

Exercise 2.0 - Installing Ansible Tower

In this exercise, we are going to get Ansible Tower installed on your control node

Installing Ansible Tower

Step 1:

Use putty to SSH to your Tower node. Change directories to /tmp

```
cd /tmp
```

Step 2:

Download the latest Ansible Tower package

```
curl -O http://releases.ansible.com/ansible-tower/setup/ansible-tower-setup-latest.tar.gz
```

Step 3:

Untar and unzip the package file

```
tar xvfz /tmp/ansible-tower-setup-latest.tar.gz
```

Step 4:

Change directories into the ansible tower package

```
cd /tmp/ansible-tower-setup-*
```

Step 5:

Using an editor of your choice, open the inventory file

```
vi inventory
```

Step 6:

Fill a few variables out in an inventory file: `admin_password`, `pg_password`, `rabbitmq_password`

A few hints for **vi** if it's not your thing.

- Move around the file with your arrow keys
- **a** enters into **add** mode
- **esc** exits add mode
- **x** when not in add mode will **delete** a character
- **esc** followed by **:wq!** will save the file
 - Just doing **:q!** will exit without saving

```
[tower]
localhost ansible_connection=local

[database]

[all:vars]
admin_password='ansibleWS'

pg_host=''
pg_port=''

pg_database='awx'
pg_username='awx'
pg_password='ansibleWS'

rabbitmq_port=5672
rabbitmq_vhost=tower
rabbitmq_username=tower
rabbitmq_password='ansibleWS'
rabbitmq_cookie=cookiemonster

# Needs to be true for fqdns and ip addresses
rabbitmq_use_long_name=false
```

Step 7:

Run the Ansible Tower setup script

```
sudo ./setup.sh
```



Step 7 will take approx. 10-15 minutes to complete. This uses ansible to install Tower. This may be a good time to take a break.

Step 8:

As our git has a self signed cert, we have one more configuration step. We need to enable tower to not verify SSL for repo syncs. To do this, execute the following commands:

```
sudo su -  
echo "AWX_TASK_ENV['GIT_SSL_NO_VERIFY'] = 'True'" >> /etc/tower/settings.py  
ansible-tower-service restart  
exit
```

End Result

At this point, your Ansible Tower installation should be complete. You can access Tower from the browser on your student#-wks host at:

```
https://student#-control.rhdemo.io
```



As we're using a self-signed cert you will have to ignore the security warning and **Add** this site as trusted.

Ensuring Installation Success

You know you were successful if you are able to browse to your Ansible Tower's url and get something like this

