| | |
|---|---|
| **Name:** John Renzo L. Mendoza | **Date Performed:** October 11, 2023 |
| **Course/Section:** CPE 31 S5 | **Date Submitted:** October 14, 2023 |
| **Instructor:** Engr. Roman Richard | **Semester and SY:** 1st Semester, 2023 - 2024 |

**Activity 6: Targeting Specific Nodes and Managing Services**

1. **Objectives:**

1.1 Individualize hosts
1.2 Apply tags in selecting plays to run
1.3 Managing Services from remote servers using playbooks

2. **Discussion**:

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.

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.

**Requirement:**

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 installation. Take note of the IP address of Server 3. Make sure to use the command *ssh-copy-id* to copy the public key to Server 3. Verify if you can successfully SSH to Server 3.

## Task 1: Targeting Specific Nodes

1. Create a new playbook and name 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"
```

Observation:

- The playbook configuration uses the when command to determine whether to perform the task which depends on the variable ansible_distribution. The package manager for CentOS was changed from dnf to yum, because dnf does not work on my end, and yum is also a Red Hat package manager.

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.

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.

Observation:

- The playbook will initially run all of the tasks on all remote servers defined on the inventory file which is “hosts”. Then, the second part only targets the hosts that are considered as web_servers.

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.

Observation:

- I have disabled one of the web_servers which is Server 2 on my end. This is because my personal computer is already lagging on handling 5 virtual machines at the same time. To achieve the experiment objectives, I have left one server for each type so that I will only run 4 virtual machines at the same time, the control node included.

Observation:

- The playbook has successfully performed targeted plays based on the remote server's group name. On the second play, the playbook only performed the commands on the web_servers which is for Server 1 and 2, but in this case, it is only Server 2 since I have disabled Server 1 due to actual computer resources issue.

4. Let's try to edit 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 packege (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.

Observation:

- The playbook has additional play which targets the remote server's under db_servers group. This is specifically the CentOS server. The first and second plays are similar to what was on the previous step. The third play however, targeted the db_servers which is the ip for the CentOS. Neglecting the other remote servers' ip that aren't part of the db_servers group.

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.

```
jhnrnzmendoza@workstation:~/CPE232_MendozaHome$ ssh jhnrnzmendoza@192.168.56.101
Last login: Thu Oct 12 11:12:23 2023
[jhnrnzmendoza@CentOS ~]$ systemctl status mariadb
● mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-10-12 11:12:13 EDT; 1min 44s ago
  Process: 1366 ExecStartPost=/usr/libexec/mariadb-wait-ready $MAINPID (code=exited, status=0/SUCCESS)
  Process: 1204 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n (code=exited, status=0/SUCCESS)
 Main PID: 1365 (mysqld_safe)
    Tasks: 20
   CGroup: /system.slice/mariadb.service
           ├─1365 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
           └─1549 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql --plugin-dir=/usr/lib64/mysql/plugin --log-error=/var/log/mariadb/mariadb.l
og --pid-file=/var/run/mari...

Oct 12 11:12:08 CentOS systemd[1]: Starting MariaDB database server...
Oct 12 11:12:09 CentOS mariadb-prepare-db-dir[1204]: Database MariaDB is probably initialized in /var/lib/mysql already, nothing is done.
Oct 12 11:12:09 CentOS mariadb-prepare-db-dir[1204]: If this is not the case, make sure the /var/lib/mysql is empty before running mariadb-prepare-db-dir
.
Oct 12 11:12:09 CentOS mysqld_safe[1365]: 231012 11:12:09 mysqld_safe Logging to '/var/log/mariadb/mariadb.log'.
Oct 12 11:12:09 CentOS mysqld_safe[1365]: 231012 11:12:09 mysqld_safe Starting mysqld daemon with databases from /var/lib/mysql
Oct 12 11:12:13 CentOS systemd[1]: Started MariaDB database server.
```

```
CPE31S5 CentOs_Mendoza [Running] - Oracle VM VirtualBox
File  Machine  View  Input  Devices  Help
Applications  Places  Terminal                                   Thu 11:13
jhnrnzmendoza@CentOS:~
File  Edit  View  Search  Terminal  Help
[jhnrnzmendoza@CentOS ~]$ systemctl status mariadb
● mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-10-12 11:12:13 EDT; 34s ago
  Process: 1366 ExecStartPost=/usr/libexec/mariadb-wait-ready $MAINPID (code=exited, status=0/SUCCESS)
  Process: 1204 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n (code=exited, status=0/SUCCESS)
 Main PID: 1365 (mysqld_safe)
    Tasks: 20
   CGroup: /system.slice/mariadb.service
           ├─1365 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
           └─1549 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql --plug...

