



# Red Hat Ansible Tower Workshop.

Workshop's Guide  
(The Language of DevOps Automation)

Prepared by: Robert Calva – LATAM Sr. Solutions Architect - Mgmt & Automation  
Modified by: Robert Calva – LATAM Sr. Solutions Architect - Mgmt & Automation

Creation date: 01/25/17  
Date of last modification: 01/08/19  
Version: 2.2

# INDEX

<b>REVIEW</b>	<b>2</b>
<b>SUMMARY</b>	<b>3</b>
Business Scenario	3
<b>Workshop's Architecture</b>	<b>3</b>
Workshop's Design	3
<b>WORKSHOP: RED HAT ANSIBLE TOWER</b>	<b>4</b>
Lab 1: Setting Up Ansible Tower: Organizations, Teams, Users and Credentials	4
Lab 2: Creating Ansible Tower Projects, Inventories and Job Templates	9
Lab 3: Automating IT Process using Ansible Tower Jobs	16
Lab 4: Automating IT Process Using Ansible Tower Workflows	20
Lab 5: Running Ad-Hoc Commands Using Ansible Tower	23
Lab 6: Automating Windows	25
Lab 7: Automating Network Devices	29

## REVIEW



### Change's log

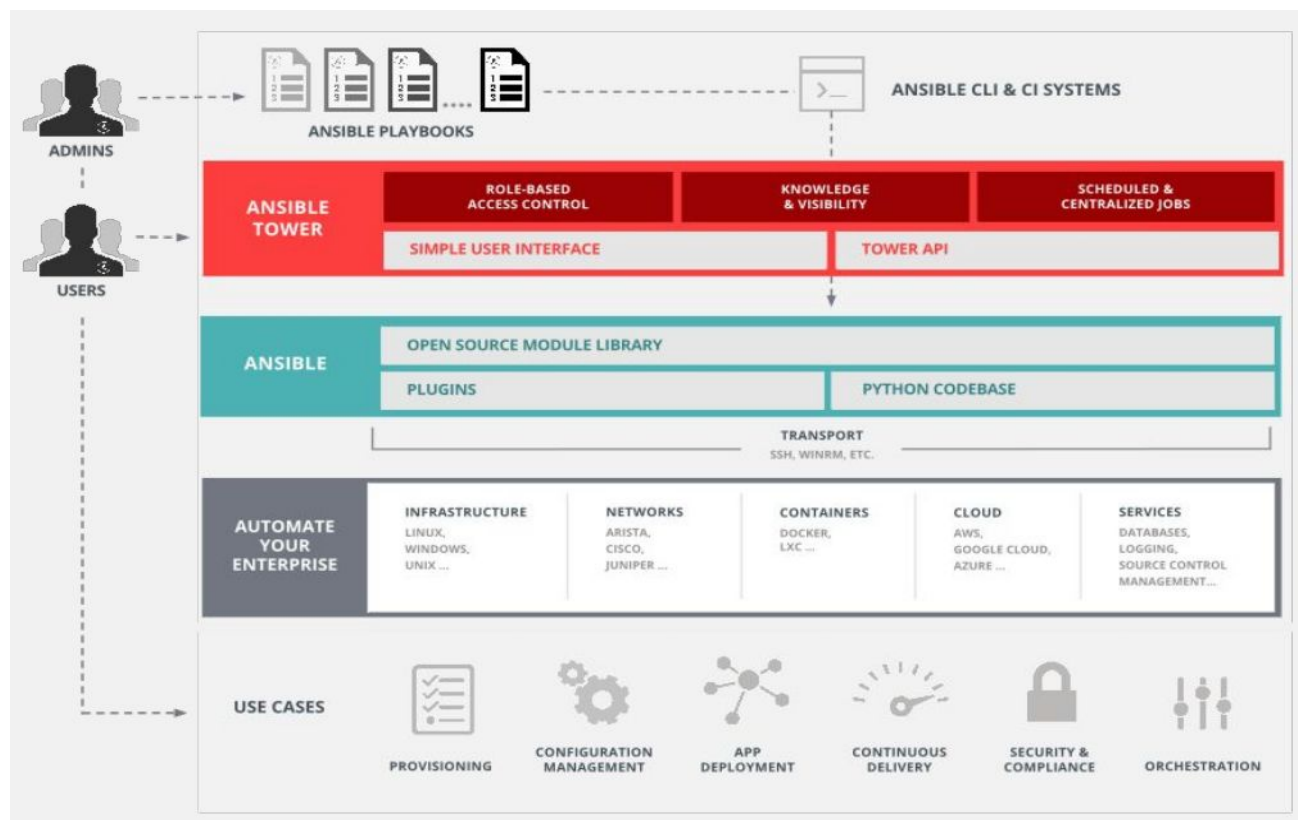
Fecha	Autor	Versión	Referencias
Sep/22/1017	Robert J. Calva	2.1	v2.0
Jan/08(2019	Robert J. Calva	2.2	v2.1

## SUMMARY

### Business Scenario

**Ansible** is the first **automation language** that can be read and written across IT. **Ansible** is the only **automation engine** that can automate the entire and continuous **delivery pipeline**.

### Workshop's Architecture



### Workshop's Design

This workshop has hands-on labs and presentations to provide a basic first-hand contact with the product:

- Introductions ~ 30 min

- Ansible Tower General Presentation ~ 30 min

- Exploring the Dashboard and Tower Interface ~ 30 min

- Setting Up Ansible Tower: Organizations, Teams, Users and Credentials **(With Lab)** ~ 40 min

- Creating Ansible Tower Projects, Inventories and Job Templates **(With Lab)** ~ 40 min

- Automating IT Process using Ansible Tower Jobs **(with Lab)** ~ 40 min

- Automating IT Process using Workflows **(with Lab)** ~ 40 min

- Running Ad-Hoc Commands using Ansible Tower **(with Lab)** ~ 40 min

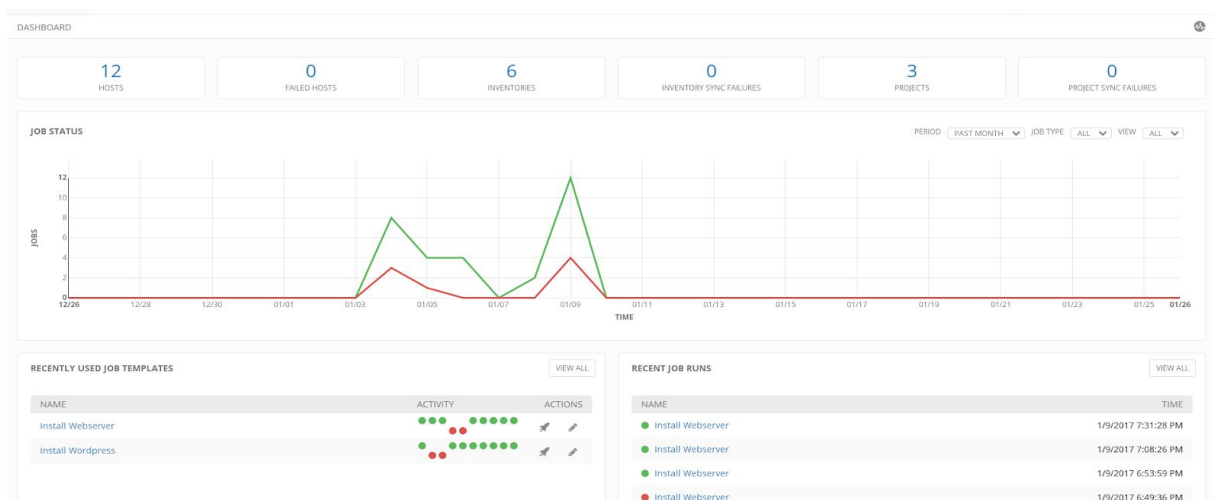
- Automating Windows **(with Lab)** ~ 40 min

- Automating Network Devices **(with Lab)** ~ 40 min

## WORKSHOP: RED HAT ANSIBLE TOWER

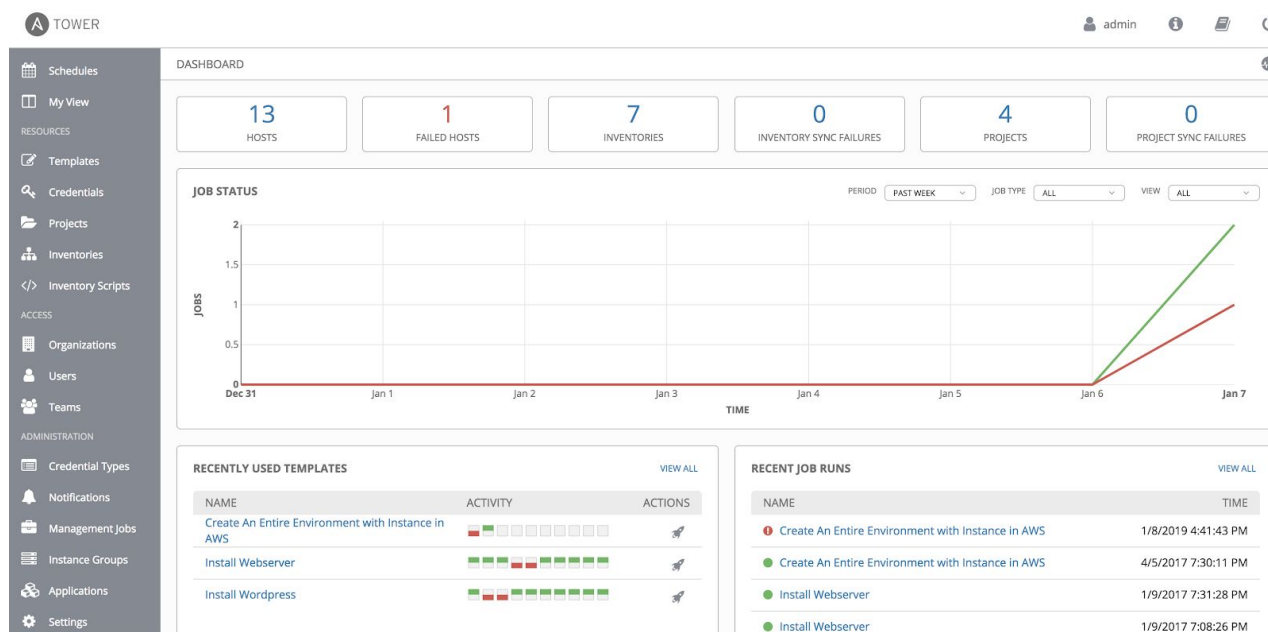
Access to Ansible Tower Web Portal,

User: admin / Password: Redhat1!