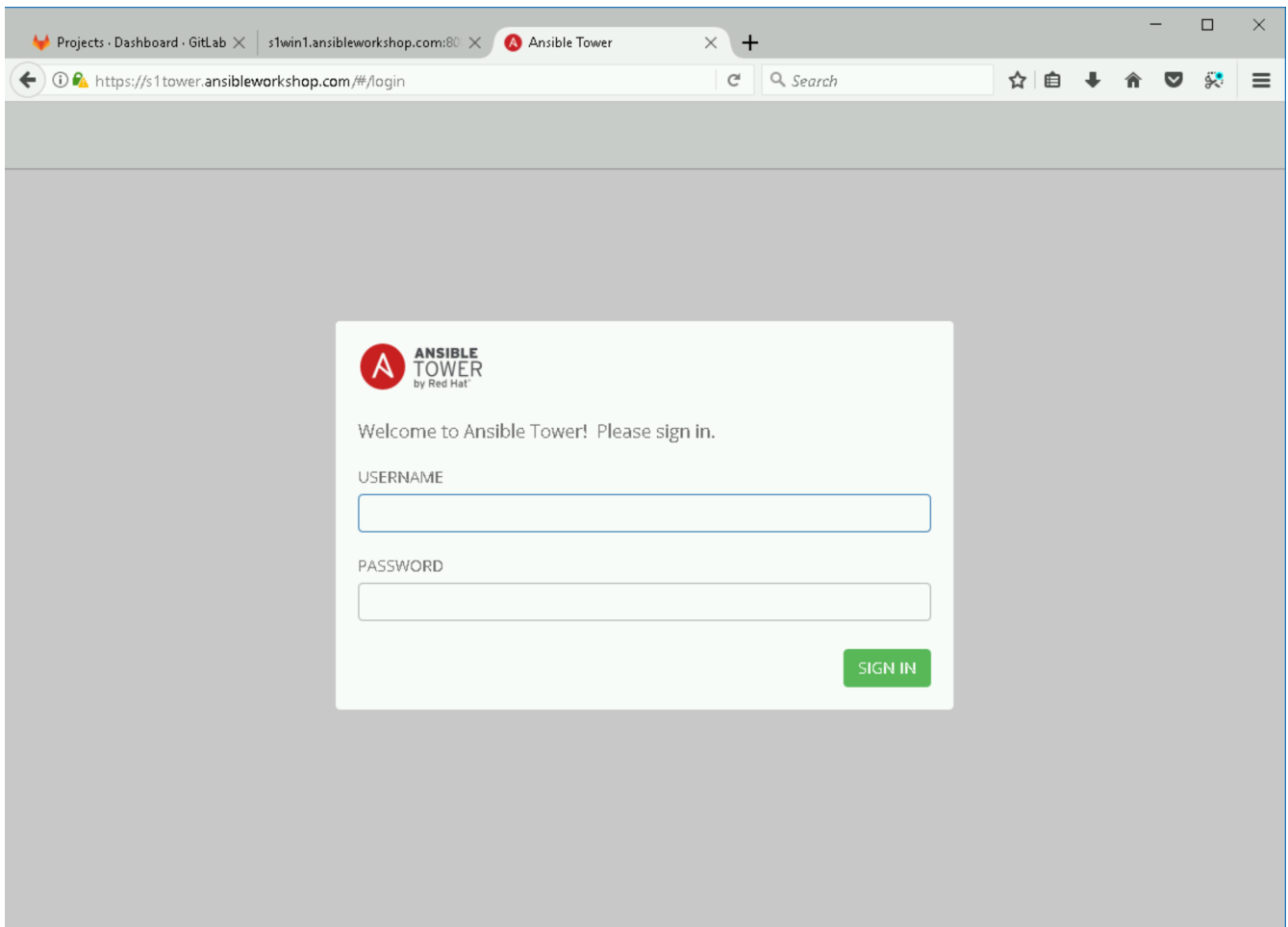


Figure 1. Ansible Tower Login Screen

Exercise 2.1 - Configuring Ansible Tower

In this exercise, we are going to configure Tower so that we can run a playbook.

Configuring Ansible Tower

There are a number of constructs in the Ansible Tower UI that enable multi-tenancy, notifications, scheduling, etc. However, we are only going to focus on a few of the key constructs that are required for this workshop today.

- Credentials
- Projects
- Inventory
- Job Template

Logging into Tower and Installing the License Key

Step 1:

To log in, use the username **admin** and the password **ansibleWS**. Note that typically AD/LDAP authentication would be setup. However, that is beyond the scope of this workshop.

As soon as you login, you will prompted to request a license or browse for an existing license file

