

**Student Name:**

**Weight: 20%**

**Student ID:**

**Marks: /20**

## Assignment: Automating Server Updates

### Scenario

As an IT administrator, one of your responsibilities is to ensure that all of the servers managed by your office are up-to-date and have the correct software installed. This task is extremely time-consuming (and tedious), so you want to show your managers that there are tools available to automate this task.

In this assignment, you will create an environment in which you deploy an Ansible configuration. The goal is to have a network environment configured with one host (controller) machine and several servers managed by the host.

Your automation will start by running a Python script. You will then set up an automation that uses the following Ansible components:

- A static inventory file (list of server node IPs)
- A playbook (easily manipulated to manage the automation)

**Note:** There are many different ways to configure Ansible and set up Python for this automation.

### Equipment and Materials

For this assessment, you will need:

- Three Linux virtual machines

### Instructions

1. Working individually, design a network that demonstrates the effectiveness of automated server management. Your network must include three Linux virtual machines:
  - a. Host (controller)
  - b. Server A
  - c. Server B

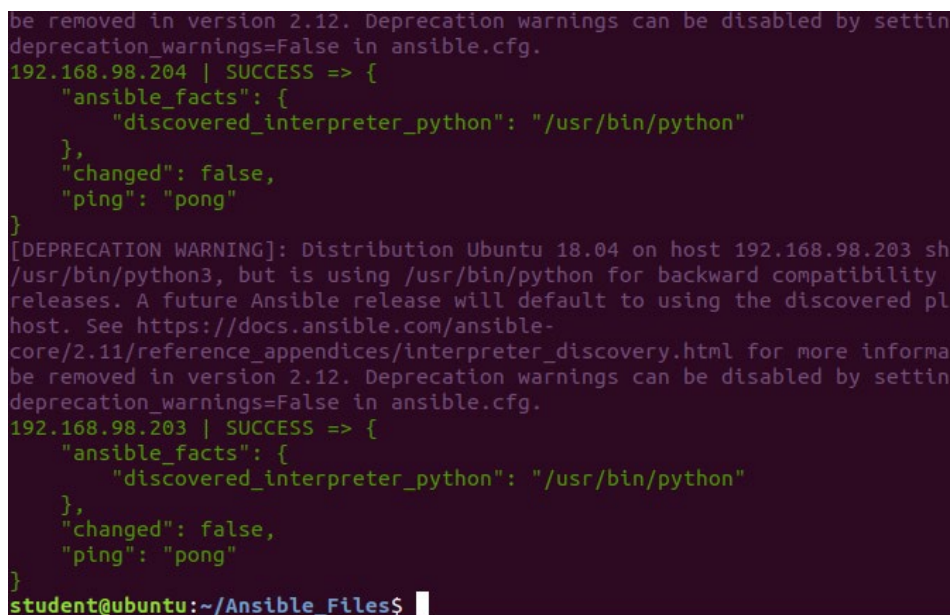
**Note:** Make sure all three machines are on the same network and have ssh installed.

2. Install Ansible on your host machine and review the basic concepts discussed in the following video: [Red Hat Certified Engineer \(EX294\) Cert Prep: 1 Foundations of Ansible](https://www.linkedin.com/learning/red-hat-certified-engineer-ex294-cert-prep-1-foundations-of-ansible) ([https://www.linkedin.com/learning/red-hat-certified-engineer-ex294-cert-prep-1-foundations-](https://www.linkedin.com/learning/red-hat-certified-engineer-ex294-cert-prep-1-foundations-of-ansible)

of-ansible/ansible-

concepts?autoAdvance=true&autoSkip=false&autoplay=true&resume=true&u=2245281).