### Lab 1: Setting Up Ansible Tower: Organizations, Teams, Users and Credentials

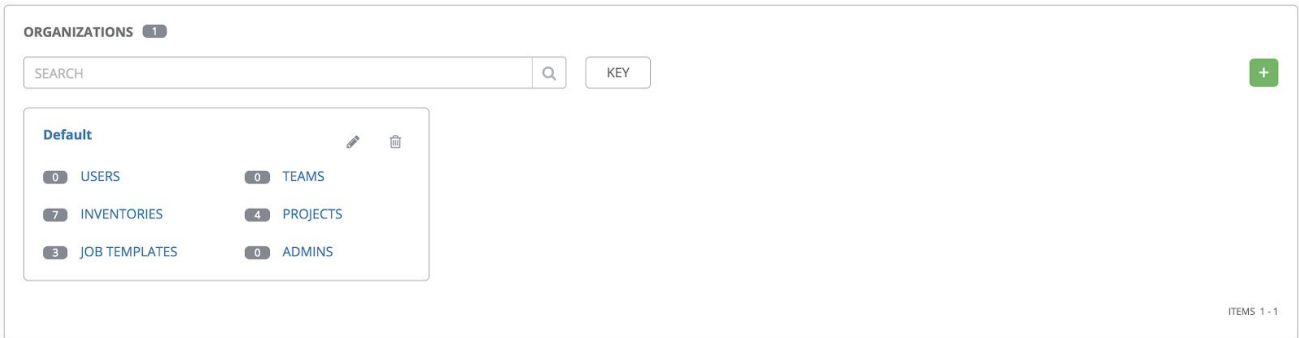
A huge part of the *Ansible Tower* value proposition is being able to limit what resources certain end users have access to via Organizations. Being able to manage multiple environments from a single pane of glass, as the admin user, but still limit what other end users can access is a major selling point for *Ansible Tower*. Another advantage is being able to utilize Ansible Tower to control and automate process via Ansible Jobs.



This lab will mainly be used as an introduction to the structure of the product, where users can create their own Organizations, Teams, Users and Credentials within those Organizations to be used for automate process.

## 1A. Create an Organization:

Navigate within the **Ansible Tower Portal** and click on **Organizations** (within ACCESS Section). Click on  to add a new Organization:



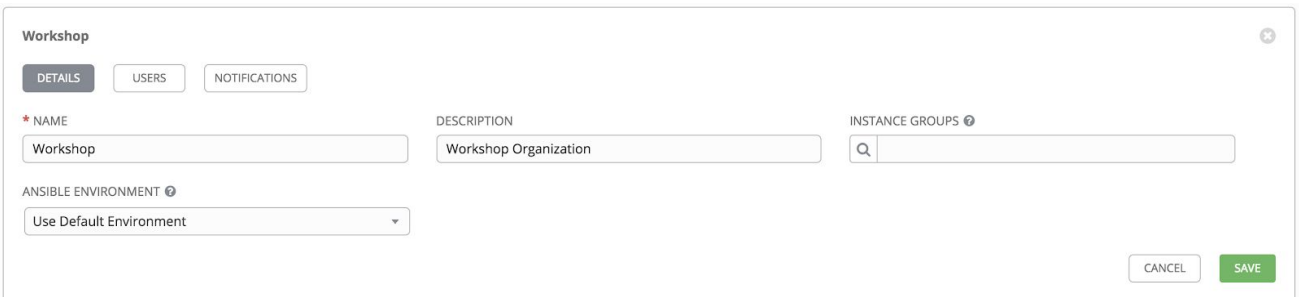
Create the new Organization as follows:

**Name:** Workshop


**Description:** Workshop Organization

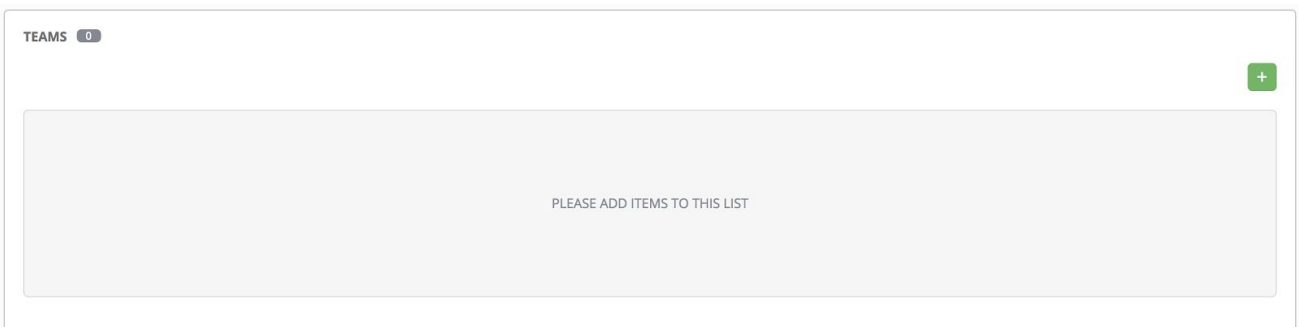
**Ansible Environment:** Use Default Environment

Click on **SAVE** to save the changes.



## 1B. Create a Team:

Navigate within the **Ansible Tower Portal** and click on **Teams** (within ACCESS Section). Click on  to add a new Team:



Create the new Team as follows:

**Name:** Workshop Team

**Description:** Workshop Team

**Organization:** Workshop

Click on **SAVE** to save the changes.

Workshop Team

DETAILS

USERS

PERMISSIONS

\* NAME

Workshop Team

DESCRIPTION

Workshop Team

\* ORGANIZATION

Q Workshop

CANCEL

SAVE

### 1C. Create a Normal User:

Navigate within the **Ansible Tower Portal** and click on **Users** (within ACCESS Section). Click on  to add a new User:


USERS


USERS 1

SEARCH

Q

KEY



USERNAME ^	FIRST NAME ^	LAST NAME ^	ACTIONS
admin	Ansible Tower	Admin	

ITEMS 1 - 1

Create the new User as follows:

**First Name:** Workshop

**Last Name:** User

**EMAIL:** workshopuser@example.com

**Username:** wuser

**Organization:** Workshop

**Password:** Redhat1!

**User Type:** Normal User

Click on **SAVE** to save the changes.

NEW USER

DETAILS

ORGANIZATIONS

TEAMS

PERMISSIONS

FIRST NAME

Workshop

LAST NAME

User

\* ORGANIZATION

Q Workshop

\* EMAIL

workshopuser@example.com

\* USERNAME

wuser

\* PASSWORD

SHOW .....

\* CONFIRM PASSWORD

SHOW .....


USER TYPE

Normal User

CANCEL

SAVE

### 1D. Create a System Admin User:

Navigate within the **Ansible Tower Portal** and click on **Users** (within ACCESS Section). Click on  to add a new User.

Create the new User as follows:

**First Name:** Workshop

**Last Name:** Admin

**EMAIL:** workshopadmin@example.com

**Username:** wadmin

**Organization:** Workshop

**Password:** Redhat1!

**User Type:** System Administrator

Click on **SAVE** to save the changes.

**NEW USER** ADMIN

DETAILS ORGANIZATIONS TEAMS PERMISSIONS

FIRST NAME: Workshop

LAST NAME: Admin

\* ORGANIZATION: Workshop

\* EMAIL: workshopadmin@example.com

\* USERNAME: wadmin

\* PASSWORD: SHOW [masked]

\* CONFIRM PASSWORD: SHOW [masked]

USER TYPE: System Administrator

CANCEL SAVE

### 1E. Create an Auditor User:

Navigate within the **Ansible Tower Portal** and click on **Users** (within ACCESS Section). Click on  to add a new User.

Create the new User as follows:

**First Name:** Workshop  
**Last Name:** Auditor  
**EMAIL:** workshopauditor@example.com  
**Username:** wauditor  
**Organization:** Workshop  
**Password:** Redhat1!  
**User Type:** System Auditor

Click on **SAVE** to save the changes.

**1F.** Now is time to **modify roles** for our **Workshop User (wuser)** to be able to create objects within our new Workshop Organization as an Admin. First Log Out, and then access to **Ansible Tower Portal**, using our new **Workshop System Administrator**:

**User:** wadmin / **Password:** Redhat1!

Go to **Organizations** (within ACCESS Section) and then click on **USERS** within **WORKSHOP** Organization:

WORKSHOP	
Workshop Organization	
3	USERS
1	TEAMS
0	INVENTORIES
0	PROJECTS
0	JOB TEMPLATES
0	ADMINS


Workshop

DETAILS USERS NOTIFICATIONS


SEARCH [ ] KEY

USER	ROLE
admin	SYSTEM ADMINISTRATOR
wadmin	MEMBER SYSTEM ADMINISTRATOR
wauditor	MEMBER SYSTEM AUDITOR
wuser	MEMBER

ITEMS 1 - 4

Click on  and then select (check the checkbox) the user **wuser** from the list to add a new role:

1 Please select Users from the list below.



USERNAME ^	FIRST NAME ^	LAST NAME ^
<input type="checkbox"/> admin	Ansible Tower	Admin
<input type="checkbox"/> wadmin	Workshop	Admin
<input type="checkbox"/> wauditor	Workshop	Auditor
<input checked="" type="checkbox"/> wuser	Workshop	User

Select the **Admin** role within SELECT ROLES section:

Add Users

assign roles to the selected users/teams

KEY

Workshop User USER

SELECT ROLES

Auditor

Admin

Member

Read

SAVE

Click on **Save** to commit the changes:

2 Please assign roles to the selected users/teams

KEY

Workshop User USER
























Admin

CANCEL

SAVE

## 1G. Create Credentials:

Navigate within the **Ansible Tower Portal** and click on **Credentials** (within RESOURCES Section). Click on  to add new Credentials:

CREDENTIALS 7 			
<input type="text" value="SEARCH"/> 		KEY	
NAME ^	KIND	OWNERS	ACTIONS
AWS Credentials	Amazon Web Services	Default	  
CloudForms Credentials	Red Hat CloudForms	Default	  
Localhost Credentials	Machine	Default	  
OpenStack Admin Tenant	OpenStack	admin	  
SSH OSP Servers	Machine	admin	  
SSH Physical Servers	Machine	Default	  
VMware Credentials	VMware vCenter	admin	  



Create the new Credentials as follows:

**Name:** SSH Server Credentials  
**Description:** SSH Server Credentials  
**Organization:** Workshop  
**Type:** Machine  
**Username:** root  
**Password:** Redhat1!

Click on **SAVE** to save the changes.

#### NEW CREDENTIAL

DETAILS

PERMISSIONS

\* NAME ?

SSH Server Credentials

DESCRIPTION ?

SSH Server Credentials

ORGANIZATION ?

Q Workshop

\* CREDENTIAL TYPE ?

Q Machine

#### TYPE DETAILS

USERNAME

root

PASSWORD

☐ Prompt on launch

SHOW

\*\*\*\*\*

## Lab 2: Creating Ansible Tower Projects, Inventories and Job Templates

Provisioning refers to the capacity an infrastructure has to deliver a resource and manage its life-cycle.