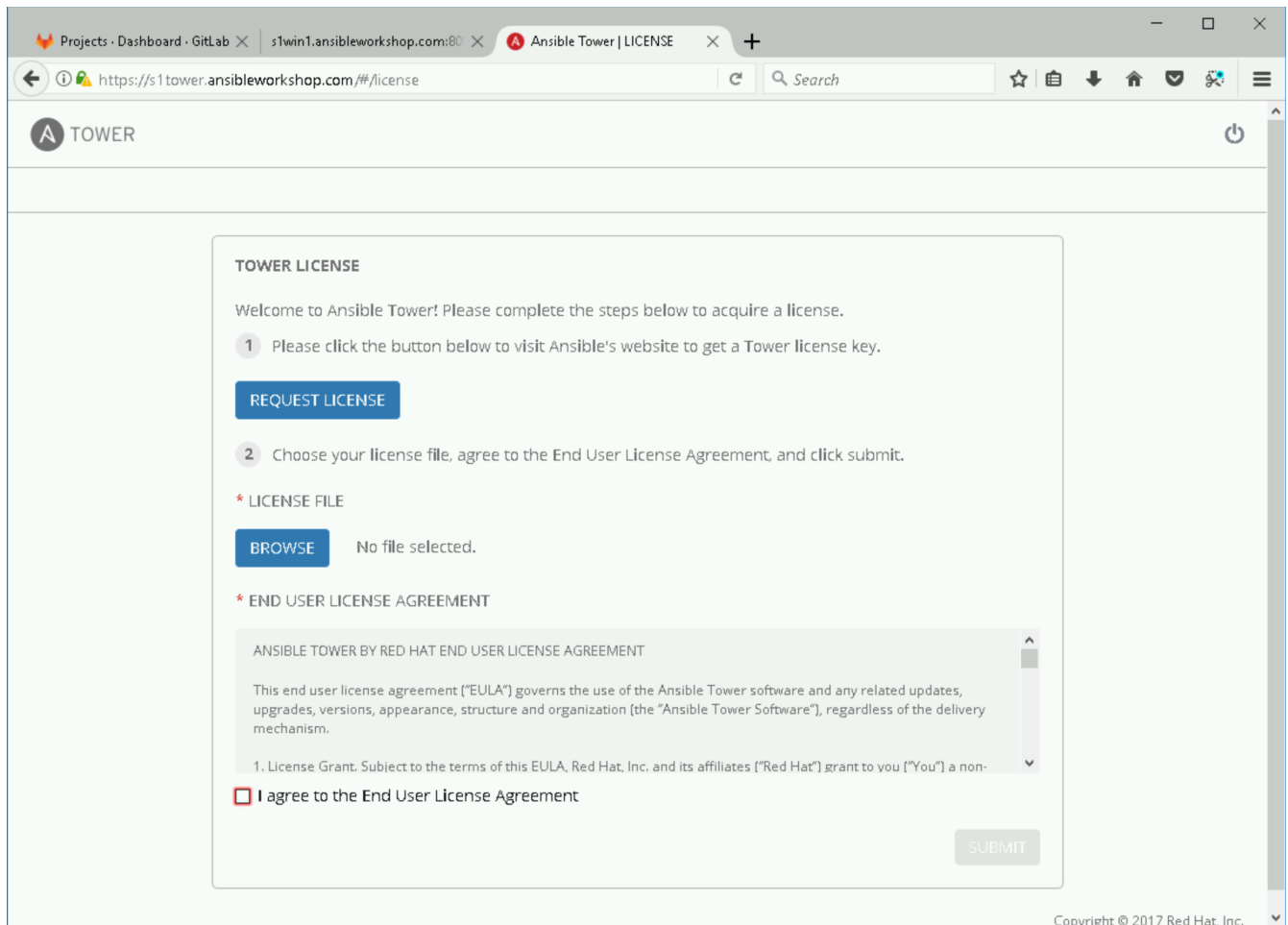


Figure 2. Uploading a License

Step 2:

In a separate browser tab, download your license key. Your license key should be provided by your instructor.

Back in the Tower UI, choose **BROWSE** and upload your recently downloaded license file into Tower. Select ***I agree to the End User License Agreement*** and click **SUBMIT**.

Creating a Machine Credential

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.

There are many [types of credentials](#) including machine, network, and various cloud providers. In this workshop, we are using a **machine** credential.

Step 1:

Select the gear icon 

Step 2:

Select **CREDENTIALS**

Step 3:

Click on **ADD** 

Step 4:

Complete the form using the following entries:

NAME	Ansible Workshop Credential
DESCRIPTION	Machine credential for run job templates during workshop
ORGANIZATION	Default
TYPE	Machine
USERNAME	student#
PASSWORD	<your AD account password - instructor provided>



Notice here we've made a change from our previous examples. Previously we were using basic authentication with a local **Administrator** account. Now we are switching to an AD user and Kerberos authentication. We will also update our inventory variables to reflect Kerberos.

Figure 3. Add Machine Credential

Step 5:

Select **SAVE** SAVE

Create an SCM Credential

Our first credential was to access our Windows machines. We need another to access our source code repository. Repeat the process as above, but with the following details:

NAME	Git Credential
DESCRIPTION	SCM credential for playbook sync
ORGANIZATION	Default
TYPE	Source Control
USERNAME	student#
PASSWORD	<your AD account password - instructor provided>

Make sure you select **SAVE**!

Figure 4. Add SCM Credential

Creating a Project

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

Step 1:

Click **PROJECTS** at the upper left

Step 2:

Select **ADD** + ADD

Step 3:

Complete the form using the following entries (using your student number)

NAME	Ansible Workshop Project
DESCRIPTION	workshop playbooks
ORGANIZATION	Default
SCM TYPE	Git
SCM URL	https://gitlab.rhdemo.io/student#/student#-playbooks.git
SCM BRANCH	<leave empty>

SCM CREDENTIAL	Git Credential
SCM UPDATE OPTIONS	<input checked="" type="checkbox"/> Clean <input checked="" type="checkbox"/> Delete on Update <input checked="" type="checkbox"/> Update on Launch

The screenshot shows the 'Define a Project' form in the Ansible Tower web interface. The form is for the 'Ansible Workshop Project' and includes the following fields and options:

- NAME:** Ansible Workshop Project
- DESCRIPTION:** workshop playbooks
- ORGANIZATION:** Default
- SCM TYPE:** Git
- SOURCE DETAILS:**
 - SCM URL:** https://gitlab.ansibleworkshop.com/student
 - SCM BRANCH/TAG/COMMIT:** (empty)
 - SCM CREDENTIAL:** Git Credential
- SCM UPDATE OPTIONS:**
 - ☒ Clean
 - ☒ Delete on Update
 - ☒ Update on Launch
- CACHE TIMEOUT (SECONDS):** 0

The form has tabs for DETAILS, PERMISSIONS, and NOTIFICATIONS. The DETAILS tab is active. The bottom right corner has CANCEL and SAVE buttons.

Figure 5. Defining a Project

Step 4:

Select SAVE SAVE

Creating a Inventory

An inventory is a collection of hosts against which jobs may be launched. 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.

