Guided Practice: Handling Complex Deployments

Introduction

Ansible's strengths include the ability to facilitate the configuration of hundreds, or even thousands, of computers that include groups that may be dependent on other groups, which can involve the secure backing up of data as well as replication.

Outcome

In this Guided Practice, you will edit the Ansible Inventory (hosts) file creating and using groups. You will then create and run Ansible playbooks that include conditional logic from your CentOS computer to monitor and configure the different operating systems on your network.

Resources Needed

- For this Guided Practice we will use CentOS 8, Windows 2019 server, and the Ubuntu 20.04 LTS machines in the VCastle pod configured for this class.

Level of Difficulty

Moderate to high

Deliverables

Deliverables are marked with a red border around the screenshot. Additionally, there are guided practice questions at the end which you must respond to. **Your username or studentID should be visible in all screenshots that you submit.**

General Considerations

You should be familiar with Linux networking. Secure Shell (ssh) should already be configured on your Ubuntu computer from a prior Guided Practice. Ansible should be installed and configured on your CentOS Computer from a prior Guided Practice. Optimizing your text editor for writing playbooks in YAML will simplify the creation and editing of playbooks.

Edit the Ansible Inventory (host) File to Reflect Your Network Architecture

1. Edit the inventory (hosts) file. **Please note** that the IP addresses shown in the examples may not reflect your network configuration. The webserver and dbserver IP address for your network will most likely be 192.168.1.3, and the win machine 192.168.1.2. The multiple entries are intended to demonstrate more complex scenarios possible with Ansible. The [win:vars] section must be entered as shown.

   The examples have been developed using vim as an editor configured for YAML, as shown in a previous Guided Practice. Ensure your inventory file reflects the architecture of your network as shown below.

```
[webserver]
192.168.1.3

[dbserver]
192.168.1.3

[win]
192.168.1.2

[win:vars]
ansible_user=cis321
ansible_password=Password1
ansible_connection=winrm
ansible_winrm_server_cert_validation=ignore
```

2. Modify the hosts file to add a **complex1** group, which should point to your Ubuntu server.

3. Write a Playbook to count processes on the machine specified in the hosts file and on the Ansible server.

4. Open a shell, and type the following input **remembering to replace "alex" with the correct user ID for your system (Note**: The command is **wc** "**dash lower-case L**" and not a negative numerical 1):

```
vim count_processes.yml
i
---
- hosts: complex1, localhost
  remote_user: alex
  tasks:
    - name: count processes running on the remote system
      shell: ps | wc -l
      register: remote_processes_number
    - name: Print remote running process
      debug:
```

```
        msg: '{{ remote_processes_number.stdout }}'
    - name: Count processes running on the local system
      local_action: shell ps | wc -l
      register: local_processes_number
    - name: Print local running processes
      debug:
        msg: '{{ local_processes_number.stdout }}'
<ESC>
:wq
```

```
---
- hosts: complex1, localhost
#  remote_user: alex
  tasks:
    - name: count processes running on the remote system
      shell: ps | wc -l
      register: remote_processes_number
    - name: Print remote running process
      debug:
        msg: '{{ remote_processes_number.stdout }}'
    - name: Count processes running on the local system
      local_action: shell ps | wc -l
      register: local_processes_number
    - name: Print local running processes
      debug:
        msg: '{{ local_processes_number.stdout }}'
```

Take a screenshot that resembles the one above, and paste it in your Lab Report.

Check the Syntax, and Run the Playbook

Note that the IP addresses in your output will reflect your network architecture and may differ from what is shown below.