**2A.** Access to **Ansible Tower Portal**, using our new **Workshop User**:

**User:** wuser / **Password:** Redhat1!

#### DASHBOARD

0

HOSTS

0

FAILED HOSTS

0

INVENTORIES

0

INVENTORY SYNC FAILURES

0

PROJECTS

0

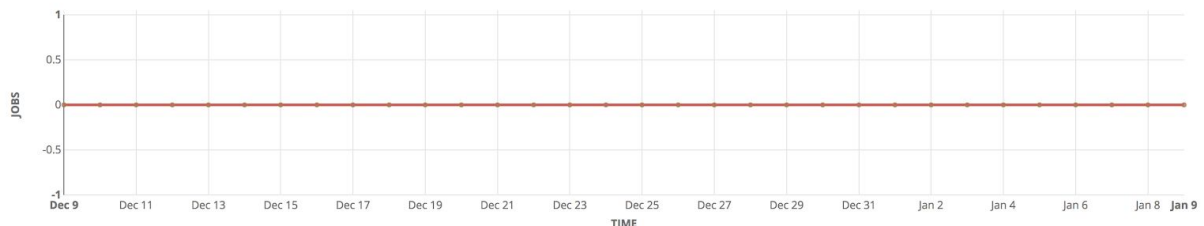
PROJECT SYNC FAILURES

#### JOB STATUS

PERIOD PAST MONTH

JOB TYPE ALL

VIEW ALL



#### RECENTLY USED JOB TEMPLATES

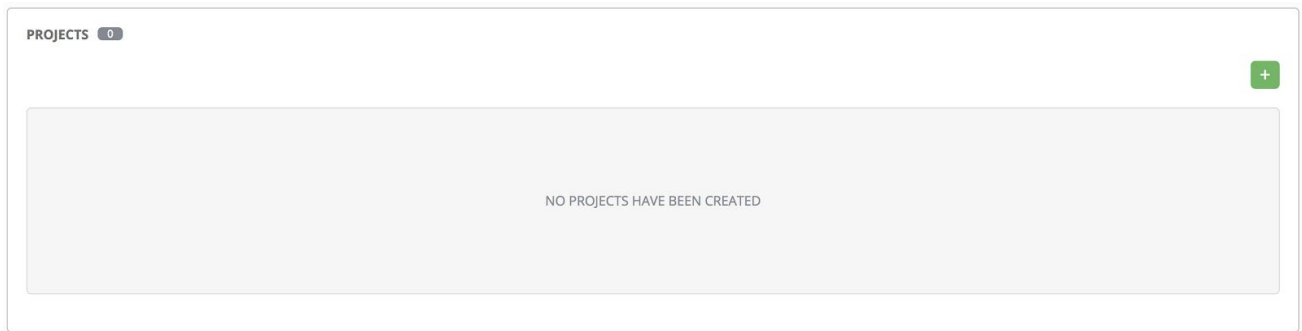
No job templates were recently used.  
You can create a job template [here](#).


#### RECENTLY RUN JOBS

No jobs were recently run.

**NOTE:** You can see that you have no hosts, no projects, no Inventories, etc. because we are working within our new **Workshop Organization**.

## 2B. Create 2 new Projects:



Click on **Projects** (within RESOURCES Section) and then click on  button. Create the new App Server Project using next information:

**NAME:** App Project

**DESCRIPTION:** An Application Server Project

**ORGANIZATION:** Workshop


**SCM TYPE:** Git

**SOURCE DETAILS (SCM URL):** <https://github.com/rcalvaga/examples-ansible.git>

**SCM UPDATE OPTIONS:**

- ☐ Clean ?
- ☐ Delete on Update ?
- ☒ Update Revision on Launch ?

Click on **SAVE** to create the App Server Project.

Click on **Projects** (within RESOURCES Section) and then click on  button. Create the new Examples Project using next information:

**NAME:** Examples Project

**DESCRIPTION:** A Project with many examples

**ORGANIZATION:** Workshop

**SCM TYPE:** Git














**SOURCE DETAILS (SCM URL):** <https://github.com/rcalvaga/ansible-examples.git>

**SCM UPDATE OPTIONS:**

- ☐ Clean ?
- ☐ Delete on Update ?
- ☒ Update Revision on Launch ?

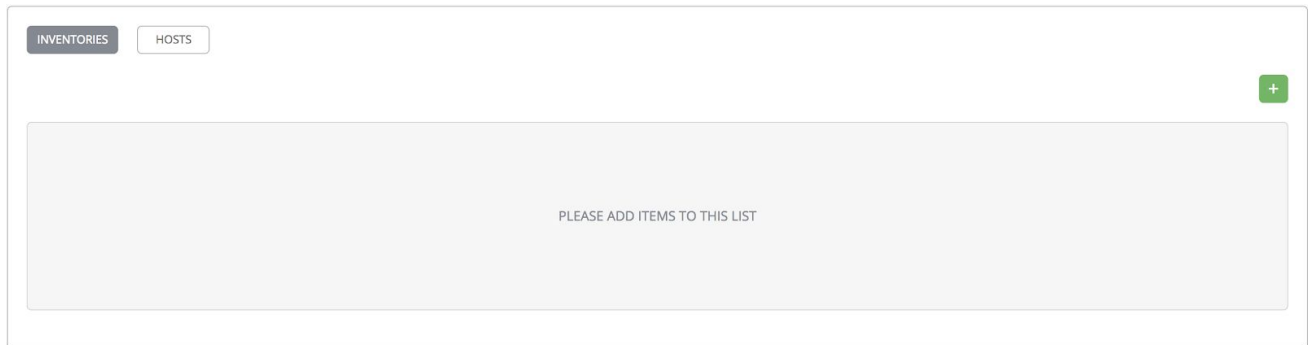
Click on **SAVE** to create the Examples Project.


Click on **Projects** (within RESOURCES Section). You will see your two new projects:

PROJECTS 2					
SEARCH				KEY	
NAME ^	TYPE ^	REVISION ^	LAST UPDATED ^	ACTIONS	
 App Project	Git	4631505 	1/9/2019 11:18:32 PM	   	
 Examples Project	Git	fca475b 	1/9/2019 11:19:17 PM	   	

ITEMS 1 - 2

## 2C. Create a new Inventory:



Click on **Inventories** (within RESOURCES Section) and then click on  button and select “Inventory” (not “Smart Inventory”). Create the new Inventory using next information:

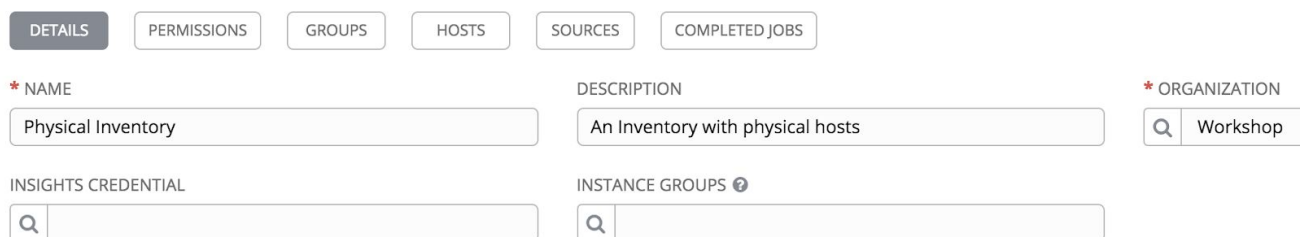
**NAME:** Physical Inventory

**DESCRIPTION:** An Inventory with physical hosts

**ORGANIZATION:** Workshop

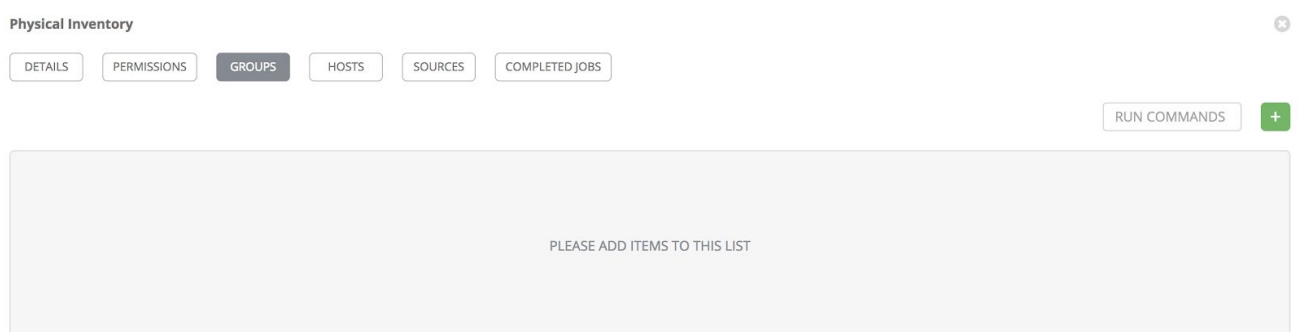
Click on **SAVE** to create the new Inventory.

### Physical Inventory



## 2D. Create 3 new Groups within Physical Inventory:

Within our recently created **Physical Inventory**, click on **GROUPS**:



Now we need to **add three groups** as follows:

Click on  button to create new group **lamp-servers**:

**NAME:** lamp-servers

**DESCRIPTION:** A Group of Webservers

Click on **SAVE** to create the new group.

lamp-servers

DETAILS

GROUPS

HOSTS

\* NAME

lamp-servers

DESCRIPTION

A Group of Webservers

VARIABLES

YAML

JSON

1

---

CANCEL

SAVE

Physical Inventory

DETAILS

PERMISSIONS

GROUPS

HOSTS

SOURCES

COMPLETED JOBS

SEARCH

Q

KEY

RUN COMMANDS

+

GROUPS

lamp-servers

ACTIONS

ITEMS 1 - 1

Click on  button again to create new group **wordpress-servers** (Physical Inventory will appear below the page):

**NAME:** wordpress-servers  
**DESCRIPTION:** A Group of Wordpress Servers

Click on **SAVE** to create the new group.