Oct 12 11:12:08 CentOS systemd[1]: Starting MariaDB database server...
Oct 12 11:12:09 CentOS mariadb-prepare-db-dir[1204]: Database MariaDB is probably i....
Oct 12 11:12:09 CentOS mariadb-prepare-db-dir[1204]: If this is not the case, make ....
Oct 12 11:12:09 CentOS mysqld_safe[1365]: 231012 11:12:09 mysqld_safe Logging to '...'.
Oct 12 11:12:09 CentOS mysqld_safe[1365]: 231012 11:12:09 mysqld_safe Starting mys...ql
Oct 12 11:12:13 CentOS systemd[1]: Started MariaDB database server.
Hint: Some lines were ellipsized, use -l to show in full.
[jhnrnzmendoza@CentOS ~]$
```

Describe the output.

Observation:

- Using both ssh and checking the server under db_servers group which is CentOs, I was able to observe that the mariadb service is now active on the CentOs server. This was possible since this server is under the db_servers group in which the third play of the playbook targets. Therefore, the other servers not included in the db_servers group do not have mariadb service, an example on Server 1 is shown below.

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.

```
jhnrnzmendoza@workstation:~/CPE232_MendozaHome$ ansible-playbook --ask-become-pass site.yml
BECOME password:

PLAY [all] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]
ok: [192.168.56.106]
ok: [192.168.56.101]

TASK [install updates (CentOS)] ************************************************
skipping: [192.168.56.108]
skipping: [192.168.56.106]
ok: [192.168.56.101]

TASK [install updates (Ubuntu)] ************************************************
skipping: [192.168.56.101]
changed: [192.168.56.108]
changed: [192.168.56.106]

PLAY [web_servers] *************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.108]

TASK [install apache and php for Ubuntu servers] *******************************
ok: [192.168.56.108]

TASK [install apache and php for CentOS servers] *******************************
skipping: [192.168.56.108]

PLAY [db_servers] **************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.101]

TASK [install mariadb package (CentOS)] ****************************************
ok: [192.168.56.101]

TASK [Mariadb- Restarting/Enabling] ********************************************
changed: [192.168.56.101]

TASK [install mariadb package (Ubuntu)] ****************************************
skipping: [192.168.56.101]

PLAY [file_servers] ************************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.56.106]

TASK [install samba package] ***************************************************
changed: [192.168.56.106]

PLAY RECAP *********************************************************************
192.168.56.101             : ok=5    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.106             : ok=4    changed=2    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.108             : ok=4    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

Observation:

- Similarly, the added fourth play only targeted the group file_servers which includes out Server 3. Among all of the added plays on the play book (4 plays, including pre-task), we can observe that the playbook can only target a group in which it will not execute on the servers or managed nodes that aren't included.

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"
```

```yaml
- 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"
```

```yaml
- 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 packege (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
```

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

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

- hosts: file_servers
  become: true
  tags: samba
  tasks:

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

Make sure to save the file and exit.
Run the *site.yml* file and describe the result.

```
TASK [install mariadb package (Ubuntu)] ************************************************************
skipping: [192.168.56.101]

PLAY [file_servers] ********************************************************************************

TASK [Gathering Facts] *****************************************************************************
ok: [192.168.56.106]

TASK [install samba package] ***********************************************************************
ok: [192.168.56.106]

PLAY RECAP *****************************************************************************************
192.168.56.101             : ok=5    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
192.168.56.106             : ok=4    changed=1    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.108             : ok=4    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

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

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

```
jhnrnzmendoza@workstation:~/CPE232_MendozaHome$ 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, mariadb, ubuntu]

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

Observation:

- This command shows the tags that have been added to each of the plays in the playbook yml file. We can observe that the tags are arbitrary and can be changed according to the user.

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

```
jhnrnzmendoza@workstation:~/CPE232_MendozaHome$ ansible-playbook --tags centos --ask-become-pass site.yml
BECOME password:

PLAY [all] *********************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.108]
ok: [192.168.56.106]
ok: [192.168.56.101]

TASK [install updates (CentOS)] ************************************************************************************
skipping: [192.168.56.108]
skipping: [192.168.56.106]
ok: [192.168.56.101]

TASK [install updates (Ubuntu)] ************************************************************************************
skipping: [192.168.56.101]
changed: [192.168.56.108]
changed: [192.168.56.106]

PLAY [web_servers] *************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.108]

TASK [install apache and php for CentOS servers] *******************************************************************
skipping: [192.168.56.108]

PLAY [db_servers] **************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [192.168.56.101]

TASK [install mariadb package (CentOS)] ****************************************************************************
ok: [192.168.56.101]

PLAY [file_servers] ************************************************************************************************

