Table of Contents

* [Ansible Deployment Lab](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_ansible_deployment_lab)
* [1. Prepare for Labs](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_prepare_for_labs)
  + [1.1. Become](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_become_code_student_code_user) student User
* [2. Install and Configure Ansible](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_install_and_configure_ansible)
  + [2.1. Set Up Environment](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_set_up_environment)
  + [2.2. Install Ansible and Confirm SSH Connection](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_install_ansible_and_confirm_ssh_connection)
  + [2.3. Create and Use Inventory File](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_create_and_use_inventory_file)
  + [2.4. Evaluate Your Progress](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_evaluate_your_progress)
* [3. Manage Ansible Configuration Files](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_manage_ansible_configuration_files)
  + [3.1. Set Up Environment](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_set_up_environment_2)
  + [3.2. Create Ansible Configuration File](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_create_ansible_configuration_file)
  + [3.3. Configure Inventory File](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_configure_inventory_file)
  + [3.4. Add Privilege Escalation to](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_add_privilege_escalation_to_code_ansible_cfg_code) ansible.cfg
  + [3.5. Check Files](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_check_files)
  + [3.6. Evaluate Your Progress](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_evaluate_your_progress_2)
* [4. Run Ad Hoc Commands](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_run_ad_hoc_commands)
  + [4.1. Set Up Environment](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_set_up_environment_3)
  + [4.2. Execute Ad Hoc Commands](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_execute_ad_hoc_commands)
  + [4.3. Evaluate Your Progress](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_evaluate_your_progress_3)
* [5. Manage Dynamic Inventory](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_manage_dynamic_inventory)
  + [5.1. Set Up Environment](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_set_up_environment_4)
  + [5.2. Set Up Inventory](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_set_up_inventory)
  + [5.3. Work With Inventory Scripts](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_work_with_inventory_scripts)
  + [5.4. Evaluate Your Progress](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_evaluate_your_progress_4)
* [6. Deploy Ansible](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_deploy_ansible)
  + [6.1. Set Up Environment](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_set_up_environment_5)
  + [6.2. Configure Control Node](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_configure_control_node)
  + [6.3. Set Up Inventory Files](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_set_up_inventory_files)
  + [6.4. Use Ad Hoc Commands](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_use_ad_hoc_commands)
  + [6.5. Evaluate Your Progress](https://cloud.scorm.com/content/courses/A9KI96X2QE/Ansible_Implementation/13/03_Ansible_Deployment/Lab_03_Deploy_Ansible.html" \l "_evaluate_your_progress_5)

## Ansible Deployment Lab

Goals

* Connect to a managed host from a control node using Ansible
* Configure the Ansible environment with persistent custom settings
* Execute commands on managed hosts on an ad hoc basis using privilege escalation
* Manage a dynamic inventory
* Configure an Ansible environment and conduct operations on managed hosts using ad hoc commands

## 1. Prepare for Labs

In this section, you learn the proper way to become the student user.

### 1.1. Become student User

|  |  |
| --- | --- |
|  | Going forward, whenever you are asked to log in to the workstation host as the student user, you must first log in to workstation using your OPENTLC SSO username and password. See the first lab if you need more information on how to do this. |

1. Become the student user:
   1. Log in to workstation with your OPENTLC SSO credentials:

$ ssh <opentlc\_sso\_username>@workstation-GUID.rhpds.opentlc.com

* 1. After accessing the system, use sudo to become the student user:

$ sudo -u student -i

## 2. Install and Configure Ansible

In this exercise, you install Ansible on a control node and configure it for connections to a managed host.

### 2.1. Set Up Environment

1. Log in as the student user on workstation and run lab install setup:

[student@workstation ~]$ lab install setup

* + This setup script ensures that the managed host, servera, is reachable on the network.

1. Verify that the prerequisite version of Python is installed on workstation:

[student@workstation ~]$ yum list installed python

### 2.2. Install Ansible and Confirm SSH Connection

1. Install Ansible on workstation:

[student@workstation ~]$ sudo yum install -y ansible

* + This enables workstation to serve as the control node.

1. Verify that you can establish an SSH connection from workstation to servera:

[student@workstation ~]$ ssh servera.lab.example.com

Warning: Permanently added 'servera.lab.example.com,172.25.250.10' (ECDSA) to the list of known hosts.