3. Configure an automatic ssh connection between the controller and server nodes using the “ssh-keygen” on the controller (to generate private and public keys) and then the command “ssh-copy-id” (to copy the public key to the server nodes).
4. Check to make sure you can initiate ssh from the controller to nodes without any authentication.
5. Create a folder that will contain your ansible files (“Ansible\_Files”) where your inventory, playbook, and python script will reside.
6. Create the inventory file that will contain the Server A and Server B IP addresses.
7. Test for the following:
  - On the controller, run the following command: Ansible all -i inventory -m ping
  - Setup is successful if you don’t get prompted for authentication and you get success results to both server nodes (as shown in Figure 2 below):



```
be removed in version 2.12. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.
192.168.98.204 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
[DEPRECATION WARNING]: Distribution Ubuntu 18.04 on host 192.168.98.203 sh
/usr/bin/python3, but is using /usr/bin/python for backward compatibility
releases. A future Ansible release will default to using the discovered pl
host. See https://docs.ansible.com/ansible-
core/2.11/reference_appendices/interpreter_discovery.html for more informa
be removed in version 2.12. Deprecation warnings can be disabled by settin
deprecation_warnings=False in ansible.cfg.
192.168.98.203 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
student@ubuntu:~/Ansible_Files$
```

**Figure 1: Screen Capture of Successful Ping**

Source: Southern Alberta Institute of Technology

8. Create a playbook file called Paybook\_1.yml to run two tasks:
  - a. System update task (install, update, system change of some kind)
  - b. Raw bash task that will create a file with content including your name.

**Note:** Test one task at a time to make sure everything works.
9. Test your playbook to make sure it works (i.e., that it will give you a successful result).

```
student@ubuntu:~/Ansible_Files$ ansible-playbook --ask-become-pass -i inventory playbook_1.yml
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with
2.12. Current version: 3.6.9 (default, Dec  8 2021, 21:08:43) [GCC 8.4.0]. This feature will be
ansible-core in version 2.12. Deprecation warnings can be disabled by setting deprecation_warr
ansible.cfg.
BECOME password:

PLAY [all] *****

TASK [Write message to File] *****
changed: [192.168.98.204]
changed: [192.168.98.203]

TASK [Show Variables] *****
ok: [192.168.98.204] => {
  "msg": "Username: Marcel Tozser, Class: CPRG217"
}
ok: [192.168.98.203] => {
  "msg": "Username: Marcel Tozser, Class: CPRG217"
}

PLAY RECAP *****
192.168.98.203      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    res
red=0
192.168.98.204      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    res
red=0
```

**Figure 2: Screen Capture of Successful Deployment of Playbook**

Source: Southern Alberta Institute of Technology

10. To run your Ansible playbook, create a python script called Project\_5.py that will run from the terminal. There are many different ways that this can be accomplished:

Example:

- PlaybookCLI function
- call function
- Etc.

11. In your demo, you need to show that your Python script can be executed and that Ansible completed successfully.

- a. Make sure your demo shows the successful result screen.
- b. Check each of the nodes to show that a file has been created with the content and the history of the system update.

## Deliverables

- Python\_5.py
- Playbook\_1.yml
- Demo

## Marking Criteria

Categories	0	1	2	Score
<b>Ansible Connection to Server A</b>	No		Yes: Shows success on the running ansible with the -m gather_facts command	/2
<b>Ansible Connection to Server B</b>	No		Yes: Shows success on the running ansible with the -m ping command	/2
<b>Task 1 running Ansible-playbook command on Server A</b>	No	Attempted but no system task ran on systems. Task needs to be a system task that will perform a useful admin operation that would otherwise be performed in person.	Yes: shows success	/2
<b>Task 1 running Ansible-playbook command on Server B</b>	No	Attempted but no system task ran on systems. Task needs to be a system task that will perform a useful admin operation that would otherwise be performed in person.	Yes: shows success	/2
<b>Task 2 running Ansible-playbook command on Server A</b>	No	Attempted but no file is created	Yes: shows success	/2
<b>Task 2 running Ansible-playbook command on Server B</b>	No	Attempted but no file is created	Yes: shows success	/2

<b>Using Variables in Playbook</b>	No	Attempted to use variables but there is insufficient implementation of variable values.	Yes: shows success	<b>/2</b>
<b>Python_5.py</b>	None	Attempted but code does not run Ansible without errors in all cases.	Yes: executing Ansible	<b>/2</b>
<b>Python_5.py Readability</b>	No	Limited (missing adequate pseudocode, commenting). Or, code structure is difficult to follow.	Yes: executing Ansible	<b>/2</b>
<b>Demo</b>	No	Limited (problems encountered during the demo without adequate resolution, does not prove that coding solution works, or missing code explanation).	Well executed	<b>/2</b>
<b>Total</b>				<b>/20</b>