Table of Contents

* [Ansible Usage Lab](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_ansible_usage_lab)
* [1. Use Ad Hoc Commands](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_use_ad_hoc_commands)
  + [1.1. Access Environment](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_access_environment)
  + [1.2. Create an Inventory File](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_create_an_inventory_file)
  + [1.3. Propagate SSH Key to Managed Hosts](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_propagate_ssh_key_to_managed_hosts)
  + [1.4. Gather Facts](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_gather_facts)
  + [1.5. Install a Package](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_install_a_package)
  + [1.6. Start a Service](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_start_a_service)
* [2. Use](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_use_code_ansible_playbook_code_command_line) ansible-playbook Command Line
  + [2.1. Create a Simple Playbook](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_create_a_simple_playbook)
  + [2.2. Add Jinja2 Template Files](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_add_jinja2_template_files)
  + [2.3. Run Playbook](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Foundations/11/04_Usage/04_Usage_Lab.html" \l "_run_playbook)

## Ansible Usage Lab

In this lab, you learn how to use Ansible ad hoc commands and the ansible-playbook command line.

## 1. Use Ad Hoc Commands

In this exercise, you use Ansible ad hoc commands to do the following:

* Structure a basic inventory of your managed hosts
* Test their reachability
* Install a web server
* Start the web server

### 1.1. Access Environment

1. Connect to the control node:

# ssh your-sso-login@workstation-GUID.rhpds.opentlc.com

1. Become the root user:

# sudo -i

### 1.2. Create an Inventory File

1. Edit the default inventory file, /etc/ansible/hosts:

# vi /etc/ansible/hosts

1. Add the following lines to assign servera to the web group and serverb to the sql group:

[web]

servera.example.com

[sql]

serverb.example.com

1. Use an ad hoc command to test the inventory file:

# ansible -m ping all

servera.example.com | UNREACHABLE! => {

"changed": false,

"msg": "Failed to connect to the host via ssh: Warning: Permanently added 'servera.example.com' (ECDSA) to the list of known hosts.\r\nPermission denied (publickey,gssapi-keyex,gssapi-with-mic).\r\n",

"unreachable": true

}

serverb.example.com | UNREACHABLE! => {

"changed": false,

"msg": "Failed to connect to the host via ssh: Warning: Permanently added 'serverb.example.com' (ECDSA) to the list of known hosts.\r\nPermission denied (publickey,gssapi-keyex,gssapi-with-mic).\r\n",

"unreachable": true

}

* + As you can see, the control node needs to be able to access the managed hosts using SSH, which is covered in the next section.

### 1.3. Propagate SSH Key to Managed Hosts

1. From the root user home folder (/root), copy the root SSH key to servera and serverb. The root password for the servers is redhat.

# ssh-copy-id -i .ssh/open servera.example.com

/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed

/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys

root@servera.example.com's password:

Number of key(s) added: 1

# This is the default ansible 'hosts' file.

Now try logging into the machine, with: "ssh 'servera.example.com'"

and check to make sure that only the key(s) you wanted were added.

# ssh-copy-id -i .ssh/open serverb.example.com

/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed

/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys

root@serverb.example.com's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'serverb.example.com'"

and check to make sure that only the key(s) you wanted were added.

1. Use an ad hoc command to test the inventory file again:

# ansible -m ping all

servera.example.com | SUCCESS => {

"changed": false,

"ping": "pong"

}

serverb.example.com | SUCCESS => {

"changed": false,

"ping": "pong"

}

* + Expect the test to succeed this time.

### 1.4. Gather Facts

1. Use an ad hoc command to get the list of facts available for servera:

# ansible web -m setup

servera.example.com | SUCCESS => {

"ansible\_facts": {

"ansible\_all\_ipv4\_addresses": [

"192.168.1.21"

],

"ansible\_all\_ipv6\_addresses": [

"fe80::2ec2:60ff:fe22:53a9"

],

"ansible\_architecture": "x86\_64",

[...omitted output...]

"ansible\_virtualization\_role": "guest",

"ansible\_virtualization\_type": "kvm",

"module\_setup": true

},

"changed": false

}

* + This JSON-formatted list contains the available facts that Ansible can eventually use.

