

Name: Ballesteros, John Erwin S.	Date Performed: 9/30/2024
Course/Section: CpE31s2	Date Submitted: 10/2/2024
Instructor: Engr. Robin Valenzuela	Semester and SY: 1st Sem, '24 - '25
Activity 6: Targeting Specific Nodes and Managing Services	
<p>1. Objectives:</p> <ul style="list-style-type: none"> 1.1 Individualize hosts 1.2 Apply tags in selecting plays to run 1.3 Managing Services from remote servers using playbooks 	
<p>2. Discussion:</p> <p>In this activity, we try to individualize hosts. For example, we don't want apache on all our servers, or maybe only one of our servers is a web server, or maybe we have different servers like database or file servers running different things on different categories of servers and that is what we are going to take a look at in this activity.</p> <p>We also try to manage services that do not automatically run using the automations in playbook. For example, when we install web servers or httpd for CentOS, we notice that the service did not start automatically.</p> <p>Requirement:</p> <p>In this activity, you will need to create another Ubuntu VM and name it Server 3. Likewise, you need to activate the second adapter to a host-only adapter after the installations. Take note of the IP address of the Server 3. Make sure to use the command <i>ssh-copy-id</i> to copy the public key to Server 3. Verify if you can successfully SSH to Server 3.</p>	
Task 1: Targeting Specific Nodes	
<ul style="list-style-type: none"> 1. Create a new playbook and named it site.yml. Follow the commands as shown in the image below. Make sure to save the file and exit. 	

```

---
- hosts: all
  become: true
  tasks:

    - name: install apache and php for Ubuntu servers
      apt:
        name:
          - apache2
          - libapache2-mod-php
        state: latest
        update_cache: yes
        when: ansible_distribution == "Ubuntu"

    - name: install apache and php for CentOS servers
      dnf:
        name:
          - httpd
          - php
        state: latest
        when: ansible_distribution == "CentOS"

```

```

- hosts: all
  become: true
  tasks:

    - name: install apache and php for Ubuntu Servers
      apt:
        name:
          - apache 2
          - libapache2-mod-php
        state: latest
        update_cache: tes
        when: ansible_distribution == "Ubuntu"

    - name: install apache and php for centos servers
      dnf:
        name:
          - apache2
          - libapache2-mod-php
        when: ansible_distribution == "CentOS"

```

2. Edit the inventory file. Remove the variables we put in our last activity and group according to the image shown below:

```
[web_servers]
192.168.56.120
192.168.56.121

[db_servers]
192.168.56.122

[file_servers]
192.168.56.123
```

Make sure to save the file and exit.

```
all:
  children:
    web_servers:
      hosts:
        192.168.56.109
    db_servers:
      hosts:
        192.168.56.110
    file_servers:
      hosts:
        192.168.56.113
  vars:
    ansible_user: erwin
    ansible_ssh_private_key_file: /home/erwin/.ssh/id_rsa
```

Right now, we have created groups in our inventory file and put each server in its own group. In other cases, you can have a server be a member of multiple groups, for example you have a test server that is also a web server.

3. Edit the *site.yml* by following the image below:

```

---
- hosts: all
  become: true
  pre_tasks:
    - name: install updates (CentOS)
      dnf:
        update_only: yes
        update_cache: yes
        when: ansible_distribution == "CentOS"

    - name: install updates (Ubuntu)
      apt:
        upgrade: dist
        update_cache: yes
        when: ansible_distribution == "Ubuntu"

- hosts: web_servers
  become: true
  tasks:
    - name: install apache and php for Ubuntu servers
      apt:
        name:
          - apache2
          - libapache2-mod-php
        state: latest
        when: ansible_distribution == "Ubuntu"

    - name: install apache and php for CentOS servers
      dnf:
        name:
          - httpd
          - php
        state: latest
        when: ansible_distribution == "CentOS"

```

Make sure to save the file and exit.

The *pre-tasks* command tells the ansible to run it before any other thing. In the *pre-tasks*, CentOS will install updates while Ubuntu will upgrade its distribution package. This will run before running the second play, which is targeted at *web_servers*. In the second play, apache and php will be installed on both Ubuntu servers and CentOS servers.