wordpress-servers

DETAILS

GROUPS

HOSTS

\* NAME

wordpress-servers

DESCRIPTION

A Group of Wordpress Servers

VARIABLES

YAML

JSON

1

---

CANCEL

SAVE

Physical Inventory

DETAILS

PERMISSIONS

GROUPS

HOSTS

SOURCES

COMPLETED JOBS

SEARCH

Q

KEY

RUN COMMANDS

+

GROUPS

lamp-servers

wordpress-servers

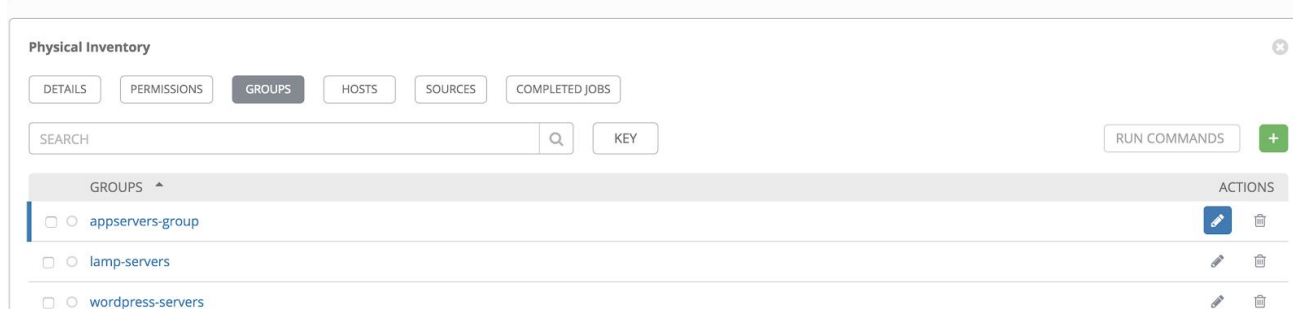
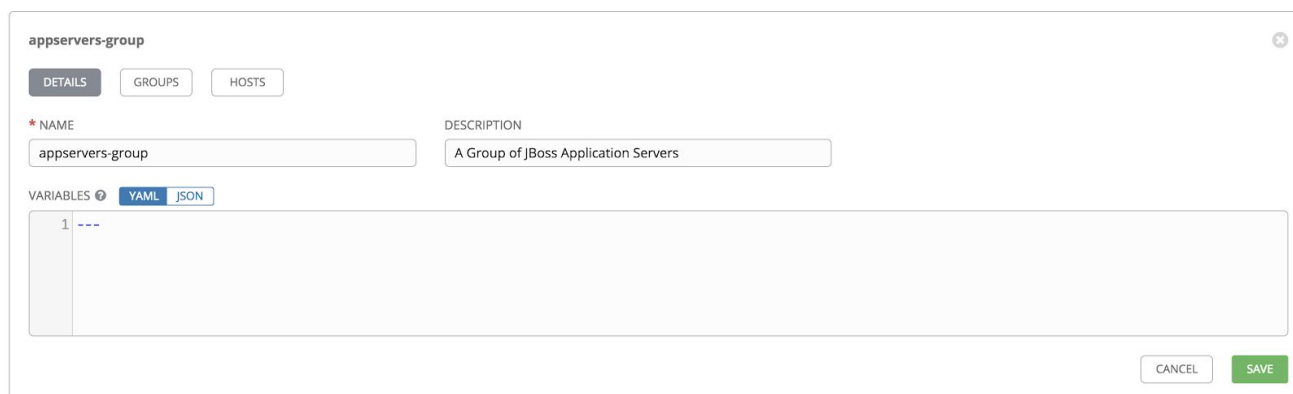
ACTIONS

ITEMS 1 - 2

Click on  button again to create new group **appservers-group** (Physical Inventory will appear below the page):

**NAME:** appservers-group  
**DESCRIPTION:** A Group of JBoss Application Servers

Click on **SAVE** to create the new group.




## 2E. Create 3 new Hosts, each one within the corresponding Group:

Click on **Inventories** → **Physical Inventory** → **GROUPS** → **lamp-servers** and then click on **HOSTS**. After that, click on  button (selecting *New Host*) and add one host as follows:


**HOST NAME:** webserver.demo.redhat.com  
**DESCRIPTION:** Web Server

Click on **SAVE** to create the new host.

Click on **Inventories** → **Physical Inventory** → **GROUPS** → **wordpress-servers** and then click on **HOSTS**. After that, click on  button (selecting *New Host*) and add one host as follows:

**HOST NAME:** wordpress.demo.redhat.com  
**DESCRIPTION:** Wordpress Server


Click on **SAVE** to create the new host.

Click on **Inventories** → **Physical Inventory** → **GROUPS** → **appservers-group** and then click on **HOSTS**. After that, click on  button (selecting *New Host*) and add one host as follows:

**HOST NAME:** jboss.demo.redhat.com  
**DESCRIPTION:** JBoss Application Server

Click on **SAVE** to create the new host.


## 2F. Create 3 new Job Templates:

Click on **Templates** (within RESOURCES Section) and then click on  button (selecting *Job Template*) to create a new Job Template as follows:

**NAME:** Install Web Server  
**DESCRIPTION:** Install Web Server  
**JOB TYPE:** Run  
**INVENTORY:** Physical Inventory  
**PROJECT:** Examples Project  
**PLAYBOOK:** lamp\_simple\_rhel7/site.yml  
**MACHINE CREDENTIAL:** SSH Server Credentials  
**LIMIT:** lamp-servers  
**VERBOSITY:** 2 (More Verbose)

Click on **SAVE** to create the new Job Template

### NEW JOB TEMPLATE

Click again on **Templates** and then click on  button (selecting *Job Template*) to create another Job Template as follows:

**NAME:** Install Wordpress Server  
**DESCRIPTION:** Install Wordpress Server  
**JOB TYPE:** Run  
**INVENTORY:** Physical Inventory  
**PROJECT:** Examples Project  
**PLAYBOOK:** wordpress\_nginx\_rhel7/site.yml  
**MACHINE CREDENTIAL:** SSH Server Credentials  
**LIMIT:** wordpress-servers  
**VERBOSITY:** 2 (More Verbose)

Click on **SAVE** to create the new Job Template

## NEW JOB TEMPLATE

DETAILS
PERMISSIONS
COMPLETED JOBS
SCHEDULES
ADD SURVEY

NAME
Install Wordpress Server

DESCRIPTION
Install Wordpress Server

JOB TYPE
Run

INVENTORY
Physical Inventory

PROJECT
Examples Project

PLAYBOOK
wordpress-nginx\_rhel7/site.yml

CREDENTIAL
SSH Server Credentials

FORKS
DEFAULT

LIMIT
wordpress-servers

VERBOSITY
2 (More Verbose)


JOB TAGS

SKIP TAGS

LABELS

ANSIBLE ENVIRONMENT
Use Default Environment

INSTANCE GROUPS

Click again on **Templates** and then click on  button (selecting *Job Template*) to create another Job Template as follows:

**NAME:** Install App Server  
**DESCRIPTION:** Install JBoss Application Server  
**JOB TYPE:** Run  
**INVENTORY:** Physical Inventory  
**PROJECT:** App Project  
**PLAYBOOK:** jboss-standalone/site.yml  
**MACHINE CREDENTIAL:** SSH Server Credentials  
**LIMIT:** appservers-group  
**VERBOSITY:** 2 (More Verbose)

Click on **SAVE** to create the new Job Template

## NEW JOB TEMPLATE

DETAILS
PERMISSIONS
COMPLETED JOBS
SCHEDULES
ADD SURVEY

NAME
Install App Server

DESCRIPTION
Install JBoss Application Server

JOB TYPE
Run

INVENTORY
Physical Inventory

PROJECT
App Project

PLAYBOOK
jboss-standalone/site.yml

CREDENTIAL
SSH Server Credentials

FORKS
DEFAULT

LIMIT
appservers-group

VERBOSITY
2 (More Verbose)

JOB TAGS

SKIP TAGS

LABELS

ANSIBLE ENVIRONMENT
Use Default Environment

INSTANCE GROUPS


Click again on **Templates**. You will see three Job Templates as follows:

TEMPLATES 3	
SEARCH	KEY
<div> <div>Install App Server</div> <div>Job Template</div> <div> INVENTORYPhysical Inventory PROJECTApp Project CREDENTIALSSH Server Credentials LAST MODIFIED1/10/2019 7:47:30 PM by wuser </div> </div>	
<div> <div>Install Web Server</div> <div>Job Template</div> <div> INVENTORYPhysical Inventory PROJECTExamples Project CREDENTIALSSH Server Credentials LAST MODIFIED1/10/2019 7:40:19 PM by wuser </div> </div>	
<div> <div>Install Wordpress Server</div> <div>Job Template</div> <div> INVENTORYPhysical Inventory PROJECTExamples Project CREDENTIALSSH Server Credentials LAST MODIFIED1/10/2019 7:43:16 PM by wuser </div> </div>	

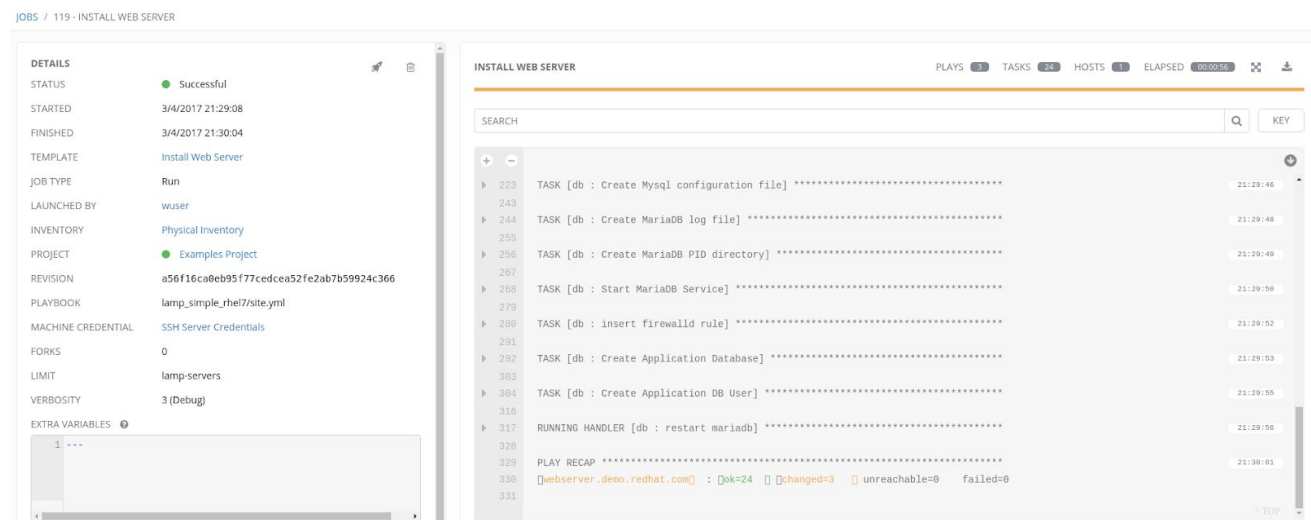
## Lab 3: Automating IT Process using Ansible Tower Jobs

### 3A. Create new Apache Webserver via Job Template:

First, click on your **Web Server URL** to be sure it's not installed.

Then, access to Ansible Tower. Click on **Templates** and find the “**Install Web Server**” Job Template. Click  on to start a job creation using this template.

This Action will launch the **Install Web Server** Job Template, starting the Webserver installation. You will be redirected to a dynamic Job Output page:



The screenshot shows the Ansible Tower interface for a job named "INSTALL WEB SERVER". On the left, the "DETAILS" panel shows the job status as "Successful", started at 3/4/2017 21:29:08, and finished at 3/4/2017 21:30:04. The job type is "Run", launched by "wuser", and uses the "Physical Inventory" inventory. The project is "Examples Project", revision "a56f16ca0eb95f77cedcea52fe2ab7b59924c366", and the playbook is "lamp\_simple\_rhel7/site.yml". The machine credential is "SSH Server Credentials", forks are 0, limit is "lamp-servers", and verbosity is 3 (Debug). The "EXTRA VARIABLES" section is empty. The main panel shows the job output, which includes a list of tasks: "TASK [db : Create Mysql configuration file]", "TASK [db : Create MariaDB log file]", "TASK [db : Create MariaDB PID directory]", "TASK [db : Start MariaDB Service]", "TASK [db : insert firewall rule]", "TASK [db : Create Application Database]", "TASK [db : Create Application DB User]", and "RUNNING HANDLER [db : restart mariadb]". The output also shows a "PLAY RECAP" summary: "[]webserver.demo.redhat.com[] : []ok=24 []changed=3 []unreachable=0 failed=0".