* + This ensures that Ansible is able to connect from the control node to the managed host using SSH.

1. Exit the SSH session:

[student@servera ~]$ exit

### 2.3. Create and Use Inventory File

1. On the control node, create an inventory file, named /home/student/dep-install/inventory, that contains a group called dev made up of a single managed host, servera.lab.example.com:
   1. Create and change to the /home/student/dep-install directory:

[student@workstation ~]$ mkdir /home/student/dep-install

[student@workstation ~]$ cd /home/student/dep-install

* 1. Create the inventory file and add the following entries to the file to create the dev host group and its single member servera.lab.example.com:

[dev]

servera.lab.example.com

1. Execute the following ansible command with the -i option to use the newly created inventory file:

[student@workstation dep-install]$ ansible dev -i inventory --list-hosts

hosts (1):

servera.lab.example.com

* 1. The --list-hosts option enables you to verify that the dev host group resolves to its only member, servera.

### 2.4. Evaluate Your Progress

1. Grade your work:

[student@workstation ~]$ lab install grade

1. Correct any reported failures.
2. Rerun the script until successful.

## 3. Manage Ansible Configuration Files

In this exercise, you customize your Ansible environment with persistent settings.

### 3.1. Set Up Environment

1. Log in as the student user on workstation and run lab manage setup:

[student@workstation ~]$ lab manage setup

1. Create and change to the /home/student/dep-manage directory:

[student@workstation ~]$ mkdir /home/student/dep-manage

[student@workstation ~]$ cd /home/student/dep-manage

* + You use this directory to hold the files for this exercise.

### 3.2. Create Ansible Configuration File

1. Create a directory-level Ansible configuration file, ansible.cfg, in the newly created directory.
2. Place the following configuration entries in the ansible.cfg file:

[defaults]

inventory=/home/student/dep-manage/inventory

* + This ensures that the file configures Ansible to use an inventory located at /home/student/dep-manage/inventory.

### 3.3. Configure Inventory File

1. In the directory-level inventory file, create a myself host group consisting of the following:
   * The localhost host
   * An intranetweb host group consisting of the servera.lab.example.com host
   * An everyone host group comprised of the myself and intranetweb groups
     1. Create the myself host group by adding the following entries:

[myself]

localhost

* + 1. Create the intranetweb host group by adding the following entries:

[intranetweb]

servera.lab.example.com

* + 1. Create the everyone host group by adding the following entries:

[everyone:children]

myself

intranetweb

1. Verify the resolution of the myself host group with the ansible command:

[student@workstation dep-manage]$ ansible myself -i inventory --list-hosts

hosts (1):

localhost

* + The --list-hosts option lists the managed hosts referenced by the host group.

### 3.4. Add Privilege Escalation to ansible.cfg

In this exercise, you activate and configure privilege escalation by creating the [privilege\_escalation] section of the ansible.cfg file.

1. Create the privilege\_escalation section:

[privilege\_escalation]

1. Add the following entry to the privilege\_escalation section to enable privilege escalation:

become=True

1. Add the following entry to set the privilege escalation to use sudo:

become\_method=sudo

1. Add the following entry to set the privilege escalation user:

become\_user=root

1. Add the following entry to enable password prompting during privilege escalation:

become\_ask\_pass=True

### 3.5. Check Files

1. Verify your changes to the inventory and ansible.cfg files:
   1. Use ansible with the --list-hosts option.
      * The --list-hosts option resolves the intranetweb host group to its managed host members.
   2. When prompted for the SUDO password, enter student.

[student@workstation dep-manage]$ ansible intranetweb -i inventory --list-hosts -v

Using /home/student/dep-manage/ansible.cfg as config file

SUDO password: student

hosts (1):

servera.lab.example.com

* + - Note that privilege escalation is occurring using sudo and password prompting.
    - Also note that it reports that /home/student/dep-manage/ansible.cfg is the configuration file in use.

1. Verify the resolution of the everyone host group with the ansible command:
   1. Execute the command in verbose mode to verify which configuration file is being used.
   2. When prompted for the SUDO password, enter student.