An Inventory can also be imported into Tower using the **tower-manage** command and this is how we are going to add an inventory for this workshop.

Step 1:

Click **INVENTORIES**

Step 2:

Select **ADD** and select Inventory + ADD

Step 3:

Complete the form using the following entries

NAME	Ansible Workshop Inventory
DESCRIPTION	workshop hosts
ORGANIZATION	Default

The screenshot shows the 'CREATE INVENTORY' form in Ansible Tower. The form is titled 'NEW INVENTORY' and has several tabs: DETAILS, PERMISSIONS, GROUPS, HOSTS, SOURCES, and COMPLETED JOBS. The 'DETAILS' tab is selected. The form contains the following fields and values:

- NAME:** Ansible Workshop Inventory
- DESCRIPTION:** workshop hosts
- ORGANIZATION:** Default
- INSIGHTS CREDENTIAL:** (searchable field)
- INSTANCE GROUPS:** (searchable field)
- VARIABLES:** (YAML/JSON toggle, with a text area for input)

At the bottom right of the form are 'CANCEL' and 'SAVE' buttons.

Figure 6. Create an Inventory

Step 4:

Select **SAVE** SAVE

Step 5:

Using putty, login into your tower node if you closed the window previously

```
student#-control.rhdemo.io
```

Step 6:

Use the `tower-manage` command to import an existing inventory. (Be sure to replace `<username>` with your actual username)

```
sudo tower-manage inventory_import
--source=/home/student#/lightbulb/lessons/lab_inventory/student#-instances.txt
--inventory-name="Ansible Workshop Inventory"
```

You should see output similar to the following:

```
[student50@ip-10-0-0-161 ~]$ sudo tower-manage inventory_import --source=/home/student50/lightbulb/lessons/lab_inventory/student
50-instances.txt --inventory-name="Ansible Workshop Inventory"
1.302 INFO      Updating inventory 2: Ansible Workshop Inventory
1.349 INFO      Reading INI source: /home/student50/lightbulb/lessons/lab_inventory/student50-instances.txt
1.350 INFO      Loaded 2 groups, 3 hosts
1.351 INFO      Inventory variables unmodified
1.360 INFO      Group "control" added
1.366 INFO      Group "web" added
1.372 INFO      Host "control" added
1.374 INFO      Host "node-1" added
1.376 INFO      Host "node-2" added
1.386 INFO      Host "control" added to group "control"
1.392 INFO      Host "node-1" added to group "web"
1.392 INFO      Host "node-2" added to group "web"
1.465 INFO      Inventory import completed for "Ansible Workshop Inventory" (id=2) in 0.2s
[student50@ip-10-0-0-161 ~]$
```

Figure 7. Importing an inventory with tower-manage

Feel free to browse your inventory in Tower. You should now notice that the inventory has been populated with Groups and that each of those groups contain hosts.

INVENTORIES / Ansible Workshop Inventory

The screenshot displays the Ansible Tower web interface for the 'Ansible Workshop Inventory'. It is divided into two main panels: 'GROUPS' and 'HOSTS'. The 'GROUPS' panel on the left shows two groups: 'control' and 'web', each with a search bar and a 'KEY' button. The 'HOSTS' panel on the right shows three hosts: 'control', 'node-1', and 'node-2', each with a search bar and a 'KEY' button. Both panels include a 'RUN COMMANDS' button and a '+ ADD GROUP' or '+ ADD HOST' button. The 'GROUPS' panel also has a 'SYSTEM TRACKING' button. The 'HOSTS' panel also has a 'SYSTEM TRACKING' button. The 'GROUPS' panel shows 'ITEMS 1 - 2 OF 2' and the 'HOSTS' panel shows 'ITEMS 1 - 3 OF 3'.

Figure 8. Inventory with Groups

End Result

At this point, we are doing with our basic configuration of Ansible Tower. In exercise 2.2, we will be solely focused on creating and running a job template so you can see Tower in action.

Exercise 2.2 - Ad-hoc commands in Tower

In this exercise, you will run ad-hoc modules from Tower.

Run setup module on windows nodes

Step 1:

In Tower UI, click **INVENTORIES**

Step 2:

Click **Ansible Workshop Inventory**.

