.

TEMPLATES 6

SEARCH

NAME	TYPE
Create User	Job Template
Demo Job Template	Job Template
Install Apache	Job Template
Install Firefox browser	Job Template
Install Nginx	Job Template
Remove Nginx	Job Template

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.

The screenshot displays the Red Hat Ansible Tower interface. On the left, the 'DETAILS' panel shows the job status as 'Successful' (highlighted with a red box). The job was started on 5/7/2017 at 23:21:24 and finished at 23:22:43. The template used is 'Install Firefox browser', and the job type is 'Run'. The job was launched by 'admin' and is associated with the 'Agent VM Inventory'. The project is 'WindowsProject', and the revision is '854a6165c4cb13a6f894f79d2247d2aba6903678'. The playbook is 'windows/install-firefox.yml', and the machine credential is 'WindowsVmCredential'. There are 0 forks, a limit of 'windowsVM', and a verbosity of 0 (Normal). The 'EXTRA VARIABLES' section is empty.

The main panel shows the execution details for the 'Install Firefox browser' job. It includes a search bar and a list of tasks. The tasks are:

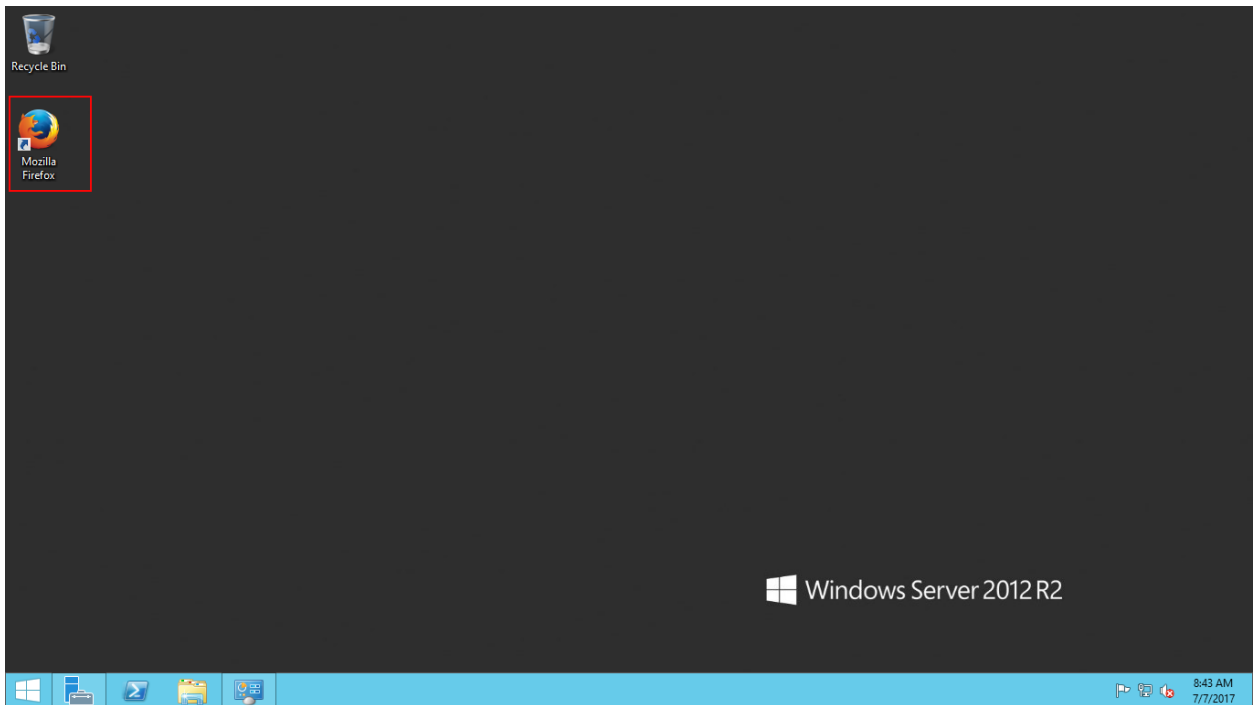
- 1. SSH password:
- 2.
- 3. PLAY [test chocolatey with ansible] *****
- 4.
- 5. TASK [Gathering Facts] *****
- 6. ok: [eastus.cloudapp.azure.com]
- 7.
- 8. TASK [Install Firefox] *****
- 9. changed: [eastus.cloudapp.azure.com]
- 10.
- 11. PLAY RECAP *****
- 12. eastus.cloudapp.azure.com : ok=2 changed=1 failed=0
- 13.

