Lab: 7 Managing Facts

Introduction:

Ansible collects pretty much all the information about the remote hosts as it runs a playbook. The task of collecting this remote system information is called as Gathering Facts by ansible and the details collected are generally known as facts or variables

Login as **root** with password as **linux**:

1. Let us create the ./data-facts directory set it as current directory as data-facts.

```
# cd
# mkdir data-facts
# cd data-facts
```

1.1 Create a fact file named. **/data-facts /custom.fact**. The fact file defines the package to install and service to start on **ansi-node1**.

```
# cat > custom.fact <<EOF
 [general]
 package = httpd
 service = httpd
 state = started
 enabled = true
EOF
```

1.2 Let us download the **setup_facts.yml** playbook to make the **/etc/ansible/facts.d** remote directory and to save the custom.fact file to that directory.

```
# wget
https://raw.githubusercontent.com/EyesOnCloud/ansible/main/setup_fac
ts.yaml
```

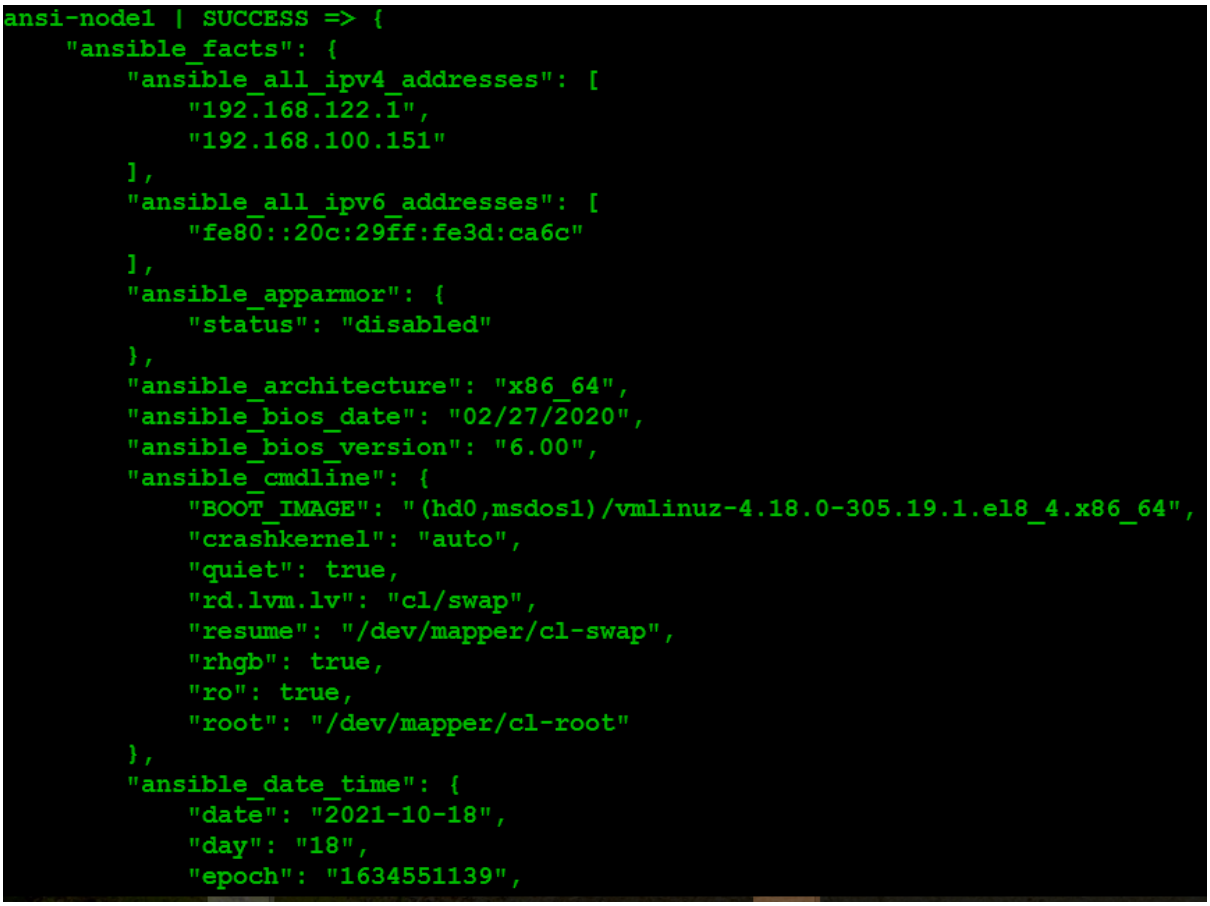
Output:

```
[root@ansi-master data-facts]# wget https://raw.githubusercontent.com/EyesOnCloud/ansible/main/setup_facts.yaml
--2021-12-11 15:23:33--  https://raw.githubusercontent.com/EyesOnCloud/ansible/main/setup_facts.yaml
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.110.133, 185.199.111.133, 185.199.108.133, ..
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 390 [text/plain]
Saving to: 'setup_facts.yaml'

setup_facts.yaml                    100%[=====================================================================>]     390

2021-12-11 15:23:33 (27.9 MB/s) - 'setup_facts.yaml' saved [390/390]
```