In a shell type the following:

```
ansible-playbook count_processes.yml --syntax-check
```

Press Enter. Then run:

```
ansible-playbook count_processes.yml
```

```
[alex@localhost ansible]$ ansible-playbook count_processes.yml

PLAY [complex1] ****************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.0.249]

TASK [count processes running on the remote system] ****************************************************************
changed: [192.168.0.249]

TASK [Print remote running process] ********************************************************************************
ok: [192.168.0.249] => {
    "msg": "6"
}

TASK [Count processes running on the local system] *****************************************************************
changed: [192.168.0.249]

TASK [Print local running processes] *******************************************************************************
ok: [192.168.0.249] => {
    "msg": "9"
}

PLAY RECAP *********************************************************************************************************
192.168.0.249              : ok=5    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Take a screenshot that resembles the one above, and paste it in your Lab Report.

Write a Playbook to Install or Update vim on the Linux Systems

Write a Playbook to install or update vim on the Linux systems, and download Notepad++ to the Windows server using an operating system conditional clause.

1. Edit the hosts file to create a group called **complex4** containing the Ubuntu and Windows systems. (**NOTE: Your output may reflect a failure to install for the Red Hat family of systems.**)

2. Create a playbook to install vim on the Debian and Red Hat-based systems and download the Notepad installer to the Windows server. You may have to edit the URL for the Notepad++ installer to reflect the most recent version. It also may be necessary to create the destination directory before running your playbook. **Remember**, it is advisable to perform a syntax check on your playbook before executing it.

   In a shell type:

```
---
- hosts: complex4
  remote_user: alex
  tasks:
    - name: Print the ansible_os_family value
      debug:
        msg: '{{ ansible_os_family }}'
    - name: Ensure vim package is updated on RedHat servers
```

```
        yum:
          name: vim
          state: latest
        become: True
        when: ansible_os_family == 'RedHat'
      - name: Ensure the vim package is updated on Debian servers
        apt:
          name: vim
          state: latest
        become: True
        when: ansible_os_family == 'Debian'
      - name: Download Notepad++
        win_get_url:
          url: https://github.com/notepad-plus-plus/notepad-plus-
plus/releases/download/v7.9.3/npp.7.9.3.Installer.exe
          dest: C:\ansible1
        when: ansible_os_family == 'Windows'
```

NOTE: Please remember that, depending upon your vim configuration, the spaces may show up as periods or as spaces. (Of course, the periods in the url always appear as periods.)