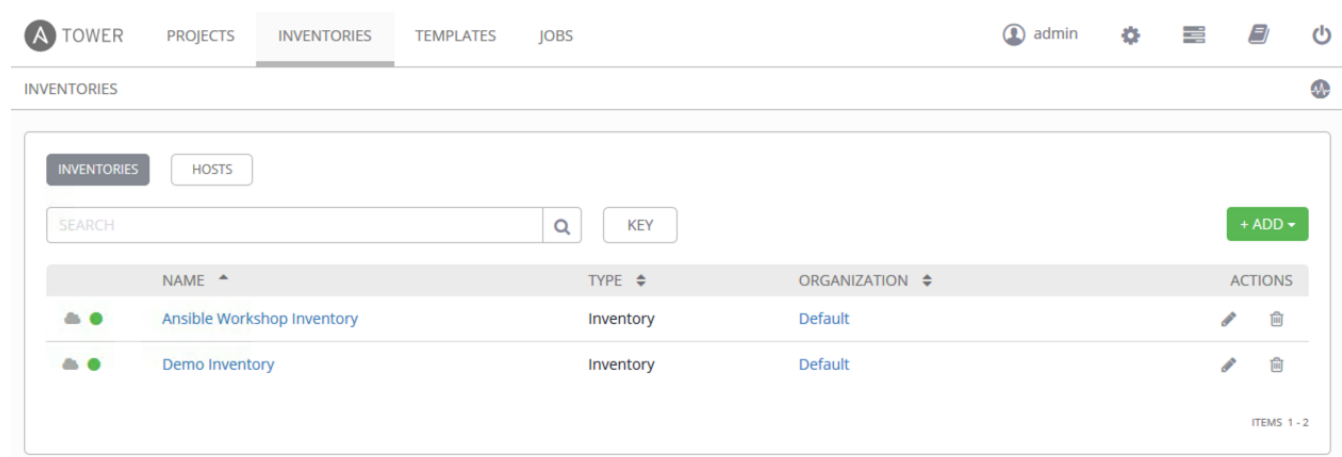


Figure 9. Select Ansible Workshop Inventory

Step 3:

Click **GROUPS**.

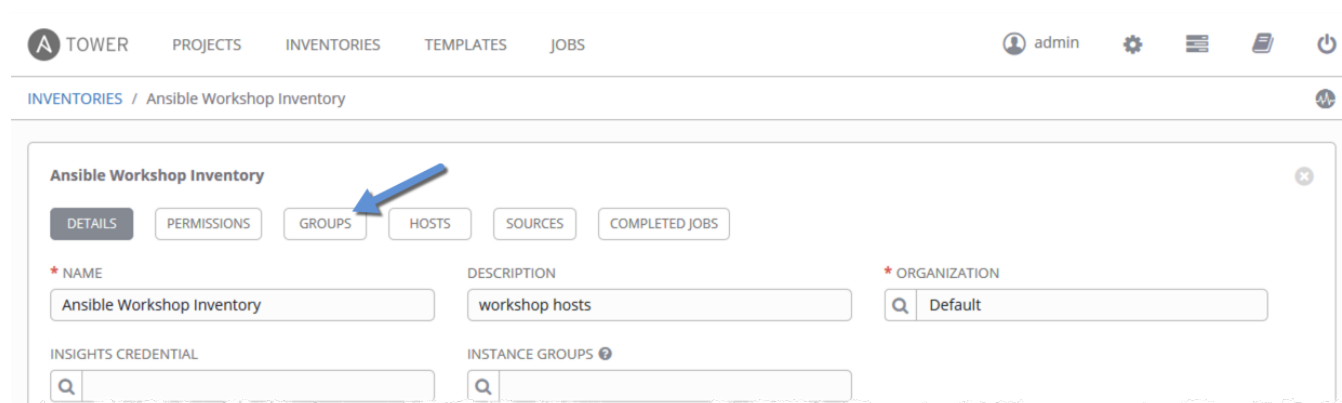


Figure 10. Select GROUPS

Step 4:

Select the check box next to **windows** and click **RUN COMMANDS**.