1.3 Run an ad hoc command with the setup module. Search for the ansible_local section in the output. There should not be any custom facts at this point.

```
$ ansible ansi-node1 -m setup
```

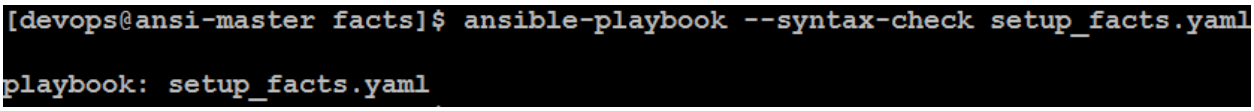
Output:



1.4 Let us verify its syntax by using below command

```
$ ansible-playbook --syntax-check setup_facts.yaml
```

Output:



1.5 Let us run the setup_facts.yml playbook.

```
$ ansible-playbook setup_facts.yaml
```

Output:

1.6 It is now possible to create the main playbook that uses both default and user facts to configure **ansi-node1.** Create the playbook **fact.yml**.

1.7 Editing the **fact.yml** file by creating the first task that installs the httpd package. Use the user fact for the name of the package.

1.8 Create another task that uses the custom fact to start the httpd service. Review the playbook and ensure all the tasks are defined

```
# wget
https://raw.githubusercontent.com/EyesOnCloud/ansible/main/fac
t.yaml
```

Output:

Let view the manifest

```
# cat -n fact.yaml
```

Output:

```
[root@ansi-master data-facts]# cat -n fact.yaml
     1  ---
     2  - name: Install Apache and starts the service
     3    hosts: ansi-node1
     4    become: yes
     5
     6    tasks:
     7      - name: Install the required package
     8        dnf:
     9          name: "{{ ansible_facts['ansible_local']['custom']['general']['package'] }}"
    10          state: latest
    11
    12      - name: Start the service
    13        service:
    14           name: "{{ ansible_facts['ansible_local']['custom']['general']['service'] }}"
    15           state: "{{ ansible_facts['ansible_local']['custom']['general']['state'] }}"
    16           enabled: "{{ ansible_facts['ansible_local']['custom']['general']['enabled'] }}"
```

1.9 Verify the syntax of the playbook by running ansible-playbook –syntax-check and if it reports any errors

```
# ansible-playbook --syntax-check fact.yaml
```

Output:

```
[root@ansi-master data-facts]# ansible-playbook --syntax-check fact.yml

playbook: fact.yml
[root@ansi-master data-facts]# ansible-playbook fact.yml
```

1.10 Run the playbook using the ansible-playbook command. Watch the output as ansible installs the package and then enables the service.

```
# ansible-playbook fact.yaml
```

Output:

```
[devops@ansi-master facts]$ ansible-playbook fact.yml

PLAY [Install Apache and starts the service] ***********************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [ansi-node1]

TASK [Install the required package] ********************************************************************
ok: [ansi-node1]

TASK [Start the service] *******************************************************************************
ok: [ansi-node1]

PLAY RECAP *********************************************************************************************
ansi-node1                 : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

1.11 Use an ad hoc command to execute systemctl to determine whether the httpd service is now running on **ansi-node1.**

```
$ ansible ansi-node1 -m command -a 'systemctl status httpd'
```

Output:

```
[root@ansi-master data-facts]# ansible ansi-node1 -m command -a 'systemctl status httpd'
ansi-node1 | CHANGED | rc=0 >>
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2021-10-20 15:48:47 IST; 19h ago
     Docs: man:httpd.service(8)
 Main PID: 4180 (httpd)
   Status: "Total requests: 14; Idle/Busy workers 100/0;Requests/sec: 0.000203; Bytes served/sec:   2 B/sec"
    Tasks: 213 (limit: 49185)
   Memory: 37.8M
   CGroup: /system.slice/httpd.service
           ├─4180 /usr/sbin/httpd -DFOREGROUND
           ├─4181 /usr/sbin/httpd -DFOREGROUND
           ├─4182 /usr/sbin/httpd -DFOREGROUND
           ├─4183 /usr/sbin/httpd -DFOREGROUND
           └─4184 /usr/sbin/httpd -DFOREGROUND

Oct 20 15:48:47 ansi-node1 systemd[1]: Starting The Apache HTTP Server...
Oct 20 15:48:47 ansi-node1 systemd[1]: Started The Apache HTTP Server.
Oct 20 15:48:47 ansi-node1 httpd[4180]: Server configured, listening on: port 80
```