[student@workstation dep-manage]$ ansible everyone --list-hosts -v

Using /home/student/dep-manage/ansible.cfg as config file

SUDO password: student

hosts (2):

localhost

servera.lab.example.com

### 3.6. Evaluate Your Progress

1. Grade your work:

[student@workstation ~]$ lab manage grade

1. Correct any reported failures.
2. Rerun the script until successful.

## 4. Run Ad Hoc Commands

In this exercise, you execute ad hoc commands on multiple managed hosts using privilege escalation. You execute these commands on workstation and servera using the devops user account. This account has the same sudo configuration on both workstation and servera.

### 4.1. Set Up Environment

1. Log in as the student user on workstation and run lab adhoc setup:

[student@workstation ~]$ lab adhoc setup

* + This setup script ensures that the managed host, servera, is reachable on the network and creates and populates the /home/student/dep-adhoc working directory with materials used in this exercise.

1. Determine the sudo configuration for the devops account configured when workstation was built:

|  |  |
| --- | --- |
|  | The password for the student account is student. |

[student@workstation ~]$ sudo cat /etc/sudoers.d/devops

[sudo] password for student: student

devops ALL=(ALL) NOPASSWD: ALL

* + Note that the user has full sudo privileges but does not require password authentication.

1. Determine the sudo configuration for the devops account configured when servera was built:

[student@workstation ~]$ ssh devops@servera.lab.example.com

[devops@servera ~]$ sudo cat /etc/sudoers.d/devops

devops ALL=(ALL) NOPASSWD: ALL

[devops@servera ~]$ exit

* + Note that the user has full sudo privileges but does not require password authentication.

1. Change to the /home/student/dep-adhoc directory and examine the contents of the ansible.cfg and inventory files:

[student@workstation ~]$ cd /home/student/dep-adhoc

[student@workstation dep-adhoc]$ cat ansible.cfg

[defaults]

inventory=inventory

[student@workstation dep-adhoc]$ cat inventory

[myself]

localhost

[intranetweb]

servera.lab.example.com

[everyone:children]

myself

intranetweb

* + The configuration file in the directory configures the inventory file as the Ansible inventory.

### 4.2. Execute Ad Hoc Commands

1. Using the command module, execute an ad hoc command on workstation to identify the user account used by Ansible to perform operations on managed hosts:
   1. Use the localhost host pattern to connect to workstation.

|  |  |
| --- | --- |
|  | Because you are connecting locally, workstation is both the control node and managed host. |

[student@workstation dep-adhoc]$ ansible localhost -m command -a 'id'

localhost | SUCCESS | rc=0 >>

uid=1000(student) gid=1000(student) groups=1000(student),10(wheel) context=unconfined\_u:unconfined\_r:unconfined\_t:s0-s0:c0.c1023

* + - Expect to see that the ad hoc command is performed on the managed host as the student user.

1. Repeat the previous ad hoc command, but use the -u option to connect and perform the operation with the devops account:

[student@workstation dep-adhoc]$ ansible localhost -m command -a 'id' -u devops

localhost | SUCCESS | rc=0 >>

uid=1001(devops) gid=1001(devops) groups=1001(devops) context=unconfined\_u:unconfined\_r:unconfined\_t:s0-s0:c0.c1023

1. Using the command module and the devops account, execute an ad hoc command to display the contents of the /etc/motd file:

[student@workstation dep-adhoc]$ ansible localhost -m command -a 'cat /etc/motd' -u devops

localhost | SUCCESS | rc=0 >>

* 1. Note that the /etc/motd file is currently empty.

1. Using the copy module and the devops account, execute an ad hoc command to change the contents of the /etc/motd file to display the message Managed by Ansible:

[student@workstation dep-adhoc]$ ansible localhost -m copy -a 'content="Managed by Ansible\n" dest=/etc/motd' -u devops

localhost | FAILED! => {

"changed": false,

"checksum": "4458b979ede3c332f8f2128385df4ba305e58c27",

"failed": true,

"msg": "Destination /etc not writable"

}

* 1. Expect the ad hoc command to fail due to insufficient permissions.