Run the *site.yml* file and describe the result.

```

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.110]
ok: [192.168.56.113]

TASK [install updates Ubuntu] *****
skipping: [192.168.56.113]
ok: [192.168.56.110]
ok: [192.168.56.109]

TASK [install updates Centos] *****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
ok: [192.168.56.113]

PLAY [web_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]

TASK [install apache and php for Ubuntu Servers] *****
changed: [192.168.56.109]

TASK [install apache and php for CentOS Servers] *****
skipping: [192.168.56.109]

PLAY RECAP *****
192.168.56.109      : ok=4    changed=1    unreachable=0    failed=0    skipped=2
192.168.56.110      : ok=2    changed=0    unreachable=0    failed=0    skipped=1
192.168.56.113      : ok=2    changed=0    unreachable=0    failed=0    skipped=1

```

With the pre-tasks the playbook always check for the updates for both ubuntu and centos before proceeding to the regular playbook of updating the web servers.

- Let's try to edit again the *site.yml* file. This time, we are going to add plays targeting the other servers. This time we target the *db_servers* by adding it on the current *site.yml*. Below is an example: (Note add this at the end of the playbooks from task 1.3.

```

- hosts: db_servers
  become: true
  tasks:

    - name: install mariadb package (CentOS)
      yum:
        name: mariadb-server
        state: latest
      when: ansible_distribution == "CentOS"

    - name: "Mariadb- Restarting/Enabling"
      service:
        name: mariadb
        state: restarted
        enabled: true

    - name: install mariadb package (Ubuntu)
      apt:
        name: mariadb-server
        state: latest
      when: ansible_distribution == "Ubuntu"

```

Make sure to save the file and exit.

Run the *site.yml* file and describe the result.

```

- hosts: db_servers
  become: true
  tasks:

    - name: install mariadb package centos
      apt:
        name: mariadb-server
        state: latest
      when: ansible_distribution == "Ubuntu"

    - name: install apache and php for CentOS Servers
      yum:
        name: mariadb-server
        state: latest
      when: ansible_distribution == "CentOS"

```

```
PLAY [db_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.110]

TASK [install mariadb package centos] *****
ok: [192.168.56.110]

TASK [install apache and php for CentOS Servers] *****
skipping: [192.168.56.110]

PLAY RECAP *****
192.168.56.109      : ok=4    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.110     : ok=4    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.113     : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

When running the playbook it also runs the previous command before reaching to the db_server where it installs the mariadb to the machine.

5. Go to the remote server (Ubuntu) terminal that belongs to the db_servers group and check the status for mariadb installation using the command: *systemctl status mariadb*. Do this on the CentOS server also.

Describe the output.

```
erwin@Server2:~$ systemctl status mariadb
● mariadb.service - MariaDB 10.11.8 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset:
   Active: active (running) since Wed 2024-10-02 07:58:38 PST; 23min ago
     Docs: man:mariabdb(8)
           https://mariadb.com/kb/en/library/systemd/
   Process: 1282 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d /va
   Process: 1300 ExecStartPre=/bin/sh -c systemctl unset-environment _WSREP_S
   Process: 1308 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] &&
   Process: 1629 ExecStartPost=/bin/sh -c systemctl unset-environment _WSREP_
   Process: 1631 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0
   Main PID: 1460 (mariabdb)
    Status: "Taking your SQL requests now..."
     Tasks: 10 (limit: 76601)
  Memory: 108.9M (peak: 112.2M)
     CPU: 1.517s
    CGroup: /system.slice/mariadb.service
            └─1460 /usr/sbin/mariabdb

Oct 02 07:58:38 Server2 mariabdb[1460]: 2024-10-02  7:58:38 0 [Note] InnoDB: L
Oct 02 07:58:38 Server2 mariabdb[1460]: 2024-10-02  7:58:38 0 [Note] Plugin 'F
Oct 02 07:58:38 Server2 mariabdb[1460]: 2024-10-02  7:58:38 0 [Note] InnoDB: B
Oct 02 07:58:38 Server2 mariabdb[1460]: 2024-10-02  7:58:38 0 [Warning] You ne
Oct 02 07:58:38 Server2 mariabdb[1460]: 2024-10-02  7:58:38 0 [Note] Server sc
```

```
[erwin@centos9 ~]$ systemctl status mariadb
Unit mariadb.service could not be found.
[erwin@centos9 ~]$
```

On the Ubuntu machine the mariadb installed because it is the one that is inserted on the inventory list of servers, meanwhile the CentOS machine is assigned in other servers. Thus not finding a mariadb service.

6. Edit the *site.yml* again. This time we will append the code to configure installation on the *file_servers* group. We can add the following on our file.