PLAY RECAP *********************************************************************************************************
192.168.56.101             : ok=4    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.106             : ok=2    changed=1    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
192.168.56.108             : ok=3    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
```

Observation:

- Since we have added a parameter or an option to the ansible-playbook command which is –centos, the playbook only executed the plays with the “centos” tag. Specifically, play 1, 2, and 3. The 4th play was skipped since it does not have a tag of “centos”. On the contrary, play 1, has executed since the tag is “always” and the target hosts are “all”.

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

Observation:

- Similarly, this command targeted specific plays which only had the tag. In this case, the tag was “db”, therefore, the only play that was run was play’s 1 and 3. Play number 1 is always run regardless of the tag, while play number 3 has the tag “db”.

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

Observation:

- Similarly, this command targeted specific plays which only had the tag. In this case, the tag was “apache”, therefore, the only play that was run was play’s 1, and 2. Play number 1 is always run regardless of the tag, while play number 2 has the tag “apache”.

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

Observation:

- Similarly, this command targeted specific plays which only had the tag. In this case, the tag was “apache” and “db”, therefore, the only play that was run was play’s 1, 2, and 3. Play number 1 is always run regardless of the tag, while play number 2 has the tag “apache”, and finally play number 3 has the tag “db”.

**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

```
CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox
Activities   Terminal                                   Oct 13 20:42
jhnrnzmendoza@workstation: ~/CPE232_MendozaHome
GNU nano 6.2                                            site.yml *
      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
  tags: samba
```

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.

Observation:

- After running the given command, the Apache HTTP server is now inactive. This was observed using the "systemctl status httpd" command.

Observation:

- Upon checking in the web browser, I have inputted the centOs virtual machine's IP address to the address bar. The browser displayed no connectivity as it was inactive or administratively shutdown.

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.

CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox
jhnrnzmendoza@workstation: ~/CPE232_MendozaHome
PLAY [db_servers] **********
TASK [Gathering Facts] **********
ok: [192.168.56.101]
TASK [install mariadb package (CentOS)] **********
ok: [192.168.56.101]
TASK [Mariadb- Restarting/Enabling] **********
changed: [192.168.56.101]
TASK [install mariadb package (Ubuntu)] **********
skipping: [192.168.56.101]
PLAY [file_servers] **********
TASK [Gathering Facts] **********
ok: [192.168.56.106]
TASK [install samba package] **********
ok: [192.168.56.106]
PLAY RECAP **********
192.168.56.101 : ok=8 changed=2 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0
192.168.56.106 : ok=4 changed=1 unreachable=0 failed=0 skipped=1 rescued=0 ignored=0
192.168.56.108 : ok=4 changed=1 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0

CPE31S5 Workstation_Mendoza [Running] - Oracle VM VirtualBox
jhnrnzmendoza@workstation: ~/CPE232_MendozaHome
GNU nano 6.2 hosts
[web_servers]
#192.168.56.107
192.168.56.108
192.168.56.101
[db_servers]
192.168.56.101
[file_servers]
192.168.56.106
[ Read 39 lines ]
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo

Observation:

- On this playbook run, I have not added a tag so that it would run all of the plays included in the playbook. The task that would enable the httpd service was included on the 2nd play which is for web_servers. to achieve this output, I have added the IP of the centOS to the web_servers group, so that we would observe the effects of having the added service task to the managed node.
- This is necessary because the playbook task will only run if the ansible_distribution is CentOS. Therefore, the web_servers group should have a CentOS managed node in order to test the effects of the commands or tasks.

Observation:

- After running the playbook, I have returned to the CentOS managed node and checked the web browser by typing its IP address. The playbook run was successful and was able to enable the httpd service on the remote node.

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.

**Reflections:**

Answer the following:

1. What is the importance of putting our remote servers into groups?
   - Based on what was done in the activity, the playbook site.yml has a lot of tasks to execute and not all of those tasks can be run on all of the remote managed nodes since they vary in their purpose and in their distribution. Putting these remote servers into groups would classify them according to use, and perform playbook tasks on them in a targeted manner. Moreover, the playbook specifically pinpoints which remote servers would execute the tasks. This will lower the cpu utilization as it does not need to execute on the servers in which we know it does not need it. It provides a more efficient and more automated way of running the playbook, and can be helpful when we get onto more remote servers / managed nodes.

2. What is the importance of tags in playbooks?
   - In this activity, tags are introduced similar to the keywords of a playbook play. Basically, we assign tags that are related to the process or purpose of the play in the playbook. Using a proper syntax on the ansible-playbook command, the system administrator would be able to use the defined tags in order to select which plays from the playbook will be run. Compared to having groups, tags makes the execution of the playbook shorter, in a way that it would not execute the tasks under a play that does not correspond on the passed argument together with the ansible-playbook command. This makes the execution efficient, time saving, and shorter.

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

- Services are the ones that run in a computer in the background and are responsible for responding to different requests. In this activity, we have observed the service httpd on our CentOS server. CentOS uses this service in order to access the web browser and reply to the clients by forming a network. This service should be enabled automatically on servers since it needs an uptime to respond to client's requests, if any. In addition, we also observed the mariaDB services on our db_servers, this service is primarily responsible for database structure and configuration. This service should be also active on database servers as this is needed to perform database commands in order to modify them.