1. Repeat the previous command using the devops user to connect to the managed host, but perform the operation as the root user:

[student@workstation dep-adhoc]$ ansible localhost -m copy -a 'content="Managed by Ansible\n" dest=/etc/motd' -u devops --become --become-user root

localhost | SUCCESS => {

"changed": true,

"checksum": "4458b979ede3c332f8f2128385df4ba305e58c27",

"dest": "/etc/motd",

"gid": 0,

"group": "root",

"md5sum": "65a4290ee5559756ad04e558b0e0c4e3",

"mode": "0644",

"owner": "root",

"secontext": "system\_u:object\_r:etc\_t:s0",

"size": 19,

"src": "/home/devops/.ansible/tmp/ansible-tmp-1463518320.68-167292050637471/source",

"state": "file",

"uid": 0

}

* 1. Note that the command succeeded this time because the ad hoc command was executed with privilege escalation.

1. Repeat the previous ad hoc command on servera:

[student@workstation dep-adhoc]$ ansible servera.lab.example.com -m copy -a 'content="Managed by Ansible\n" dest=/etc/motd' -u devops --become --become-user root

servera.lab.example.com | SUCCESS => {

"changed": true,

"checksum": "4458b979ede3c332f8f2128385df4ba305e58c27",

"dest": "/etc/motd",

"gid": 0,

"group": "root",

"md5sum": "65a4290ee5559756ad04e558b0e0c4e3",

"mode": "0644",

"owner": "root",

"secontext": "system\_u:object\_r:etc\_t:s0",

"size": 19,

"src": "/home/devops/.ansible/tmp/ansible-tmp-1464120223.42-93111008249456/source",

"state": "file",

"uid": 0

}

1. Execute an ad hoc command to verify the changes to /etc/motd on workstation and servera:
   1. Use the everyone host group and the devops user to make the connection and perform the operation.

[student@workstation dep-adhoc]$ ansible everyone -m command -a 'cat /etc/motd' -u devops

servera.lab.example.com | SUCCESS | rc=0 >>

Managed by Ansible

localhost | SUCCESS | rc=0 >>

Managed by Ansible

### 4.3. Evaluate Your Progress

1. Grade your work:

[student@workstation ~]$ lab adhoc grade

1. Correct any reported failures.
2. Rerun the script until successful.

## 5. Manage Dynamic Inventory

In this exercise, you fix a custom script that retrieves a dynamic inventory.

### 5.1. Set Up Environment

1. Log in to workstation as student and run the lab deploy-dynamic setup script:

[student@workstation ~]$ lab deploy-dynamic setup

* + This script checks if Ansible is installed on workstation and also creates a working directory for this exercise.

1. Change to the working directory for the exercise, /home/student/dep-dynamic:

[student@workstation ~]$ cd /home/student/dep-dynamic

### 5.2. Set Up Inventory

1. Create an Ansible configuration file in the working directory and populate it as follows:

[defaults]

inventory = inventory

* + This ensures that the inventory directory is configured as the default inventory.

1. Create the /home/student/dep-dynamic/inventory directory:

[student@workstation dep-dynamic]$ mkdir inventory

1. Download the inventorya.py, inventoryw.py, and hosts files from <http://materials.example.com/dynamic/> to the inventory directory:

[student@workstation dep-dynamic]$ wget http://materials.example.com/dynamic/inventorya.py -O inventory/inventorya.py

[student@workstation dep-dynamic]$ wget http://materials.example.com/dynamic/inventoryw.py -O inventory/inventoryw.py

[student@workstation dep-dynamic]$ wget http://materials.example.com/dynamic/hosts -O inventory/hosts

* + The files ending in .py are scripts that retrieve dynamic inventories.
  + The dynamic inventory used by the inventorya.py script has the webservers group, which includes the servera.lab.example.com host.
  + The dynamic inventory used by the inventoryw.py script includes the workstation.lab.example.com host.
  + The hosts file defines the servers group, which is a parent group of the webservers group.

### 5.3. Work With Inventory Scripts

1. List the managed hosts associated with the webservers group using the inventorya.py script:

[student@workstation dep-dynamic]$ ansible -i inventory/inventorya.py webservers --list-hosts

ERROR! The file inventory/inventorya.py looks like it should be an

executable inventory script, but is not marked executable. Perhaps you

want to correct this with `chmod +x inventory/inventory.py`?

* + This raises an error about the permissions on the dynamic inventory script.