```
- hosts: file_servers
  become: true
  tasks:

    - name: install samba package
      package:
        name: samba
        state: latest
```

Make sure to save the file and exit.

```
- hosts: file_servers
  become: true
  tasks:

    - name: install samba package
      package:
        name: samba
        state: latest
```

Run the *site.yml* file and describe the result.

```
PLAY [file_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.113]

TASK [install samba package] *****
changed: [192.168.56.113]

PLAY RECAP *****
192.168.56.109      : ok=4    changed=0    unreachable=0    failed=0    skipped=2
192.168.56.110     : ok=4    changed=0    unreachable=0    failed=0    skipped=2
192.168.56.113     : ok=4    changed=1    unreachable=0    failed=0    skipped=1
erwin@workstation: /Act6-Collectors$
```

The samba package is installed after all the previous server playbooks have been performed.

The testing of the *file_servers* is beyond the scope of this activity, and as well as our topics and objectives. However, in this activity we were able to show that we can target hosts or servers using grouping in ansible playbooks.

Task 2: Using Tags in running playbooks

In this task, our goal is to add metadata to our plays so that we can only run the plays that we want to run, and not all the plays in our playbook.

1. Edit the `site.yml` file. Add tags to the playbook. After the name, we can place the tags: `name_of_tag`. This is an arbitrary command, which means you can use any name for a tag.

```
---  
  
- hosts: all  
  become: true  
  pre_tasks:  
  
    - name: install updates (CentOS)  
      tags: always  
      dnf:  
        update_only: yes  
        update_cache: yes  
        when: ansible_distribution == "CentOS"  
  
    - name: install updates (Ubuntu)  
      tags: always  
      apt:  
        upgrade: dist  
        update_cache: yes  
        when: ansible_distribution == "Ubuntu"
```

```
- hosts: web_servers
  become: true
  tasks:

    - name: install apache and php for Ubuntu servers
      tags: apache,apache2,ubuntu
      apt:
        name:
          - apache2
          - libapache2-mod-php
        state: latest
      when: ansible_distribution == "Ubuntu"

    - name: install apache and php for CentOS servers
      tags: apache,centos,httpd
      dnf:
        name:
          - httpd
          - php
        state: latest
      when: ansible_distribution == "CentOS"
```

```

- hosts: db_servers
  become: true
  tasks:

    - name: install mariadb package (CentOS)
      tags: centos, db, mariadb
      dnf:
        name: mariadb-server
        state: latest
        when: ansible_distribution == "CentOS"

    - name: "Mariadb- Restarting/Enabling"
      service:
        name: mariadb
        state: restarted
        enabled: true

    - name: install mariadb package (Ubuntu)
      tags: db, mariadb, ubuntu
      apt:
        name: mariadb-server
        state: latest
        when: ansible_distribution == "Ubuntu"

- hosts: file_servers
  become: true
  tasks:

    - name: install samba package
      tags: samba
      package:
        name: samba
        state: latest

```

Make sure to save the file and exit.

```

- name: install mariadb package ubuntu
  tags: db, mariadb, ubuntu
  apt:

```

```

- name: install apache and php for Ubuntu Servers
  tags: apache,apache2,ubuntu
  apt:
    name:
      - apache2
      - libapache2-mod-php
    state: latest
  when: ansible_distribution == "Ubuntu"

- name: install apache and php for CentOS Servers
  tags: apache,centos,httpd
  dnf:

```

Run the *site.yml* file and describe the result.

```

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.110]
ok: [192.168.56.113]

TASK [install updates Ubuntu] *****
skipping: [192.168.56.113]
ok: [192.168.56.110]
ok: [192.168.56.109]

TASK [install updates Centos] *****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
ok: [192.168.56.113]

PLAY [web_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]

TASK [install apache and php for Ubuntu Servers] *****
ok: [192.168.56.109]

TASK [install apache and php for CentOS Servers] *****
skipping: [192.168.56.109]

```

When the playbook is performed it runs as usual without regard for the tags that we have implemented.

2. On the local machine, try to issue the following commands and describe each result:

2.1 *ansible-playbook --list-tags site.yml*