```
1 ---
2 -.hosts:.complex4
3 ..remote_user:.alex
4 ..tasks:
5 ....-.name:.Print.the.ansible_os_family.value
6 ......debug:
7 ........msg:.'{{.ansible_os_family.}}'
8 ....-.name:.Ensure.vim.package.is.updated.on.RedHat.servers
9 ......yum:
10 ........name:.vim
11 ........state:.latest
12 ......become:.True
13 ......when:.ansible_os_family.==.'RedHat'
14 ....-.name:.Ensure.the.vim.package.is.updated.on.Debian.servers
15 ......apt:
16 ........name:.vim
17 ........state:.latest
18 ......become:.True
19 ......when:.ansible_os_family.==.'Debian'
20 ....-.name:.Download.Notepad++
21 ......win_get_url:
22 ........url:.https://github.com/notepad-plus-plus/notepad-plus-plus/releases/download/v7.9.1/npp.7.9.1.Installer.x64.exe
23 ........dest:.C:\ansible1
24 ......when:.ansible_os_family.==.'Windows'
```

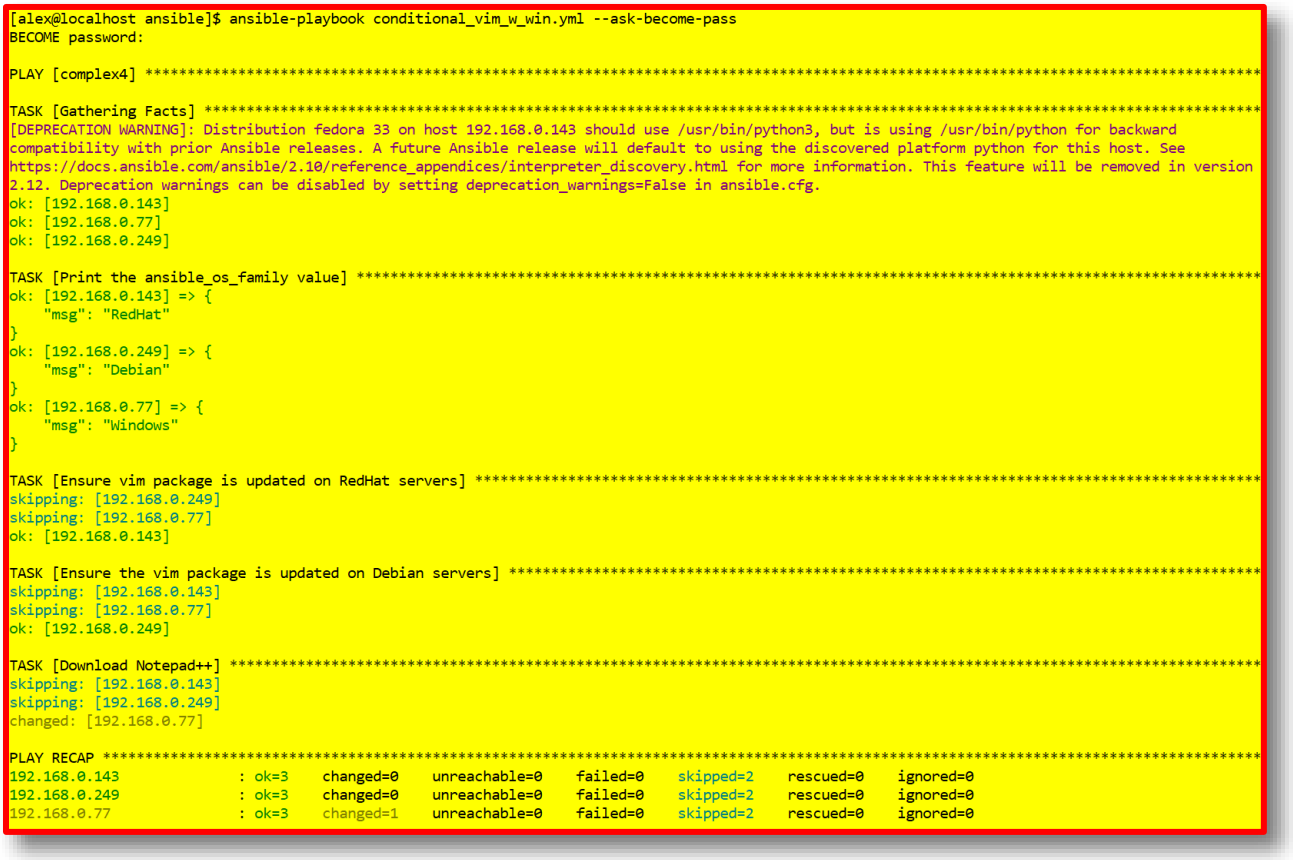
Take a screenshot that resembles the one above, and paste it in your Lab Report.

In a shell, type:

```
ansible-playbook conditional_vim_w_win.yml --syntax-check
```

```
ansible-playbook conditional_vim_w_win.yml --ask-become-pass
```

Your results may vary based on what has already been installed, and you may disregard any deprecation warning, but there should not be any errors.



Take a screenshot that resembles the one above, and paste it in your Lab Report.

Write a Playbook to Install Apache on Windows

NOTE: Make sure that your hosts file contains a [win] group that points to your Windows server.

In a shell, type:

```
vim install_apache_win.yml
i
---
- name: Install Apache
  hosts: win

  tasks:
```

```
    - name: Create Directory
      win_shell: mkdir c:\ansible_examples
      args:
        executable: cmd.exe

    - name: Download the Apache installer
      win_get_url:
        url: https://archive.apache.org/dist/httpd/binaries/win32/httpd-
2.2.25-win32-x86-no_ssl.msi
        dest: C:\ansible_examples\httpd-2.2.25-win32-x86-no_ssl.msi

    - name: Install MSI
      win_package:
        path: C:\ansible_examples\httpd-2.2.25-win32-x86-no_ssl.msi
        state: present
 <ESC> :wq
```

NOTE: Please remember that, depending upon your vim configuration, the spaces may show up as periods or as spaces. (Of course, the periods in the url always appear as periods.)