1. Check the current permissions for the inventorya.py script, and change them to 755:

[student@workstation dep-dynamic]$ ls -la inventory/inventorya.py

-rw-r--r--. 1 student student 0 Apr 29 14:20 inventory/inventorya.py

[student@workstation dep-dynamic]$ chmod 755 inventory/inventorya.py

1. Change the permissions for the inventoryw.py script to 755:

[student@workstation dep-dynamic]$ chmod 755 inventory/inventoryw.py

1. Check the current output for the inventorya.py script using the --list parameter:

[student@workstation dep-dynamic]$ inventory/inventorya.py --list

{"webservers": {"hosts": ["servera.lab.example.com"], "vars": {} } }

* + The hosts associated with the webservers group are displayed.

1. Repeat the previous command for the inventoryw.py script:

[student@workstation dep-dynamic]$ inventory/inventoryw.py --list

{"all": {"hosts": ["workstation.lab.example.com"], "vars": {} } }

* + The workstation.lab.example.com host is displayed.

1. Check the servers group definition in the /home/student/dep-dynamic/inventory/hosts file:

[student@workstation dep-dynamic]$ cat inventory/hosts

[servers:children]

webservers

* + The webservers group defined in the dynamic inventory is configured as a child of the servers group.

1. Verify the list of hosts in the webservers group:

[student@workstation dep-dynamic]$ ansible webservers --list-hosts

ERROR! inventory/hosts:46: Section [servers:children] includes undefined group: webservers

* + Because the static and dynamic inventories are executed in alphabetical order, the inventorya.py script, which defines the webservers group, must be executed before the hosts file, which includes the webservers group as a dependency.

1. Rename the inventorya.py script to ainventory.py to solve the previous error:

[student@workstation dep-dynamic]$ mv inventory/inventorya.py inventory/ainventory.py

1. Verify the list of hosts in the webservers group:

[student@workstation dep-dynamic]$ ansible webservers --list

hosts (1):

servera.lab.example.com

* + Expect it to work without errors.

### 5.4. Evaluate Your Progress

1. Grade your work:

[student@workstation ~]$ lab deploy-dynamic grade

1. Correct any reported failures.
2. Rerun the script until successful.

## 6. Deploy Ansible

In this exercise, you configure an Ansible control node for connections to inventory hosts and use ad hoc commands to perform actions on managed hosts.

You have been asked to configure workstation as an Ansible control node so that it can be used to manage a new managed host, serverb. The new managed host is used to host Internet-facing websites.

This exercise takes you through all of the steps from verifying that the prerequisite software is installed to confirming that the ad hoc commands work as desired.

### 6.1. Set Up Environment

1. Log in as the student user on workstation and run lab deploy setup:

[student@workstation ~]$ lab deploy setup

* + This setup script ensures that the managed host, serverb, is reachable on the network.

1. Verify the Ansible installation on the control node:
   * Verify that the ansible package is installed:

[student@workstation ~]$ yum list installed ansible

Installed Packages

ansible.noarch 2.0.1.0-2.el7 @ansible

* + Determine the configuration file that is in use by the Ansible installation:

[student@workstation ~]$ ansible --version

ansible 2.0.1.0

config file = /etc/ansible/ansible.cfg

configured module search path = Default w/o overrides

1. Verify that sshd is running on the managed host and accepts connections with key authentication from the control node:
   * Verify that the SSH daemon is running on the managed host:

[student@workstation ~]$ ssh serverb.lab.example.com 'hostname'

Warning: Permanently added 'serverb.lab.example.com,172.25.250.11' (ECDSA) to the list of known hosts.

serverb.lab.example.com

* + Verify that the control node can connect to the managed host as the devops user with key authentication:

[student@workstation ~]$ ssh devops@serverb.lab.example.com

Last login: Thu Mar 31 16:07:06 2016 from workstation.lab.example.com

[devops@serverb ~]$

* + Verify that the devops user has full sudo privileges and does not require password authentication:

[devops@serverb ~]$ sudo -l

Matching Defaults entries for devops on this host:

requiretty, !visiblepw, always\_set\_home, env\_reset, env\_keep="COLORS DISPLAY HOSTNAME HISTSIZE INPUTRC KDEDIR LS\_COLORS",

env\_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC\_ADDRESS LC\_CTYPE", env\_keep+="LC\_COLLATE LC\_IDENTIFICATION LC\_MEASUREMENT