```
erwin@workstation:~/Act6-Ballesteros$ ansible-playbook --list-tags site.yml

playbook: site.yml

play #1 (all): all    TAGS: []
TASK TAGS: [always]

play #2 (web_servers): web_servers    TAGS: []
TASK TAGS: [apache, apache2, centos, httpd, ubuntu]

play #3 (db_servers): db_servers    TAGS: []
TASK TAGS: [centos, db, db. mariadb, mariadb, ubuntu]

play #4 (file_servers): file_servers TAGS: []
TASK TAGS: [samba]
```

Using the `--list-tags` command allows us to see what are the tags on each play number and the different tags that we have put on the tasks that is created.

2.2 *ansible-playbook --tags centos --ask-become-pass site.yml*

```
TASK [install apache and php for CentOS Servers] *****
skipping: [192.168.56.109]

PLAY [db_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.110]

TASK [install mariadb for CentOS Servers] *****
skipping: [192.168.56.110]

PLAY [file_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.113]

PLAY RECAP *****
192.168.56.109      : ok=3    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.110     : ok=3    changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.113     : ok=3    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

Specifying using the `--tags` and putting which tag you would like to run. On the output it skipped all the plays without the centos tag.

2.3 *ansible-playbook --tags db --ask-become-pass site.yml*

```
PLAY [web_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.109]

PLAY [db_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.110]

TASK [install mariadb package ubuntu] *****
ok: [192.168.56.110]

PLAY [file_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.113]

PLAY RECAP *****
192.168.56.109      : ok=3    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.110      : ok=4    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.113      : ok=3    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

On the results after running with only the db tags, it only performed the db_servers play while the other is skipped.

2.4 *ansible-playbook --tags apache --ask-become-pass site.yml*

```
TASK [install updates Centos] *****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
ok: [192.168.56.113]

PLAY [web_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.109]

TASK [install apache and php for Ubuntu Servers] *****
ok: [192.168.56.109]

TASK [install apache and php for CentOS Servers] *****
skipping: [192.168.56.109]

PLAY [db_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.110]

PLAY [file_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.113]
```

Using the apache and viewing the results, the only playbook that has been performed is the web_servers play and both centos and ubuntu tasks has been performed as well

2.5 *ansible-playbook --tags "apache,db" --ask-become-pass site.yml*

```
PLAY [web_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.109]

TASK [install apache and php for Ubuntu Servers] *****
ok: [192.168.56.109]

TASK [install apache and php for CentOS Servers] *****
skipping: [192.168.56.109]

PLAY [db_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.110]

TASK [install mariadb package ubuntu] *****
ok: [192.168.56.110]

PLAY [file_servers] *****
TASK [Gathering Facts] *****
ok: [192.168.56.113]

PLAY RECAP *****
192.168.56.109      : ok=4    changed=0    unrea
192.168.56.110      : ok=4    changed=0    unrea
192.168.56.113      : ok=3    changed=0    unrea
```

By using quotation marks (“ ”) on the - - tags we are able to insert multiple tags at a time, when looking at the play recap. Tags with apache and db has been performed.

Task 3: Managing Services

1. Edit the file site.yml and add a play that will automatically start the httpd on CentOS server.

```
- name: install apache and php for CentOS servers
  tags: apache,centos,httpd
  dnf:
    name:
      - httpd
      - php
    state: latest
  when: ansible_distribution == "CentOS"

- name: start httpd (CentOS)
  tags: apache, centos,httpd
  service:
    name: httpd
    state: started
  when: ansible_distribution == "CentOS"
```

Figure 3.1.1

Make sure to save the file and exit.

You would also notice from our previous activity that we already created a module that runs a service.

```
- hosts: db_servers
  become: true
  tasks:

  - name: install mariadb package (CentOS)
    tags: centos, db,mariadb
    dnf:
      name: mariadb-server
      state: latest
    when: ansible_distribution == "CentOS"

  - name: "Mariadb- Restarting/Enabling"
    service:
      name: mariadb
      state: restarted
      enabled: true
```

Figure 3.1.2


```
- name: start httpd (centos)
  tags: apache, centos, httpd
  service:
    name: httpd
    state: started
  when: ansible_distribution == "CentOS"

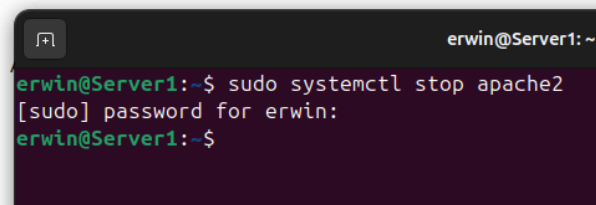
- name: start apache2 (ubuntu)
  tags: apache, apache2, ubuntu
  service:
    name: apache2
    state: started
```

This is because in CentOS, installed packages' services are not run automatically. Thus, we need to create the module to run it automatically.

2. To test it, before you run the saved playbook, go to the CentOS server and stop the currently running httpd using the command `sudo systemctl stop httpd`. When prompted, enter the sudo password. After that, open the browser and enter the CentOS server's IP address. You should not be getting a display because we stopped the httpd service already.

🔒 https://192.168.56.109

Unable to connect