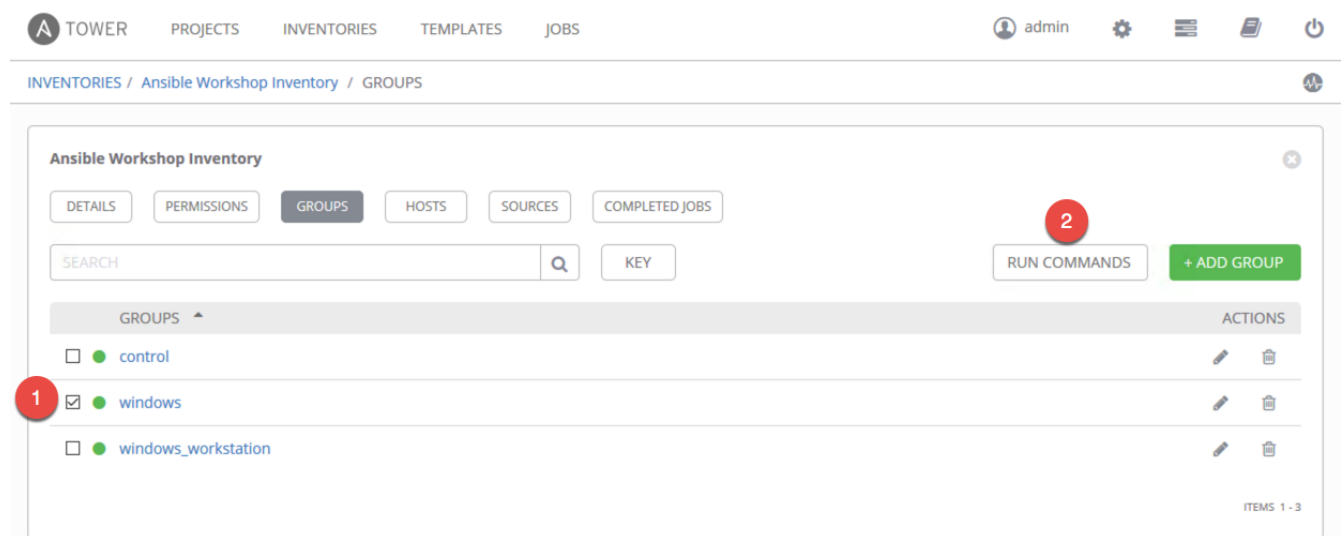


Figure 11. Select windows group

Step 5:

Select the following values and click **LAUNCH**:

Module	win_ping
Machine Credential	Ansible Workshop Credentials

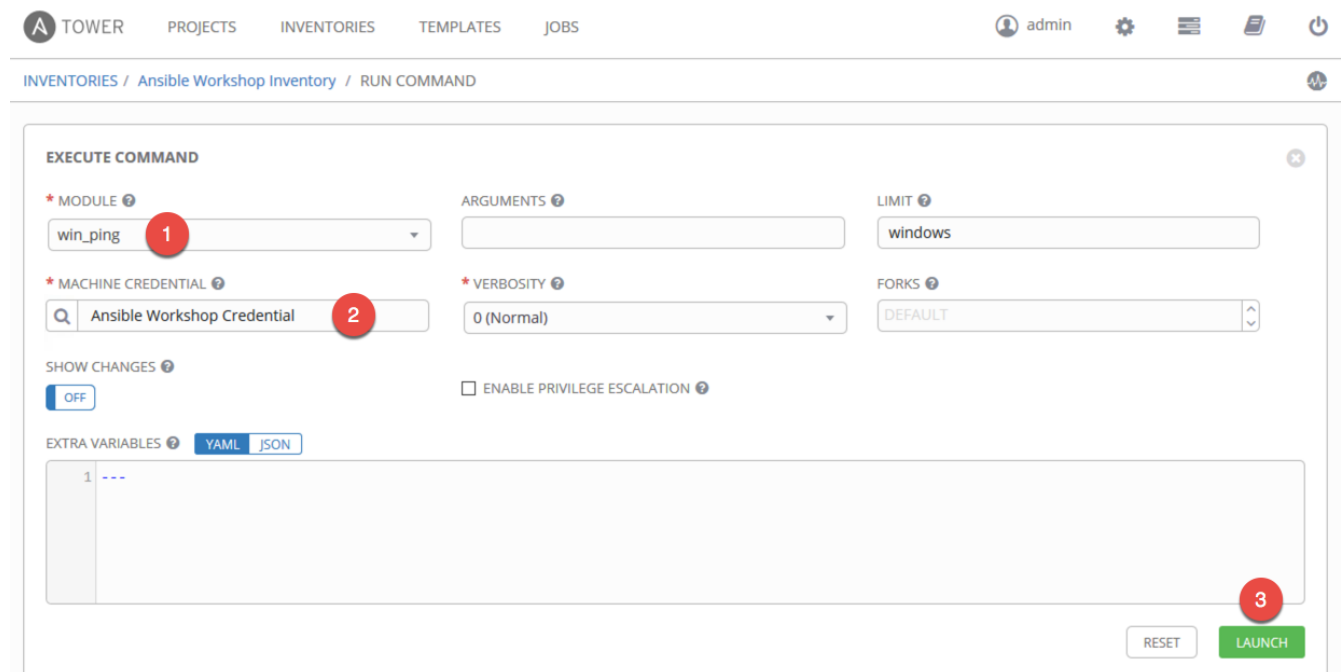


Figure 12. Run command

Step 6:

Review the results:

The screenshot shows the Tower web interface. At the top, there is a navigation bar with links for TOWER, PROJECTS, INVENTORIES, TEMPLATES, and JOBS. On the right, there is a user profile for 'admin' and several icons for settings, menu, and power. Below the navigation bar, the breadcrumb 'JOBS / win_ping' is visible. The main content area is divided into two panels. The left panel, titled 'RESULTS', contains a table with the following information: NAME (win_ping), STATUS (Successful), STARTED (3/11/2018 6:32:15 AM), FINISHED (3/11/2018 6:32:18 AM), ELAPSED (3.358 seconds), INVENTORY (Ansible Workshop Inventory), CREDENTIAL (Ansible Workshop Credential), LAUNCHED BY (admin), FORKS (0), LIMIT (windows), and VERBOSITY (0). Below this table is a section for 'EXTRA VARIABLES' with a single variable '1' and a value '---'. The right panel, titled 'STANDARD OUT', displays the output of the job, which includes the text 'SSH password:' followed by two lines of JSON output: 'student1-node1 | SUCCESS => { "changed": false, "ping": "pong" }' and 'student1-node2 | SUCCESS => { "changed": false, "ping": "pong" }'.

Figure 13. Run command result

End Result

In this section, you ran a module against your inventory. This could be useful when you need to run quick actions on target systems.

Exercise 2.3 - Creating and Running a Job Template

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

Creating a Job Template

Step 1:

Select **TEMPLATES**.

Step 2:

Click **ADD** , and select **JOB TEMPLATE**.

Step 3:

Complete the form using the following values

NAME	IIS Basic Job Template
DESCRIPTION	Template for the iis-basic-playbook
JOB TYPE	Run
INVENTORY	Ansible Workshop Inventory
PROJECT	Ansible Workshop Project
PLAYBOOK	iis-basic-playbook/site.yml
MACHINE CREDENTIAL	Ansible Workshop Credential
LIMIT	windows
OPTIONS	<input checked="" type="checkbox"/> Enable Privilege Escalation

TOWER

PROJECTS

INVENTORIES

TEMPLATES

JOBS

admin

TEMPLATES / IIS Basic Job Template

IIS Basic Job Template

DETAILSDetailsPERMISSIONSPERMISSIONSNOTIFICATIONSNOTIFICATIONSCOMPLETED JOBSCOMPLETED JOBSADD SURVEYADD SURVEY

*

NAME

IIS Basic Job Template

*

INVENTORY

Ansible Workshop Inventory

PROMPT ON LAUNCH

*

CREDENTIAL

MACHINE: Ansible Workshop Credential

PROMPT ON LAUNCH

*

VERBOSITY

0 (Normal)

PROMPT ON LAUNCH

Skip Tags

PROMPT ON LAUNCH

Options

☐ Enable Privilege Escalation

☐ Allow Provisioning Callbacks

☐ Enable Concurrent Jobs

☐ Use Fact Cache

EXTRA VARIABLES

YAMLYAMLJSONJSON

PROMPT ON LAUNCH

DESCRIPTION

Template for the iis-basic-playbook

* PROJECT

Ansible Workshop Project

FORKS

DEFAULT

INSTANCE GROUPS

JOB TYPE

Run

PROMPT ON LAUNCH

* PLAYBOOK

iis_basic_playbook/site.yml

LIMIT

windows

PROMPT ON LAUNCH

JOB TAGS

PROMPT ON LAUNCH



SHOW CHANGES

OFF

PROMPT ON LAUNCH

Figure 14. Job Template Form

Step 4:

Click **SAVE**  and then select **ADD SURVEY** 

15

Step 5:

Complete the survey form with following values

PROMPT	Please enter a test message for your new website
DESCRIPTION	Website test message prompt
ANSWER VARIABLE NAME	iis_test_message
ANSWER TYPE	Text
MINIMUM/MAXIMUM LENGTH	Use the defaults
DEFAULT ANSWER	Be creative, keep it clean, we're all professionals here

ADD SURVEY PROMPT

*** PROMPT**

Please enter a test message for your new website

DESCRIPTION

Website test message prompt

*** ANSWER VARIABLE NAME ?**

iis_test_message

*** ANSWER TYPE ?**

Text

MINIMUM LENGTH

0

MAXIMUM LENGTH

1024

DEFAULT ANSWER

Be creative, keep it clean, we're all professionals here

☒ **REQUIRED**

CLEAR

+ ADD

Figure 15. Survey Form

Step 6:

Select ADD + ADD

Step 7:

Select SAVE SAVE

Step 8:

Back on the main Job Template page, select SAVE  again.

Running a Job Template

Now that you've successfully creating your Job Template, you are ready to launch it. Once you do, you will be redirected to a job screen which is refreshing in realtime showing you the status of the job.

Step 1:

Select TEMPLATES



Alternatively, if you haven't navigated away from the job templates creation page, you can scroll down to see all existing job templates

Step 2:

Click on the rocketship icon  for the **IIS Basic Job Template**

Step 3:

When prompted, enter your desired test message

LAUNCH JOB | IIS Basic Job Template



SURVEY

* PLEASE ENTER A TEST MESSAGE FOR YOUR NEW WEBSITE

Website test message prompt

Be creative, keep it clean, we're all professionals here

INVENTORY

CREDENTIAL

Ansible Workshop InventoryMachine: Ansible Workshop Credential

CANCEL

LAUNCH

Figure 16. Survey Prompt

Step 4:

Select LAUNCH 

Step 5:

Sit back, watch the magic happen

One of the first things you will notice is the summary section. This gives you details about your job such as who launched it, what playbook it's running, what the status is, i.e. pending, running, or complete.