LC\_MESSAGES", env\_keep+="LC\_MONETARY LC\_NAME LC\_NUMERIC LC\_PAPER LC\_TELEPHONE", env\_keep+="LC\_TIME LC\_ALL LANGUAGE LINGUAS

\_XKB\_CHARSET XAUTHORITY", secure\_path=/sbin\:/bin\:/usr/sbin\:/usr/bin\:/usr/local/sbin\:/usr/local/bin

User devops may run the following commands on this host:

(ALL) NOPASSWD: ALL

1. Exit the SSH session when verification is complete:

[devops@serverb ~]$ exit

### 6.2. Configure Control Node

In this exercise, you configure the control node to connect to managed hosts as the devops user and set privilege escalation to be disabled by default.

|  |  |
| --- | --- |
|  | It is good practice to use the least privilege required for operations. If privilege escalation is performed, set the configuration to execute it using sudo with the root user account and no password authentication. |

1. Create and change to the working directory for the exercise:

[student@workstation ~]$ mkdir /home/student/dep-lab

[student@workstation ~]$ cd /home/student/dep-lab

1. Create the ansible.cfg file and add the following entries to create the [defaults] section:

[defaults]

remote\_user=devops

inventory=inventory

* + Note that this sets connections to managed hosts to use the devops account on the managed hosts, and sets Ansible to use /home/student/dep-lab/inventory as the default inventory directory.

1. Create the [privilege\_escalation] section and add the following entries:

[privilege\_escalation]

become=False

become\_method=sudo

become\_user=root

become\_ask\_pass=False

* + These entries disable privilege escalation and set the privilege escalation method to use the root account with sudo and without password authentication.

### 6.3. Set Up Inventory Files

In this section, you create the /home/student/dep-lab/inventory directory and download both the static inventory file and the dynamic inventory script to it.

1. Create the /home/student/dep-lab/inventory directory:

[student@workstation dep-lab]$ mkdir inventory

1. Download the inventory file to the inventory directory:

[student@workstation dep-lab]$ wget http://materials.example.com/dynamic/inventory -O inventory/inventory

1. Download the binventory.py script to the inventory directory, and change its permission to 755:

[student@workstation dep-lab]$ wget http://materials.example.com/dynamic/binventory.py -O inventory/binventory.py

[student@workstation dep-lab]$ chmod 755 inventory/binventory.py

1. Configure the internetweb group as a child of the everyone group by adding the following entry to the inventory file:

...

[everyone:children]

myself

intranetweb

internetweb

... Output omitted ...

### 6.4. Use Ad Hoc Commands

1. Execute an ad hoc command using privilege escalation to verify that devops is the remote user and that privilege escalation is disabled by default:

[student@workstation dep-lab]$ ansible serverb.lab.example.com -m command -a 'id'

serverb.lab.example.com | SUCCESS | rc=0 >>

uid=1001(devops) gid=1001(devops) groups=1001(devops) context=unconfined\_u:

unconfined\_r:unconfined\_t:s0-s0:c0.c1023

1. Execute an ad hoc command using the copy module and privilege escalation to modify the contents of the /etc/motd file on serverb:

[student@workstation dep-lab]$ ansible serverb.lab.example.com -m copy -a 'content="This server is managed by Ansible.\n" dest=/etc/motd' --become

serverb.lab.example.com | SUCCESS => {

"changed": true,

"checksum": "1e2105d6df22c8d696bc82d3c613cbcb88f36fa5",

"dest": "/etc/motd",

"gid": 0,

"group": "root",

"md5sum": "3e6ebda65f0497d3af806622baf7b625",

"mode": "0644",

"owner": "root",

"secontext": "system\_u:object\_r:etc\_t:s0",

"size": 34,

"src": "/home/devops/.ansible/tmp/ansible-tmp-1464121888.97-234193031694771/source",

"state": "file",

"uid": 0

}

1. Use an ad hoc command to verify the contents of the /etc/motd file on serverb:

[student@workstation dep-lab]$ ansible serverb.lab.example.com -m command -a 'cat /etc/motd'

serverb.lab.example.com | SUCCESS | rc=0 >>

This server is managed by Ansible.

* + Expect the contents of the file to display the message This server is managed by Ansible..

### 6.5. Evaluate Your Progress

1. Grade your work:

[student@workstation dep-lab]$ lab deploy grade

1. Correct any reported failures.
2. Rerun the script until successful.