```
erwin@Server1: ~
erwin@Server1:~$ sudo systemctl stop apache2
[sudo] password for erwin:
erwin@Server1:~$
```

3. Go to the local machine and this time, run the `site.yml` file. Then after running the file, go again to the CentOS server and enter its IP address on the browser. Describe the result.
To automatically enable the service every time we run the playbook, use the command `enabled: true` similar to Figure 7.1.2 and save the playbook.

```
erwin@workstation:~/Act6-Ballesteros$ ansible-playbook --tags apache
BECOME password:

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]
ok: [192.168.56.113]
ok: [192.168.56.110]

TASK [install updates Ubuntu] *****
skipping: [192.168.56.113]
ok: [192.168.56.109]
ok: [192.168.56.110]

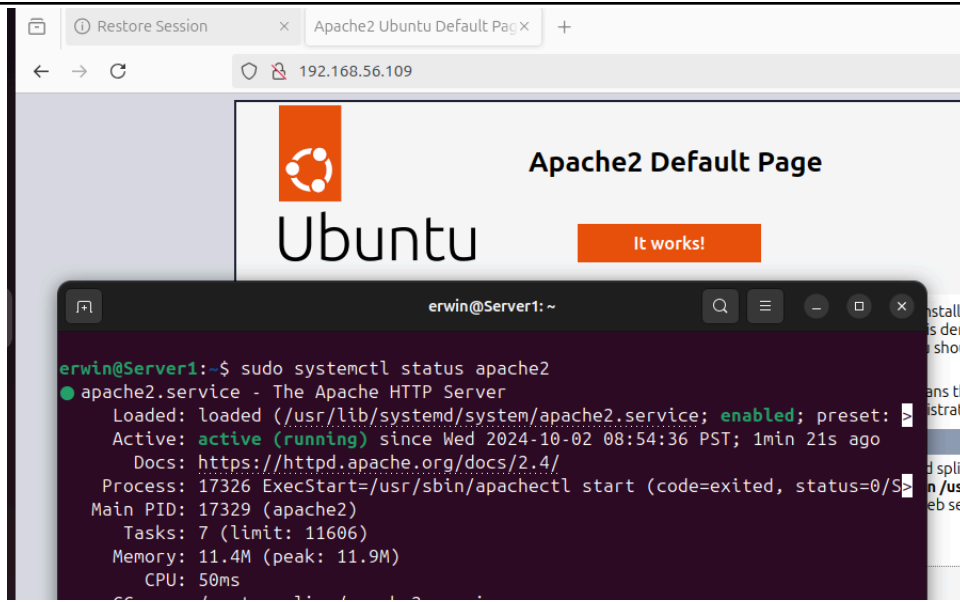
TASK [install updates Centos] *****
skipping: [192.168.56.109]
skipping: [192.168.56.110]
ok: [192.168.56.113]

PLAY [web_servers] *****

TASK [Gathering Facts] *****
ok: [192.168.56.109]

TASK [install apache and php for Ubuntu Servers] *****
ok: [192.168.56.109]

TASK [start apache2 (ubuntu)] *****
changed: [192.168.56.109]
```



By specifying apache tags and running the playbook we have seen that there are changes on the task of starting apache 2. When checking the status of the apache2 on the CLI of the server machine it has been performed successfully.

Reflections:

Answer the following:

1. What is the importance of putting our remote servers into groups?

The importance of managing our servers into groups is to be able to segregate them properly and avoid confusion especially in a professional setting where some physical setups are scattered

2. What is the importance of tags in playbooks?

The importance of tags in playbook so that we are able to identify them easily especially in larger scale playbooks where things are connected we need a way to specify them without making it tedious by creating multiple tasks for multiple systems

3. Why do you think some services need to be managed automatically in playbooks?

Services such as mariadb need constant checking to be managed constantly to avoid errors and standardization so that it will be the same when it is played across the server. This also saves time as it does not need a person to do this.

Github Repo: github.com/Moznaim/Act6-Ballesteros