You will see the Status of the Job, Started / Finished Time, Tasks, etc.

If your **Job** finished **Ok**, with Status **Successful**, go to your **Web Server URL** and make sure your Web Server is Up & Running.

← → ↺ ⓘ No seguro | webserver-latamsmeansibletow-zczats8h.srv.ravcloud.com

# Hello World!

My Web App was deployed via **Red Hat Ansible Tower**.

We will be taking a look at the Ansible Playbook we are using for this Job Template, following this link:

[https://github.com/rcalvaga/ansible-examples/blob/master/lamp\\_simple\\_rhel7/site.yml](https://github.com/rcalvaga/ansible-examples/blob/master/lamp_simple_rhel7/site.yml)



### 3B. Create new Wordpress Website via Job Template:

First, click on your **Wordpress Server URL** to be sure it's not installed.

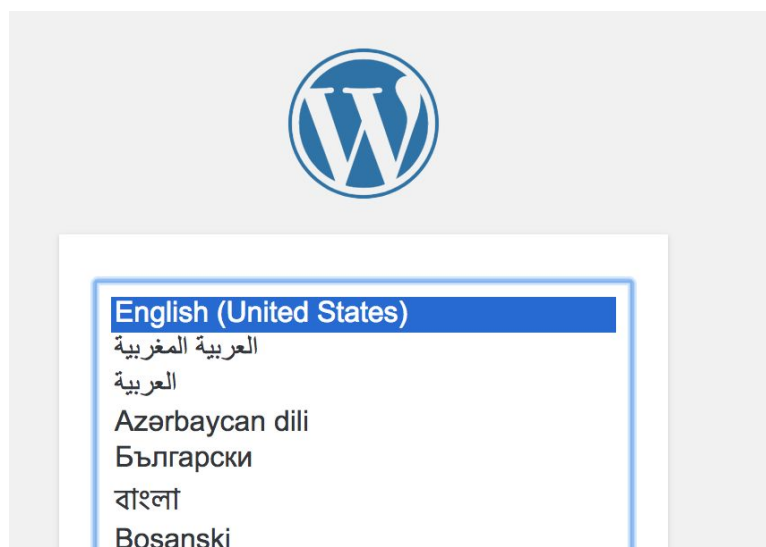


Then, access to Ansible Tower. Click on **Templates** and find the “**Install Wordpress Server**” Job Template. Click on **Run** to start a job creation using this template.

This Action will launch the **Install Wordpress Server** Job Template, starting the Website installation. You will be redirected to a dynamic Job Output page:

You will see the Status of the Job, Started / Finished Time, Tasks, etc.

If your **Job** finished **Ok**, with Status **Successful**, go to your **Wordpress Server URL** and make sure your Wordpress Server is Up & Running. **Play a moment with your new Wordpress!**




We will be taking a look at the Ansible Playbook we are using for this Job Template, following this link:

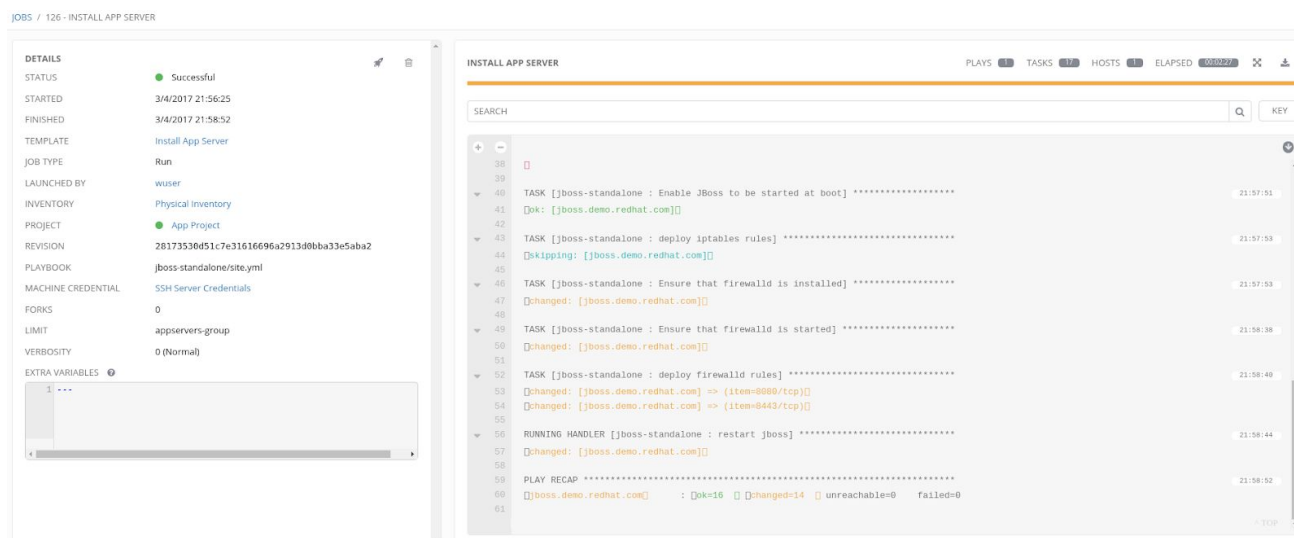
[https://github.com/rcalvaga/ansible-examples/blob/master/wordpress-nginx\\_rhel7/site.yml](https://github.com/rcalvaga/ansible-examples/blob/master/wordpress-nginx_rhel7/site.yml)

### 3C Create new JBoss Application Server via Job Template:

First, click on your **JBoss Application Server URL** to be sure it's not installed.

Then, access to Ansible Tower. Click on **Templates** and find the “**Install App Server**” Job Template. Click  on to start a job creation using this template.

This Action will launch the **Install App Server** Job Template, starting the JBoss App Server installation. You will be redirected to a dynamic Job Output page:



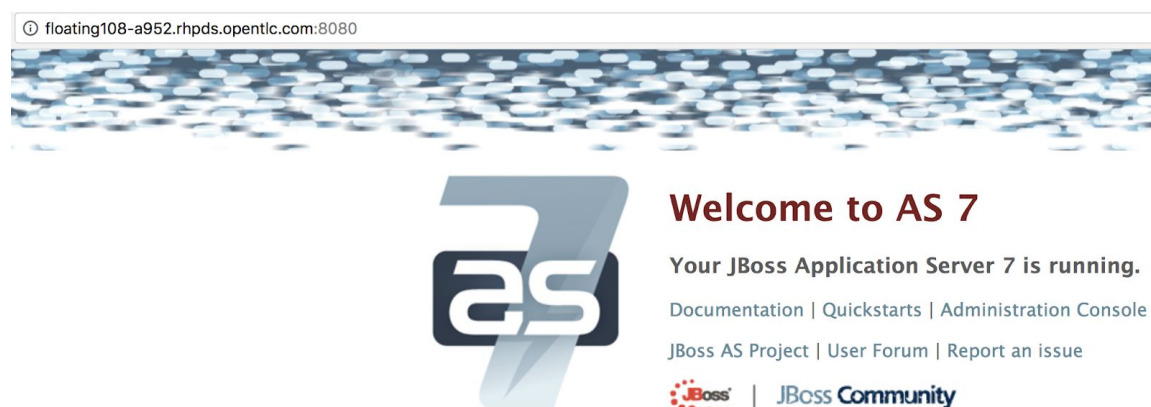
The screenshot shows the Ansible Tower interface for a job named 'INSTALL APP SERVER'. On the left, the 'DETAILS' sidebar shows the job status as 'Successful', started on 3/4/2017 at 21:56:25, and finished at 21:58:52. The job type is 'Run', launched by 'wuser', and uses the 'Physical Inventory'. The project is 'App Project' and the playbook is 'jboss-standalone/site.yml'. The right pane shows the job output with a search bar and a list of tasks. The tasks include enabling JBoss to start at boot, deploying iptables rules, ensuring the firewall is installed and started, and deploying firewall rules. The output shows that the firewall rules were successfully installed and started. The job recap at the bottom shows that the job was successful with no errors.

You will see the Status of the Job, Started / Finished Time, Tasks, etc.

If your **Job** finished **Ok**, with Status **Successful**, go to your **JBoss Application Server URL** and make sure your App Server is Up & Running, **accessing via :8080 Port!**

Click on **Administration Console** link. You can access to the **JBoss Administration Console**:

**User:** admin / **Password:** Redhat1!

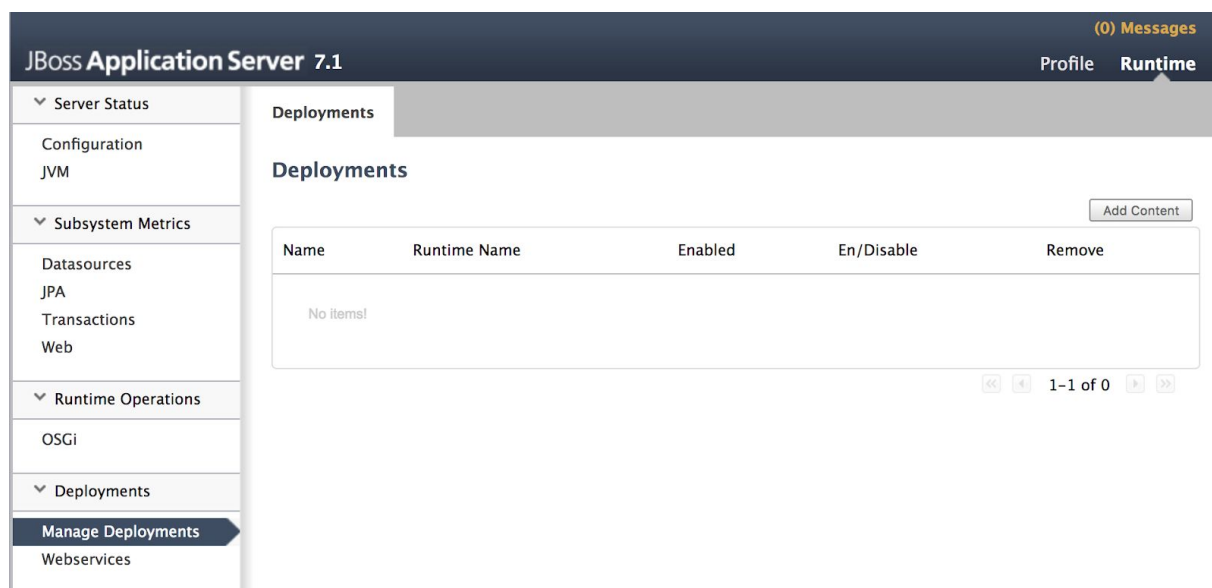


The screenshot shows the JBoss AS 7 Welcome screen. The URL bar at the top shows 'floating108-a952.rhps.opentlc.com:8080'. The main content area features the JBoss AS 7 logo on the left and a welcome message on the right. The message says 'Welcome to AS 7' and 'Your JBoss Application Server 7 is running.' Below this, there are links for 'Documentation | Quickstarts | Administration Console', 'JBoss AS Project | User Forum | Report an issue', and the 'JBoss Community' logo.