The screenshot displays the Ansible Tower web interface. At the top, navigation tabs include TOWER, PROJECTS, INVENTORIES, TEMPLATES, and JOBS. The user is logged in as 'admin'. The breadcrumb trail shows 'JOBS / 21 - IIS Basic Job Template'.

DETAILS

- STATUS: Successful
- STARTED: 3/11/2018 12:53:21 AM
- FINISHED: 3/11/2018 12:53:41 AM
- TEMPLATE: IIS Basic Job Template
- JOB TYPE: Run
- LAUNCHED BY: admin
- INVENTORY: Ansible Workshop Inventory
- PROJECT: Ansible Workshop Project
- REVISION: 5baf3a9
- PLAYBOOK: iis_basic_playbook/site.yml
- MACHINE CREDENTIAL: Ansible Workshop Credential
- FORKS: 0
- LIMIT: windows
- VERBOSITY: 0 (Normal)
- INSTANCE GROUP: tower

EXTRA VARIABLES

```
1 iis_test_message: 'Be creative, keep it
```

IIS Basic Job Template

PLAYS 1 TASKS 6 HOSTS 2 ELAPSED 00:00:20

SEARCH [] KEY []

- 1 SSH password:
- 2
- 3 PLAY [This is my role-based playbook] ***** 00:53:26
- 4
- 5 TASK [Gathering Facts] ***** 00:53:26
- 6 ok: [student1-node1]
- 7 ok: [student1-node2]
- 8
- 9 TASK [iis_simple : Install IIS] ***** 00:53:28
- 10 ok: [student1-node1]
- 11 ok: [student1-node2]
- 12
- 13 TASK [iis_simple : Create site directory structure] ***** 00:53:30
- 14 ok: [student1-node1] => (item={u'path': u'C:\\sites\\playbooktest', u'name': u'Ansible Playbook Test', u'port': u'8080'})
- 15 ok: [student1-node2] => (item={u'path': u'C:\\sites\\playbooktest', u'name': u'Ansible Playbook Test', u'port': u'8080'})

Copyright © 2017 Red Hat, Inc.

Figure 17. Job Summary

To the left, you will be able to see details on the play.

DETAILS



STATUS	● Successful
STARTED	3/11/2018 12:53:21 AM
FINISHED	3/11/2018 12:53:41 AM
TEMPLATE	IIS Basic Job Template
JOB TYPE	Run
LAUNCHED BY	admin
INVENTORY	Ansible Workshop Inventory
PROJECT	● Ansible Workshop Project
REVISION	5baf3a9
PLAYBOOK	iis_basic_playbook/site.yml
MACHINE CREDENTIAL	Ansible Workshop Credential
FORKS	0
LIMIT	windows
VERBOSITY	0 (Normal)
INSTANCE GROUP	tower

EXTRA VARIABLES

```
1 iis_test_message: 'Be creative, keep it  
2
```

Figure 18. Play and Task Details

To the right, you can view standard output; the same way you could if you were running Ansible Core from the command line.

```

+ -
1  SSH password:
2
3  PLAY [This is my role-based playbook] ***** 00:53:26
4
5  TASK [Gathering Facts] ***** 00:53:26
6  ok: [student1-node1]
7  ok: [student1-node2]
8
9  TASK [iis_simple : Install IIS] ***** 00:53:28
10 ok: [student1-node1]
11 ok: [student1-node2]
12
13 TASK [iis_simple : Create site directory structure] ***** 00:53:30
14 ok: [student1-node1] => (item={u'path': u'C:\\sites\\playbooktest', u'name': u'Ansible Playbook
15 ok: [student1-node2] => (item={u'path': u'C:\\sites\\playbooktest', u'name': u'Ansible Playbook
    Test', u'port': u'8080'})
    Test', u'port': u'8080'})

```

Figure 19. Job Standard Output

Step 6:

Once your job is successful, navigate to your new website

`http://student#-node1.rhdemo.io`

If all went well, you should see something like this, but with your own custom message of course.



STUDENT1-NODE1 --- Be creative, keep it clean, we're all professionals here

Figure 20. New Website with Personalized Test Message

End Result

At this point in the workshop, you've experienced the core functionality of Ansible Tower. But wait... there's more! You've just begun to explore the possibilities of Ansible Core and Tower. Take a

look at the resources page in this guide to explore some more features.