```
1 ---
2 -.name:.Install.Apache
3 ..hosts:.win
4
5 ..tasks:
6
7 ....-.name:.Create.Directory
8 ......win_shell:.mkdir.c:\ansible_examples
9 ......args:
10 ........executable:.cmd.exe
11
12 ....-.name:.Download.the.Apache.installer
13 ......win_get_url:
14 ........url:.https://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.25-win32-x86-no_ssl.msi
15 ........dest:.C:\ansible_examples\httpd-2.2.25-win32-x86-no_ssl.msi
16
17 ....-.name:.Install.MSI
18 ......win_package:
19 ........path:.C:\ansible_examples\httpd-2.2.25-win32-x86-no_ssl.msi
20 ........state:.present
```

In a shell, type the following. (**NOTE**: If you run this playbook twice, you may receive an error because the C:\ansible_examples directory would have been created already. If that is the case, simply delete the directory.)

```
ansible-playbook install_apache_win.yml --syntax-check
```

```
ansible-playbook install_apache_win.yml
```

```
playbook: install_apache_win.yml
[alex@localhost ansible]$ ansible-playbook install_apache_win.yml

PLAY [Install Apache] *********************************************************************************

TASK [Gathering Facts] ********************************************************************************
ok: [192.168.0.77]

TASK [Download the Apache installer] ******************************************************************
ok: [192.168.0.77]

TASK [Install MSI] ************************************************************************************
changed: [192.168.0.77]

PLAY RECAP ********************************************************************************************
192.168.0.77               : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Take a screenshot that resembles the one above, and paste it in your Lab Report.

Log in to the Windows machine, open a command prompt, navigate to the **C:\Program Files (x86)\Apache Software Foundation\Apache2.2\bin** directory, and type:

```
httpd -v <ENTER>
```

```
c:\Program Files (x86)\Apache Software Foundation\Apache2.2\bin>httpd -v
Server version: Apache/2.2.25 (Win32)
Server built:   Jul 10 2013 01:52:12
```

Take a screenshot that resembles the one above, and paste it in your Lab Report.

Guided Practice Questions

In your **Guided Practice Lab Report**, answer the following questions about this learning activity. Some may require research.

1. Break down and explain the output of the `shell ps | wc -l` command.
2. Does it make sense to create a separate group for the local_action exercise, or would it be easier to use the IP address? Discuss.
3. What does the command '`{{ local_processes_number.stdout }}`' generate?
4. What procedures and or policies would you enact to manage your inventory file?

References

GeekFlare. (2020). *9 Ansible playbooks example for Windows administration.* https://geekflare.com/ansible-playbook-windows-example/

Henderson, B. (2019). Connecting to a Windows host. https://www.ansible.com/blog/connecting-to-a-windows-host

McKay, D. (2019). *How to use pipes on Linux*. https://www.howtogeek.com/438882/how-to-use-pipes-on-linux/

Hochstein, L. & Moser, R. (2017) *Ansible: Up and running.* [Kindle].

JavaTpoint. (2018). *Ansible commands cheat sheets*. https://www.javatpoint.com/ansible-commands-cheat-sheets

Kenlon, S. (2019). *10 YAML tips for people who hate YAML*. https://www.redhat.com/sysadmin/yaml-tips

LinuxBuz. (2020). *Ansible apt module – tutorial and examples.* https://linuxbuz.com/linuxhowto/ansible-apt-module

NGEL. (2020). *YAML Script: Second ansible playbook to get date and server uptime status*. https://ngelinux.com/yaml-script-first-ansible-playbook-to-get-date-and-server-uptime-status/

OS Radar. (2020). *How to enable SSH in Windows Server 2019*. https://www.osradar.com/how-to-enable-ssh-in-windows-server-2019/

RedHat. (2020c). Glossary. https://docs.ansible.com/ansible/latest/reference_appendices/glossary.html

Kelom. (2020). Ansible - MYSQL installation. https://medium.com/@kelom.x/ansible-mysql-installation-2513d0f70faf

https://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.25-win32-x86-no_ssl.msi