### 1.5. Install a Package

1. Install an Apache server on the web server (servera):

# ansible web -b -m yum -a "name=httpd state=present"

servera.example.com | SUCCESS => {

"changed": true,

"msg": "",

"rc": 0,

"results": [

"Loaded plugins: search-disabled-repos\nResolving Dependencies\n--> Running transaction check\n---> Package httpd.x86\_64 0:2.4.6-45.el7 will be installed\n--> Processing Dependency: httpd-tools = 2.4.6-45.el7 for package: httpd-2.4.6-45.el7.x86\_64\n--> Processing Dependency: /etc/mime.types for package: httpd-2.4.6-45.el7.x86\_64\n--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.4.6-45.el7.x86\_64\n--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.4.6-45.el7.x86\_64\n--> Running transaction check\n---> Package apr.x86\_64 0:1.4.8-3.el7 will be installed\n---> Package apr-util.x86\_64 0:1.5.2-6.el7 will be installed\n---> Package httpd-tools.x86\_64 0:2.4.6-45.el7 will be installed\n---> Package mailcap.noarch 0:2.1.41-2.el7 will be installed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n================================================================================\n Package Arch Version Repository Size\n================================================================================\nInstalling:\n httpd x86\_64 2.4.6-45.el7 rhelosp-rhel-7.3-server 1.2 M\nInstalling for dependencies:\n apr x86\_64 1.4.8-3.el7 rhelosp-rhel-7.3-server 103 k\n apr-util x86\_64 1.5.2-6.el7 rhelosp-rhel-7.3-server 92 k\n httpd-tools x86\_64 2.4.6-45.el7 rhelosp-rhel-7.3-server 84 k\n mailcap noarch 2.1.41-2.el7 rhelosp-rhel-7.3-server 31 k\n\nTransaction Summary\n================================================================================\nInstall 1 Package (+4 Dependent packages)\n\nTotal download size: 1.5 M\nInstalled size: 4.3 M\nDownloading packages:\n--------------------------------------------------------------------------------\nTotal 1.4 MB/s | 1.5 MB 00:01 \nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\n Installing : apr-1.4.8-3.el7.x86\_64 1/5 \n Installing : apr-util-1.5.2-6.el7.x86\_64 2/5 \n Installing : httpd-tools-2.4.6-45.el7.x86\_64 3/5 \n Installing : mailcap-2.1.41-2.el7.noarch 4/5 \n Installing : httpd-2.4.6-45.el7.x86\_64 5/5 \n Verifying : httpd-tools-2.4.6-45.el7.x86\_64 1/5 \n Verifying : apr-1.4.8-3.el7.x86\_64 2/5 \n Verifying : mailcap-2.1.41-2.el7.noarch 3/5 \n Verifying : httpd-2.4.6-45.el7.x86\_64 4/5 \n Verifying : apr-util-1.5.2-6.el7.x86\_64 5/5 \n\nInstalled:\n httpd.x86\_64 0:2.4.6-45.el7 \n\nDependency Installed:\n apr.x86\_64 0:1.4.8-3.el7 apr-util.x86\_64 0:1.5.2-6.el7 \n httpd-tools.x86\_64 0:2.4.6-45.el7 mailcap.noarch 0:2.1.41-2.el7 \n\nComplete!\n"

]

}

|  |  |
| --- | --- |
|  | For information on the -b and -m options, check the [documentation](http://docs.ansible.com/ansible/intro_adhoc.html). |

1. Manually confirm that the package was installed:

# ssh servera.example.com "rpm -qa | grep httpd-[0-9]\*"

httpd-tools-2.4.6-45.el7.x86\_64

httpd-2.4.6-45.el7.x86\_64

### 1.6. Start a Service

1. Start the Apache server:

# ansible web -b -m service -a "name=httpd state=started"

servera.example.com | SUCCESS => {

"changed": true,

"name": "httpd",

"state": "started",

"status": {

"ActiveEnterTimestampMonotonic": "0",

"ActiveExitTimestampMonotonic": "0",

"ActiveState": "inactive",

[...omitted output...]

"UnitFilePreset": "disabled",

"UnitFileState": "disabled",

"Wants": "system.slice",

"WatchdogTimestampMonotonic": "0",

"WatchdogUSec": "0"

},

"warnings": []

}

1. Manually check that the server was started:

# ssh servera.example.com "systemctl status httpd"

● httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)

Active: active (running) since Mon 2017-02-27 17:20:16 EST; 6min ago

[...omitted output...]