We will be taking a look at the Ansible Playbook we are using for this Job Template, following this link:


<https://github.com/rcalvaga/examples-ansible/blob/master/jboss-standalone/site.yml>

Now click on **Deployments** → **Manage Deployments**. You will see that we have no deployments yet. We will be creating a workflow template to deploy two new Java Applications on JBoss in Lab 4.



## Lab 4: Automating IT Process Using Ansible Tower Workflows

First as **wuser**, we need to create another Ansible Tower Job to deploy a Java Application within our JBoss App Server:














Click on **Templates** and then click on  button (selecting *Job Template*) → **Job Template** to create a new Job Template as follows:

**NAME:** Install Java App  
**DESCRIPTION:** Install a Java App within JBoss  
**JOB TYPE:** Run  
**INVENTORY:** Physical Inventory  
**PROJECT:** App Project  
**PLAYBOOK:** jboss-standalone/deploy-application.yml  
**MACHINE CREDENTIAL:** SSH Server Credentials  
**LIMIT:** appservers-group  
**VERBOSITY:** 2 (More Verbose)


Click on **SAVE** to create the new Job Template

NEW JOB TEMPLATE

DETAILS PERMISSIONS COMPLETED JOBS SCHEDULES ADD SURVEY

* NAME	DESCRIPTION	* JOB TYPE 	<input type="checkbox"/> PROMPT ON LAUNCH
<input type="text" value="Install Java App"/>	<input type="text" value="Install a Java App within JBoss"/>	<input type="text" value="Run"/>	
* INVENTORY 	* PROJECT 	* PLAYBOOK 	
<input type="text" value="Physical Inventory"/>	<input type="text" value="App Project"/>	<input type="text" value="jboss-standalone/deploy-application.yml"/>	
CREDENTIAL 	FORKS 	LIMIT 	<input type="checkbox"/> PROMPT ON LAUNCH
<input type="text" value="SSH Server Credentials"/>	<input type="text" value="DEFAULT"/>	<input type="text" value="appservers-group"/>	
* VERBOSITY 	JOB TAGS 	SKIP TAGS 	<input type="checkbox"/> PROMPT ON LAUNCH
<input type="text" value="1 (Verbose)"/>	<input type="text"/>	<input type="text"/>	
LABELS 	ANSIBLE ENVIRONMENT 	INSTANCE GROUPS 	
<input type="text"/>	<input type="text" value="Use Default Environment"/>	<input type="text"/>	

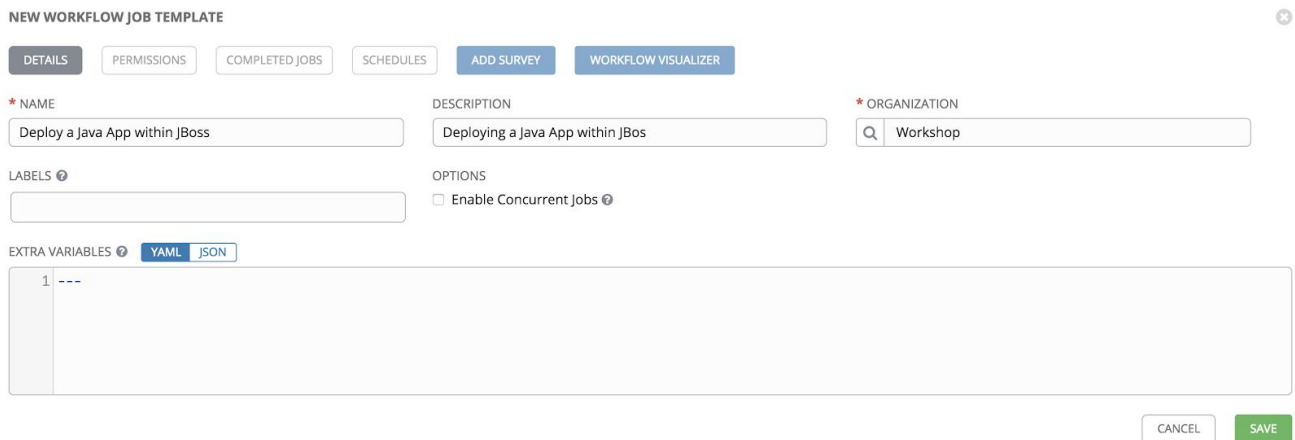
Then, we need to create a new **Workflow Job Template**, so we can later create a workflow of Job Templates:

Click on **Templates** and then click on  button (selecting *Workflow Template*)→ **Workflow Template** to create a new Workflow Template as follows:

**NAME:** Deploy a Java App within JBoss

**DESCRIPTION:** Deploying a Java App within JBoss

**ORGANIZATION:** Workshop



Click on **SAVE**. Then, create a workflow within our new Workflow Job Template, so click on **WORKFLOW VISUALIZER**:

### Deploy a Java App within JBoss



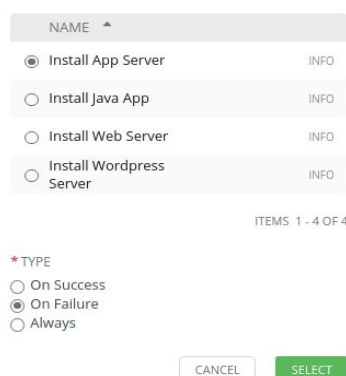
Next, click on **START** button: 

We can now select our first initial Job Template. Select **Install Java App** Job Template and click on **SELECT**.

Now click on **Install Java App** Job box:

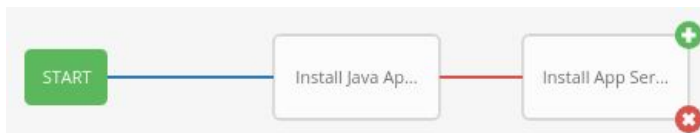


Click on **Green Plus** to add a sequential Job Template. Select **Install App Server** Job Template and select **On Failure** as **TYPE**. This Job will run if our first Job *Install Java App* has a **failed status**, so *Install App Server* Job will ensure to have all the necessary JBoss App Server infrastructure.



Click on **SELECT** to select the Job Template.

Now click on on **Install App Server** Job box:



Click on **Green Plus** to add a sequential Job Template. Select **Install Java App** Job Template and select **On Success** as **TYPE**. This Job will run if our *Install App Server Job* has a **successful status**, so *Install Java App* Job will install the Java Application on our JBoss App Server infrastructure.

NAME ^	
<input type="radio"/> Install App Server	INFO
<input checked="" type="radio"/> Install Java App	INFO
<input type="radio"/> Install Web Server	INFO
<input type="radio"/> Install Wordpress Server	INFO

ITEMS 1 - 4 OF 4

\* TYPE

☒ On Success

☐ On Failure

☐ Always

Click on **SELECT** to select the Job Template.

You will have something like this:



Then click on **SAVE** to create the new Workflow. You will be coming back to the main **workflow template** screen:

Deploy a Java App within JBoss

DETAILS PERMISSIONS COMPLETED JOBS SCHEDULES ADD SURVEY WORKFLOW VISUALIZER

\* NAME Deploy a Java App within JBoss DESCRIPTION Deploying a java App within JBos \* ORGANIZATION Workshop

LABELS

OPTIONS

☐ Enable Concurrent Jobs

EXTRA VARIABLES

YAML JSON

1

CANCEL SAVE

Now, click on **SAVE** to save **Deploy a Java App within JBoss** again.

**NOTE:** Make sure your **JBoss Java Application URL** is not available (**HTTP Status 404 - :8080/ticket-monster**) before starting you workflow! → Access your URL using **:8080 Port!**

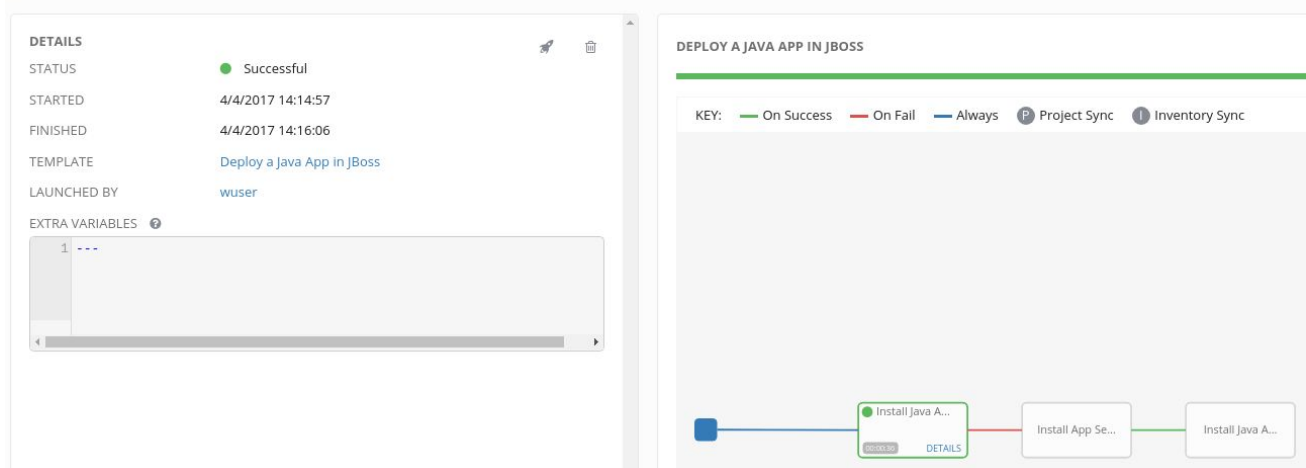
Now Click on **Templates** and find the “**Deploy a Java App in JBoss**” Workflow Template. Click  on to start a Workflow Job creation using this workflow template.