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 access the Virtual Machine using a RDP client and check.

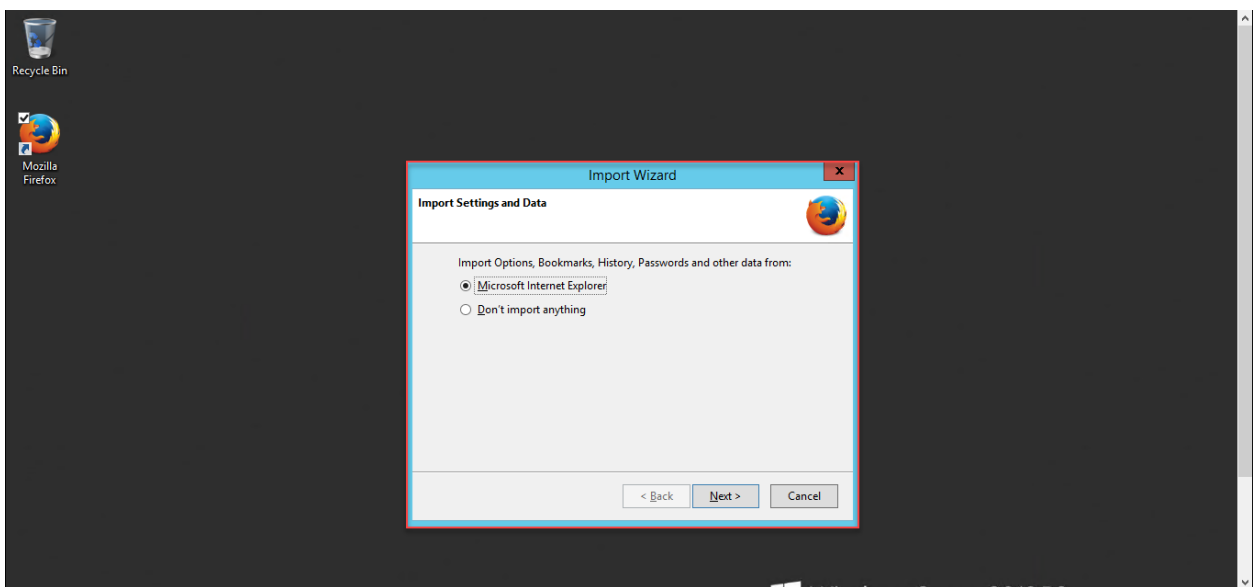
Using RDP Client, DNS Name of Windows VM, username and password of VM we received via email, we will log in to the Windows VM.

Once we are logged in to the virtual machine, we can see the Mozilla Firefox browser icon on the Desktop Home Screen.