## 2. Use ansible-playbook Command Line

In this exercise, you use the Ansible ansible-playbook command line to do the following:

* Structure a basic playbook
* Install a web server
* Start the web server

### 2.1. Create a Simple Playbook

1. Create a site.yml file containing the playbook you reviewed in Lab 3:

---

- hosts: web

name: Install the web server and start it

become: yes

vars:

httpd\_packages:

- httpd

- mod\_wsgi

apache\_test\_message: This is a test message

apache\_max\_keep\_alive\_requests: 115

tasks:

- name: Install the apache web server

yum:

name: "{{ item }}"

state: present

with\_items: "{{ httpd\_packages }}"

notify: restart apache service

- name: Generate apache's configuration file from jinja2 template

template:

src: templates/httpd.conf.j2

dest: /etc/httpd/conf/httpd.conf

notify: restart apache service

- name: Generate a basic homepage from jinja2 template

template:

src: templates/index.html.j2

dest: /var/www/html/index.html

- name: Start the apache web server

service:

name: httpd

state: started

enabled: yes

handlers:

- name: restart apache service

service:

name: httpd

state: restarted

enabled: yes

### 2.2. Add Jinja2 Template Files

|  |  |
| --- | --- |
|  | You can learn more about jinja2 templates [here](http://jinja.pocoo.org/docs). |

In this section, you add the Jinja2 template files needed for the configuration of the web server.

1. Create a templates folder:

$ mkdir templates

1. Add the httpd.conf.j2 template to the folder:

ServerRoot "/etc/httpd"

Listen 80

Include conf.modules.d/\*.conf

User apache

Group apache

ServerAdmin root@localhost

<Directory />

AllowOverride none

Require all denied

</Directory>

DocumentRoot "/var/www/html"

<Directory "/var/www">

AllowOverride None

Require all granted

</Directory>

<Directory "/var/www/html">

Options Indexes FollowSymLinks

AllowOverride None

Require all granted

</Directory>

<IfModule dir\_module>

DirectoryIndex index.html

</IfModule>

<Files ".ht\*">

Require all denied

</Files>

ErrorLog "logs/error\_log"

MaxKeepAliveRequests {{ apache\_max\_keep\_alive\_requests }}

LogLevel warn

<IfModule log\_config\_module>

LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined

LogFormat "%h %l %u %t \"%r\" %>s %b" common

<IfModule logio\_module>

LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O" combinedio

</IfModule>

CustomLog "logs/access\_log" combined

</IfModule>

<IfModule alias\_module>

ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"

</IfModule>

<Directory "/var/www/cgi-bin">

AllowOverride None

Options None

Require all granted

</Directory>

<IfModule mime\_module>

TypesConfig /etc/mime.types

AddType application/x-compress .Z

AddType application/x-gzip .gz .tgz

AddType text/html .shtml

AddOutputFilter INCLUDES .shtml

</IfModule>

AddDefaultCharset UTF-8

<IfModule mime\_magic\_module>

MIMEMagicFile conf/magic

</IfModule>

EnableSendfile on

IncludeOptional conf.d/\*.conf

1. Add the index.html.j2 template to the folder:

{{ apache\_test\_message }} {{ ansible\_distribution }} {{ ansible\_distribution\_version }} <br>

Current Host: {{ ansible\_hostname }} <br>

Server list: <br>

{% for host in groups['web'] %}

{{ host }} <br>

{% endfor %}

1. Confirm that the folder structure looks like this:

└── templates

├── httpd.conf.j2

└── index.html.j2

### 2.3. Run Playbook

1. Run the Ansible playbook using the ansible-playbook command:

# ansible-playbook site.yml

PLAY [Install the web server and start it] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [setup] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera.example.com]

TASK [Install the apache web server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera.example.com] => (item=[u'httpd', u'mod\_wsgi'])

TASK [Generate apache's configuration file from jinja2 template] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera.example.com]

TASK [Generate a basic homepage from jinja2 template] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [servera.example.com]

TASK [Start the apache web server] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera.example.com]

RUNNING HANDLER [restart apache service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [servera.example.com]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

servera.example.com : ok=6 changed=4 unreachable=0 failed=0

1. Confirm that the web server responds and serves the index.html file generated by Ansible:

# curl servera.example.com

This is a test message RedHat 7.3 <br>

Current Host: servera <br>

Server list: <br>

servera.example.com <br>

## Installation Ansible ower

* Most common and preferred method:

$ sudo wget https://bit.ly/towerlatest -O ansible-tower-setup-latest.tar.gz

$ tar -xvzf ansible-tower-setup-latest.tar.gz

$ cd ansible-tower-setup-\*

$ ./setup.sh

* If no access to EPEL, use <https://bit.ly/towerbundlelatest>

|  |  |
| --- | --- |
|  | Bundled installer still needs rhel-7-server-optional-rpms and rhel-7-server-extras-rpms enabled. |