### Deploy a Java App within JBoss Workflow Template

LAST MODIFIED 1/11/2019 7:12:59 PM by [wuser](#)

This Action will launch the **Install Java App** Job Template, starting the Java App Server installation. You will be redirected to a dynamic Workflow Job Output page:

[JOBS](#) / 183 - DEPLOY A JAVA APP IN JBOSS

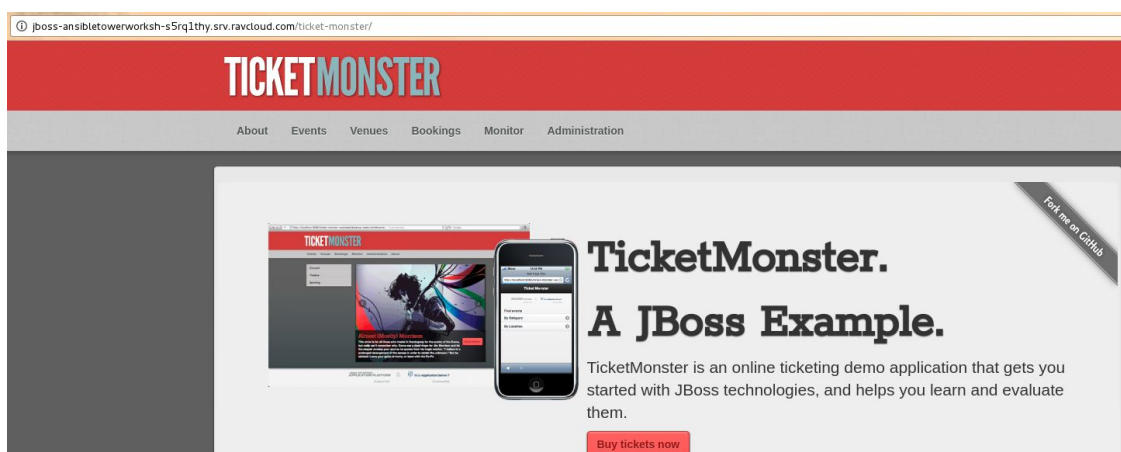


The screenshot shows the workflow job output page for 'DEPLOY A JAVA APP IN JBOSS'. On the left, the 'DETAILS' panel shows the job status as 'Successful', started on 4/4/2017 at 14:14:57, and finished at 14:16:06. The template used is 'Deploy a Java App in JBoss' and it was launched by 'wuser'. The 'EXTRA VARIABLES' section shows a single variable '1' with a value of '---'. On the right, the workflow diagram shows three steps: 'Install Java A...' (green box), 'Install App Se...' (white box), and 'Install Java A...' (white box). The first step is marked as successful with a green dot and a '00:00:30' duration. The workflow is connected by a blue line, indicating it ran 'Always'.

If our first Job Template has a **Successful status**, Workflow finishes, but if it has a **Failure status**, workflow will continue until finishes.

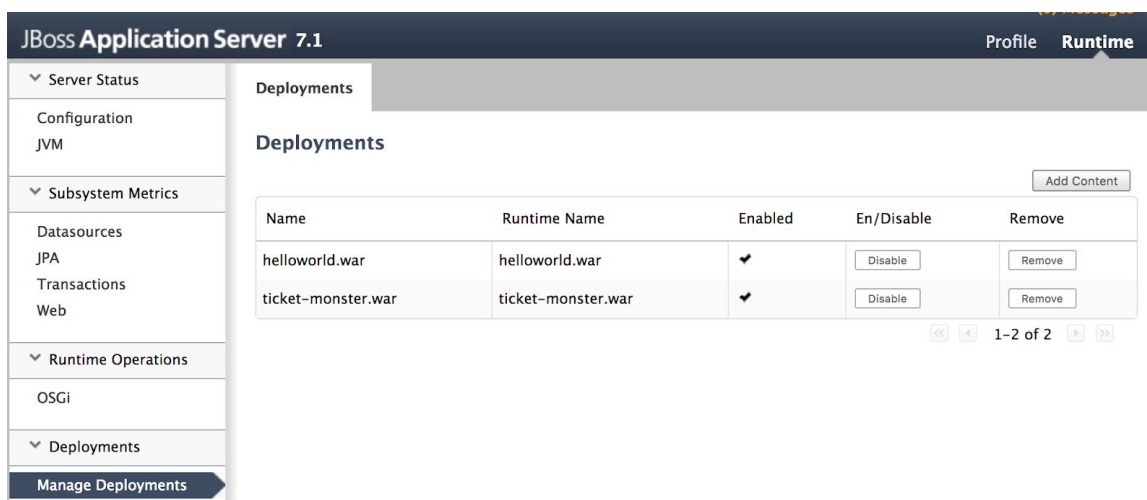
If everything finishes **Ok**, you can access to your **JBoss Java Application URL** (with **:8080/ticket-monster** context root and then try with **:8080/helloworld**)

You can access to **TicketMonster Web Application Server** to buy some tickets:



**Just play a moment with your new Java Application!**

Now come back to your **Administration Console** link. Then click on **Deployments** → **Manage Deployments**. Refresh page if necessary. You will see your new deployments:



## Lab 5: Running Ad-Hoc Commands Using Ansible Tower

### Scenario

An **Ad-Hoc command** is something that you might type in to do something really quick, but don't want to save for later. We will use **Ansible Tower** to run some **Ad-Hoc commands**, such as **date**, **last**, **whoami** and so on, into some different servers at the same time.

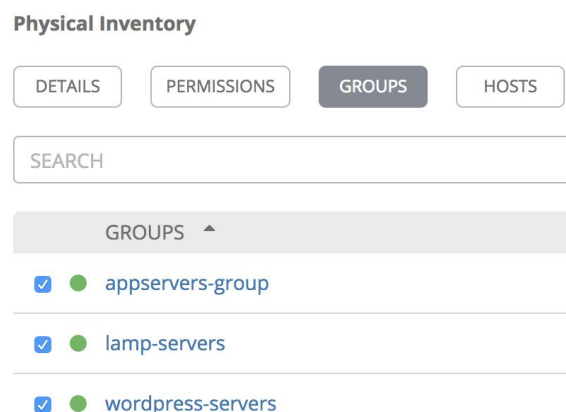
### Tasks

Using our servers in **Physical Inventory**, run some commands using different **Ansible Modules**.

Access to **Ansible Tower Portal**, using our new **Workshop User**:

**User: wuser / Password: Redhat1!**

Click on **Inventories** and then click on **Physical Inventory** → **GROUPS**. Then select (check) our 3 different Groups: **appservers-group**, **lamp-servers** and **wordpress-servers**:



After that, click on **RUN COMMANDS** button:





### 5A. Using Command Module:

Within **EXECUTE COMMAND** section, run **date** command as follows:

**MODULE:** command

### MACHINE CREDENTIAL: SSH Server Credentials

**ARGUMENTS:** date

**LIMIT:** appservers-group:lamp-servers:wordpress-servers ←(keep it as is)

Click on **LAUNCH** to run the command. Observe the results.

You will observe within **STANDARD OUT** section something like this:

```
webserver.demo.redhat.com | CHANGED | rc=0 >>
Sat Jan 12 01:35:27 UTC 2019

jboss.demo.redhat.com | CHANGED | rc=0 >>
Sat Jan 12 01:35:27 UTC 2019

wordpress.demo.redhat.com | CHANGED | rc=0 >>
Sat Jan 12 01:35:27 UTC 2019
```

**NOTE: You can click on Download Output  to download the results.**

Click on **Inventories** and then click on **Physical Inventory** → **GROUPS**. Select the same Groups and click on **RUN COMMANDS** again.

Within **EXECUTE COMMAND** section, run **last** command as follows:

**MODULE:** command

### MACHINE CREDENTIAL: SSH Server Credentials

**ARGUMENTS:** last

**LIMIT:** appservers-group:lamp-servers:wordpress-servers ←(keep it as is)

Click on **LAUNCH** to run the command. Observe the results.

You will observe within **STANDARD OUT** section something like this:

[illegible]



## 5B. Using Yum Module:

Click on **Inventories** and then click on **Physical Inventory** → **GROUPS**. Select the same Groups and click on **RUN COMMANDS** again.

Within **EXECUTE COMMAND** section, using **yum module**, install **telnet** package as follows:

**MODULE:** yum

**MACHINE CREDENTIAL:** SSH Server Credentials

**ARGUMENTS:** name=telnet state=latest

**LIMIT:** appservers-group:lamp-servers:wordpress-servers ←(keep it as is)

Click on **LAUNCH** to run the command. Observe the results.

You will observe within **STANDARD OUT** section something like that:

```
SSH password:
jboss.demo.redhat.com | SUCCESS => {
  "changed": true,
  "msg": "",
  "rc": 0,
  "results": [
    "Loaded plugins: fastestmirror\nLoading mirror speeds from cached hostfile\n * base: mirror.cogentco.com\n * extras: mirror.netdepot.com\n * updates: mirror.vcu.edu\n\nResolving Dependencies\n--> Running transaction check\n--> Package telnet.x86_64 1:0.17-60.el7 will be installed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n===== Package Arch\nVersion Repository Size\nx86_64 1:0.17-60.el7 base 63 k\n\nTransaction Summary\n\nInstall 1 Package\n\nTotal download size: 63 k\nInstalled size: 113 k\n\nDownloading packages:\nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\nInstalling : 1:telnet-0.17-60.el7.x86_64\n1/1 \n\nVerifying : 1:telnet-0.17-60.el7.x86_64\n1/1 \n\nInstalled:\n  telnet.x86_64 1:0.17-60.el7\n\nComplete!\n"
```

## 5C. Using User Module:

Click on **Inventories** and then click on **Physical Inventory** → **GROUPS**. Select the same Groups and click on **RUN COMMANDS** again.

Within **EXECUTE COMMAND** section, using **User module**, create **demo** user as follows:

**MODULE:** user

**MACHINE CREDENTIAL:** SSH Server Credentials

**ARGUMENTS:** name=demo comment="Demo User" password=\$1\$ctRQ8kmb\$PMF.2YAjQrdjiDGFuE4uw0 ←(needs to be crypted)

**LIMIT:** appservers-group:lamp-servers:wordpress-servers ←(keep it as is)

Click on **LAUNCH** to run the command. Observe the results.

You will observe within **STANDARD OUT** section something like that:

```
SSH password:
webserver.demo.redhat.com | SUCCESS => {
  "changed": true,
  "comment": "Demo User",
  "createhome": true,
  "group": 1001,
  "home": "/home/demo",
  "name": "demo",
  "password": "NOT_LOGGING_PASSWORD",
  "shell": "/bin/bash",
  "state": "present",
  "system": false,
  "uid": 1001
}
```

## 5D. Using Ping Module:

Click on **Inventories** and then click on **Physical Inventory** → **GROUPS**. Select the same Groups and click on **RUN COMMANDS** again.

Within **EXECUTE COMMAND** section, using **Ping module**, **ping** hosts in Groups as follows:

**MODULE:** ping

**MACHINE CREDENTIAL:** SSH Server Credentials

**ARGUMENTS:**

**LIMIT:** appservers-group:lamp-servers:wordpress-servers ←(keep it as is)

Click on **LAUNCH** to run the command. Observe the results.

You will observe within **STANDARD OUT** section something like that:

```
SSH password:
wordpress.demo.redhat.com | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
webserver.demo.redhat.com | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
jboss.demo.redhat.com | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
```

## Lab 6: Automating Windows


Ansible knew that the key was to bring the same simple, agentless paradigm to managing Windows, while still feeling native to Windows administrators. Ansible's native **Windows support** uses Windows PowerShell remoting to manage Windows like Windows in the same Ansible agentless way that Ansible manages Linux like Linux.