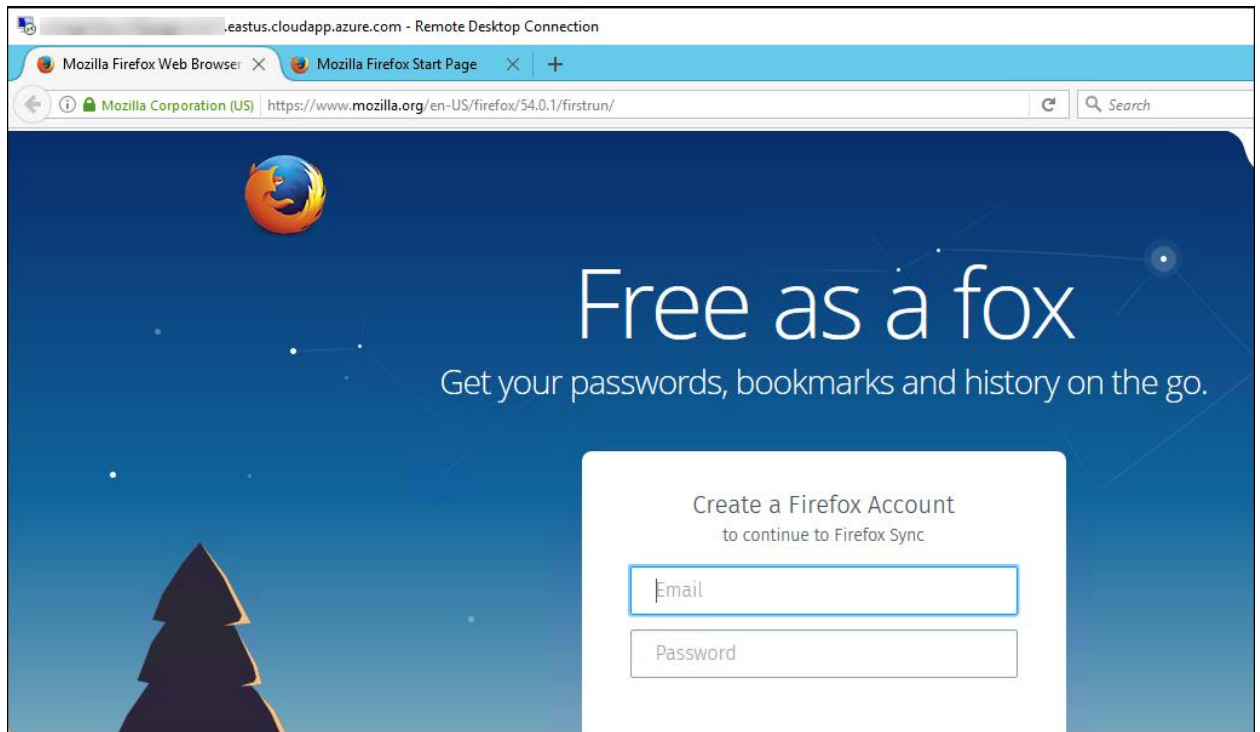


To verify that browser was properly installed, we will open the browser and check.

An Import Wizard will come up to import any existing bookmarks.




Once the initial configuration is done, the browser will be launched.



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  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

NEW JOB TEMPLATE

DETAILS COMPLETED JOBS PERMISSIONS NOTIFICATIONS

* NAME DESCRIPTION

* JOB TYPE
 ☐ Prompt on launch

* INVENTORY * PROJECT * PLAYBOOK
 ☐ Prompt on launch

* MACHINE CREDENTIAL CLOUD CREDENTIAL NETWORK CREDENTIAL
 ☐ Prompt on launch

FORKS LIMIT * VERBOSITY
 ☐ Prompt on launch

JOB TAGS SKIP TAGS
 ☐ Prompt on launch

OPTIONS
 ☐ Enable Privilege Escalation
 ☐ Allow Provisioning Callbacks
 ☐ Enable Concurrent Jobs

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.

TEMPLATES 7

SEARCH

NAME	TYPE
Create User	Job Template
Demo Job Template	Job Template
Install Apache	Job Template
Install Firefox browser	Job Template
Install IIS	Job Template
Install Nginx	Job Template
Remove Nginx	Job Template

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.