Plus, with Ansible's easy extensibility, you can write your own modules in **PowerShell** and extend Ansible for whatever other functionality you need. Ansible users have written modules for managing filesystem ACLs, managing Windows Firewall, and managing hostname and domain membership, and more.

In this lab we will be automating many tasks (such as creating users, copying files, configuring services, etc.) within a **Windows 2012 Server**.

Access to Ansible Tower as **wuser** and follow next steps:

### 6A. Create Windows Credentials:

Navigate within the **Ansible Tower Portal** and click on **Credentials** (within RESOURCES Section). Click on  to add new Windows Credentials as follows:

**Name:** Windows Credentials

**Description:** Windows Credentials

**Organization:** Workshop

**Type:** Machine

**Username:** Administrator

**Password:** Redhat1!

Click on **SAVE** to save the changes.

## 6B. Create Windows Project:

Click on **Projects** (within RESOURCES Section). Click on  button. Create the new **Windows Project** using next information:

**NAME:** Windows Project

**DESCRIPTION:** Windows Project

**ORGANIZATION:** Workshop

**SCM TYPE:** Git

**SOURCE DETAILS (SCM URL):** <https://github.com/rcalvaga/windows-ansible.git>

**SCM UPDATE OPTIONS:**

SCM UPDATE OPTIONS


☐ Clean ?

☐ Delete on Update ?

☒ Update Revision on Launch ?

Click on **SAVE** to create the new Project.

## 6C. Create windows-servers Group:

Click on **Inventories** → **Physical Inventory** → **GROUPS** and then click on  to add **windows-servers** group as follows:

**NAME:** windows-servers

**DESCRIPTION:** Windows Servers Group

**VARIABLES (YAML):**

---

ansible\_connection: winrm

ansible\_ssh\_port: 5986

ansible\_winrm\_server\_cert\_validation: ignore

Click on **SAVE** to create the new group.

WINDOWS-SERVERS

DETAILS

NOTIFICATIONS

\* NAME

windows-servers

DESCRIPTION

windows-servers

SOURCE

Manual

VARIABLES ? ☒ YAML ☐ JSON

```
1 ---
2 ansible_connection: winrm
3 ansible_ssh_port: 5986
4 ansible_winrm_server_cert_validation: ignore
```

## 6D. Add a new host to windows-servers Group:

Click on **Inventories** → **Physical Inventory** → **GROUPS** → **windows-servers** and then click on **HOSTS**. Now click on



(select *New Host*) to add one host as follows:

**HOST NAME:** windows.demo.redhat.com

**DESCRIPTION:** Windows 2012

Click on **SAVE** to create the new host.

## 6E. Create a Windows Job Template:

Click on **Templates** and then click on  button (selecting *Job Template*) to create a new Job Template as follows:

**NAME:** Automating Windows  
**DESCRIPTION:** Automating Windows  
**JOB TYPE:** run  
**INVENTORY:** Physical Inventory  
**PROJECT:** Windows Project  
**PLAYBOOK:** tower-ansible-automating-windows.yml  
**MACHINE CREDENTIAL:** Windows Credentials  
**LIMIT:** windows-servers  
**VERBOSITY:** 0 (Normal)

Click on **SAVE** to create the new Job Template

## 6F. First, verify that our Windows 2012 has not been yet automated:

Access to **Windows 2012** via RDP:

**Username:** Administrator / **Domain:** local / **Password:** Redhat1!

**TASK:** Verify that **Telnet Client** and **IIS Web-Server** has **not** been yet installed:

- Click on **Server Manager** → **Add roles and features**.
- Then click on **Server Selection** and then click on **Server Roles**. Verify that **Web Server IIS** has **not** been yet installed.
- Then click on **Features** and verify that **Telnet Client** has **not** been yet installed


**TASK:** Verify that **Ansible User** has **not** been yet created:

- Click on **Control Panel** → **User Accounts** → **User Accounts** → **Manage Another Account**. Verify that **Ansible User** has **not** been yet created

**TASK:** Verify that **nerd.jpg** image does **not** exist:

- Click on **File Explorer** and then click on **Downloads**. Verify that **nerd.jpg** image does **not** exist.

## 6G. Run the Windows Job Template:

Click on **Templates** and find the “**Automating Windows**” Job Template. Click  on to start a job creation using this template.

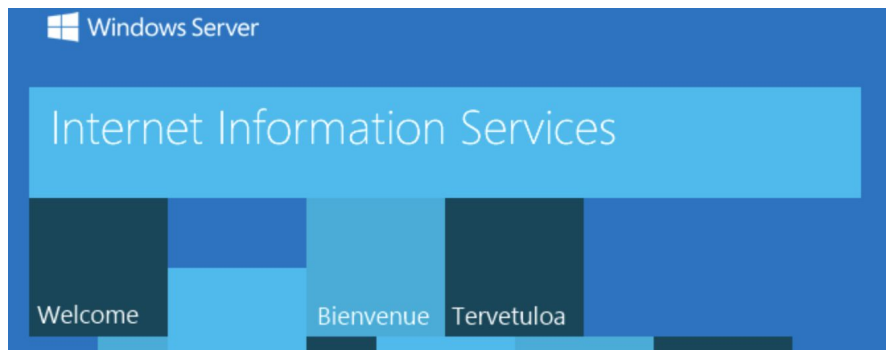
<b>Automating Windows</b> Job Template	
INVENTORY	Physical Inventory
PROJECT	Windows Project
LAST MODIFIED	1/11/2019 8:09:13 PM by wuser

This Action **will automate** some tasks on **Windows 2012**, such as:

- Add a Windows Admin User (Ansible User)
- Install Software (Telnet and IIS Server)
- Copy a JPG archive (neerdd.jpg as nerd.jpg)
- Run a PowerShell script (Hello World)
- Run some Windows commands (ipconfig)
- Check Status of file win.in

If your **Job Template** ran **successfully**, verify again steps in **6F** section, but now you will find that **Windows 2012** has been automated!

Access to **Windows 2012** URL but now via **Web Browser**, just to make sure **Web Server IIS** is **Up** and **Running**!




Now just take a look at the Ansible Playbook:

<https://github.com/rcalvaga/windows-ansible/blob/master/tower-ansible-automating-windows.yml>

## Lab 7: Automating Network Devices

### 7a. Create VyOS Credentials:

Navigate within the **Ansible Tower Portal** and click on **Credentials** (within RESOURCES Section). Click on  to add new Network Credentials as follows:

**NAME:** VyOS Credentials

**DESCRIPTION:** VyOS Credentials

**ORGANIZATION:** Workshop


**TYPE:** Network

**USERNAME:** vyos

**PASSWORD:** Redhat1!

Click on **SAVE** to save the changes.

### 7B. Create VyOS Project:

Click on **Projects** (within RESOURCES Section). Click on  button. Create the new **VyOS Project** using next information:

**NAME:** VyOS Project

**DESCRIPTION:** a VyOS Project

**ORGANIZATION:** Workshop

**SCM TYPE:** Git

**SCM\_URL:** <https://github.com/rcalvaga/vyos-ansible>

**SCM UPDATE OPTIONS:**

SCM UPDATE OPTIONS


☐ Clean ?

☐ Delete on Update ?

☒ Update Revision on Launch ?

Click on **SAVE** to create the new Project.

### 7C. Create vyos-servers group:


Click on **Inventories** → **Physical Inventory** → **GROUPS**. Now click on  to add **vyos-servers** group as follows:

**NAME:** vyos-servers

**DESCRIPTION:** VyOS Servers Group

Click on **SAVE** to create the new group.

### 7D. Add a new host to vyos-servers group:

Click on **Inventories** → **Physical Inventory** → **GROUPS** → **vyos-servers** and then click on **HOSTS**. Now click on  button (selecting *New Host*) to add one host as follows:

**HOST NAME:** vyos.demo.redhat.com

**DESCRIPTION:** VyOS Network Appliance

Click on **SAVE** to create the new host.

### 7E. Create a VyOs Job Template:

Click on **Templates** and then click on  button (selecting *Job Template*) to create a new Job Template as follows:

**NAME:** Automating VyOS

**DESCRIPTION:** Automating VyOS

**JOB TYPE:** run

**INVENTORY:** Physical Inventory

**PROJECT:** VyOS Project

**PLAYBOOK:** tower-ansible-automating-vyos.yml

**MACHINE CREDENTIAL:** SSH Server Credentials

**NETWORK CREDENTIAL:** VyOS Credentials

**LIMIT:** vyos-servers

**VERBOSITY:** 0 (Normal)


**TIP: Selecting Machine Credential:** just click on  icon within **CREDENTIAL** field:

CREDENTIAL ⓘ ☐ PROMPT ON LAUNCH




Select **Machine** as **CREDENTIAL TYPE** and now select **SSH Server Credentials**:

**CREDENTIALS** ⓘ

SELECTED:  SSH Server Credentials ✕

CREDENTIAL TYPE: Machine ▼

SEARCH  KEY

NAME ▲


☒ SSH Server Credentials


☐ Windows Credentials




ITEMS 1 - 2

CANCEL SELECT


Finally click on **SELECT** button.





**TIP: Selecting Network Credential:** just click again on  icon within **CREDENTIAL** field:


CREDENTIAL  ☐ PROMPT ON LAUNCH


  SSH Server Credentials 


Select **Network** as **CREDENTIAL TYPE** and now select **VyOS Credentials**:

**CREDENTIALS** 

SELECTED:  SSH Server Credentials   VyOS Credentials 

CREDENTIAL TYPE: Network 

SEARCH  KEY

NAME 

☒ VyOS Credentials

ITEMS 1 - 1

Finally click on **SELECT** button.



You will have your CREDENTIAL field as follows:

CREDENTIAL  ☐ PROMPT ON LAUNCH

  SSH Server Credentials   VyOS Credentials 

Click on **SAVE** to create the new Job Template

## 7F. Run the “Automating VyOS” Job Template:

Click on **Templates** and find the “**Automating VyOS**” Job Template. Click  on  to start a job creation using this template.

This Action **will automate** some tasks on **VyOS Network Virtual Appliance**, providing a **NAT Gateway** for this device with two interfaces, as described within the VyOS Quick Start Guide:

[https://wiki.vyos.net/wiki/User\\_Guide#Quick\\_Start\\_Guide](https://wiki.vyos.net/wiki/User_Guide#Quick_Start_Guide)

If your **Job Template** ran **successfully**, just click on some tasks to verify that **your VyOS has been automated!**

Now take a look at the Ansible Playbook:

<https://github.com/rcalvaga/vyos-ansible/blob/master/tower-ansible-automating-vyos.yml>

**If you finished the Workshop, please answer this Survey:**

Select **Ansible Tower Automation** as **Test Drive Name**:

[https://docs.google.com/forms/d/e/1FAIpQLSdauHtguNMYICRE5x1nrE0Y11ASfDNnptSEqLZi\\_TCsNgb2g/viewform](https://docs.google.com/forms/d/e/1FAIpQLSdauHtguNMYICRE5x1nrE0Y11ASfDNnptSEqLZi_TCsNgb2g/viewform)