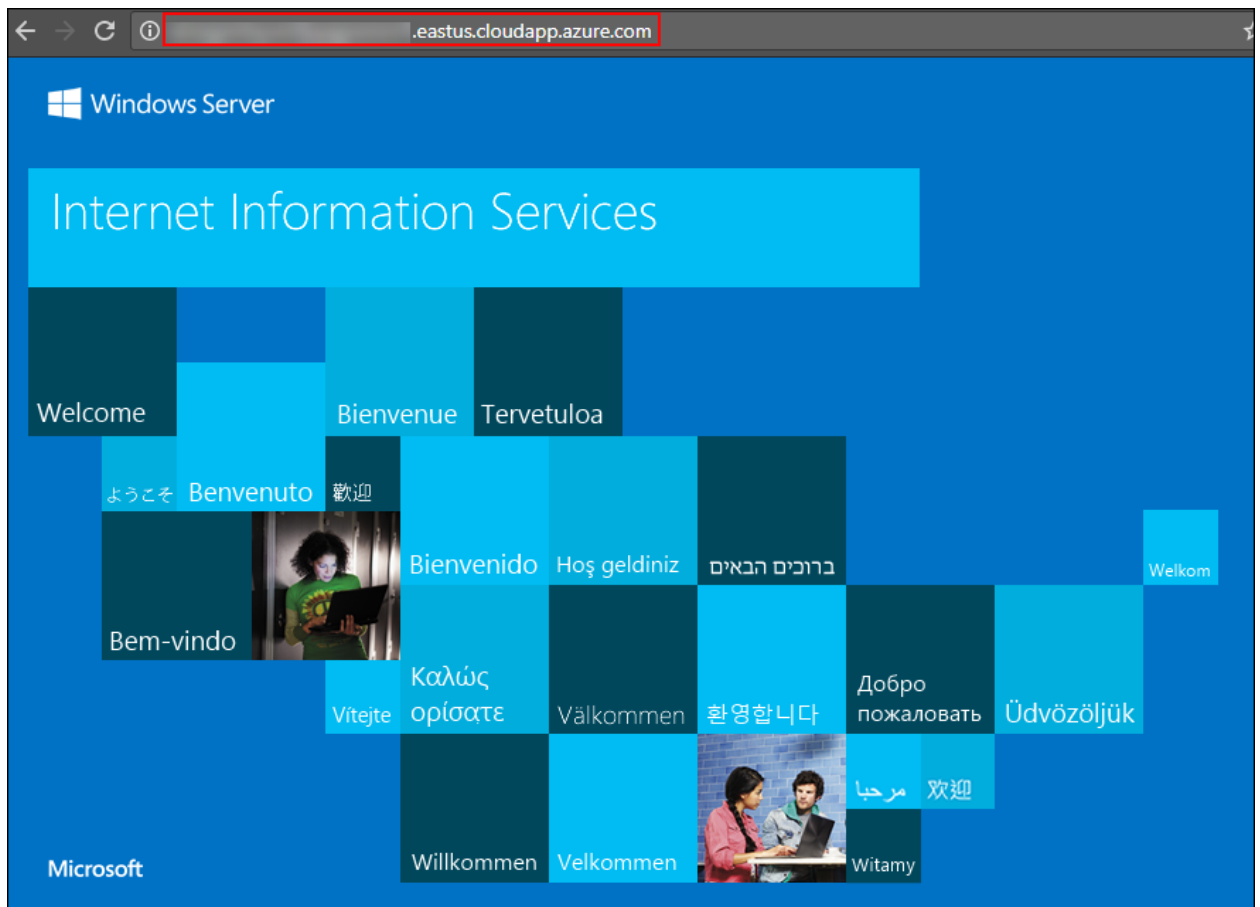
The screenshot displays the Red Hat Ansible Tower interface. On the left, the 'DETAILS' panel shows the job status as 'Successful' (highlighted with a red box). The job was started on 6/7/2017 at 00:22:58 and finished at 00:23:14. The template used is 'Install IIS', and the job type is 'Run'. The inventory is 'Agent VM Inventory', the project is 'WindowsProject', and the revision is '7a7db27d0ed98097549ef1a1873182a10704ee0'. The playbook is 'windows/enable-iis.yml', the machine credential is 'WindowsVmCredential', there are 0 forks, the limit is 'windowsVM', and the verbosity is 0 (Normal). The 'EXTRA VARIABLES' section is empty.

The main panel shows the execution details for the 'Install IIS' job. The top bar indicates 1 play, 2 tasks, 1 host, and 0 elapsed time. The search bar is empty. The execution log shows the following steps:

- 1 SSH password:
- 2
- 3 PLAY [installIIServer] *****
- 4
- 5 TASK [Gathering Facts] *****
- 6 Ok: [redacted].eastus.cloudapp.azure.com
- 7
- 8 TASK [Install IIS Web-Server with sub features and management tools] *****
- 9 Ok: [redacted].eastus.cloudapp.azure.com
- 10
- 11 PLAY RECAP *****
- 12 [redacted].eastus.cloudapp.azure.com : Ok=2 changed=0
- 13 failed=0

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.
