

**Universidad ORT Uruguay**

**Facultad de Ingeniería**

**Bernard Wand Polak**



# **Obligatorio**

# **Taller de Servidores Linux**

**Agosto 2023**

**Vladimir Demari N° - 291649**

**Sebastián Rodríguez N° - 166526**

**Docente: Enrique Verdes**

## Contenido

Introducción del Proyecto.....	3
Particionado de disco.....	4
Configuración de servidores .....	5
Playbook en Ansible .....	8
Roles Ansible Galaxy .....	10
geerlingguy.apache.....	10
container_tomcat .....	10
mariadb_verdes .....	11
create.database .....	11
Files en los roles.....	11

## Introducción del Proyecto

Este proyecto consta de la creación de tres servidores.

El primero llamado Bastion, cuya IP es 192.168.0.10 y con sistema operativo Rocky server, será nuestra interfaz grafica desde donde administraremos los otros dos servidores.

En el implementaremos el formato Workstation, para tener interfaz gráfica.

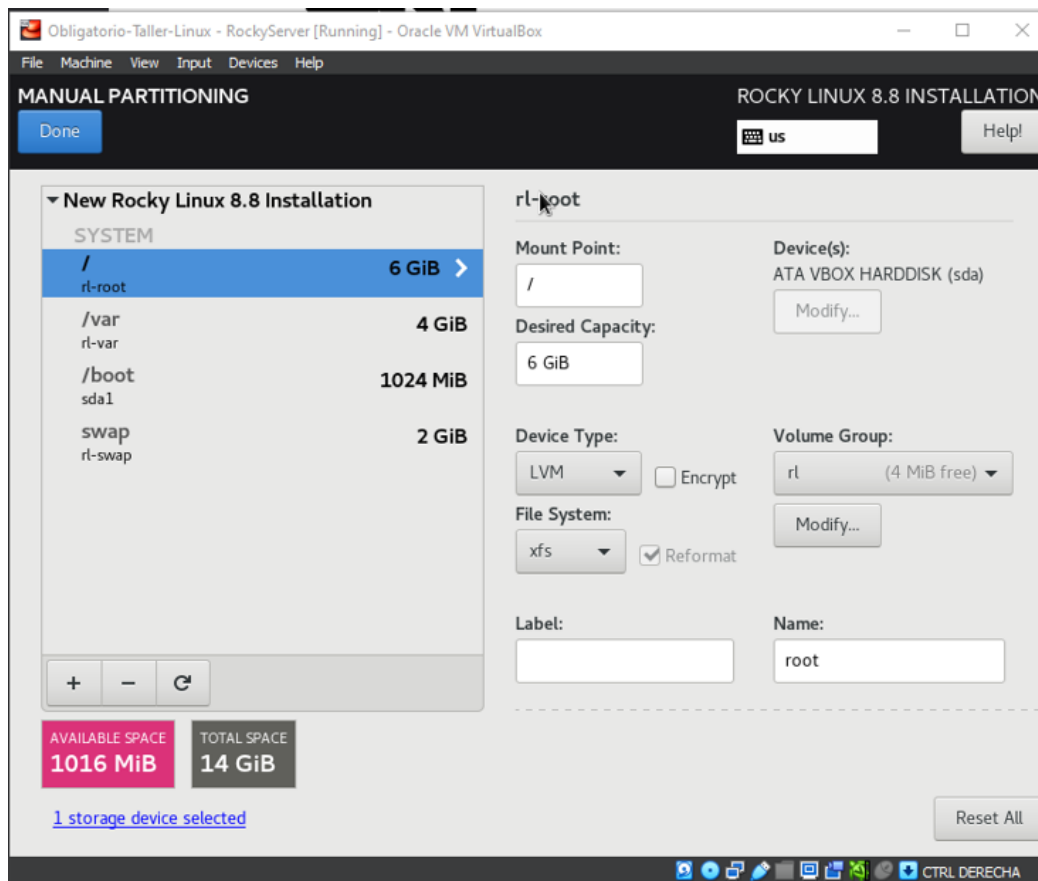
A su vez crearemos un servidor llamado RockyServer, cuya IP es 192.168.0.11, también con Linux Rocky server. Para este caso utilizaremos el formato “minimal” para reducir funcionalidad al mínimo, instalando nosotros los aplicativos necesario. El que sea “minimal” nos ayuda a reducir el acceso a personas que no tengan conocimientos de administración por línea de comandos.

Instalaremos un servidor con la distribución de Linux Ubuntu, que se llamara UbuntuServer, cuya IP es 192.168.0.12, en formato “minimal” con el mismo objetivo que para nuestro Rocky server. Reducir los servicios al mínimo necesario, y acotar las posibilidades de acceso.

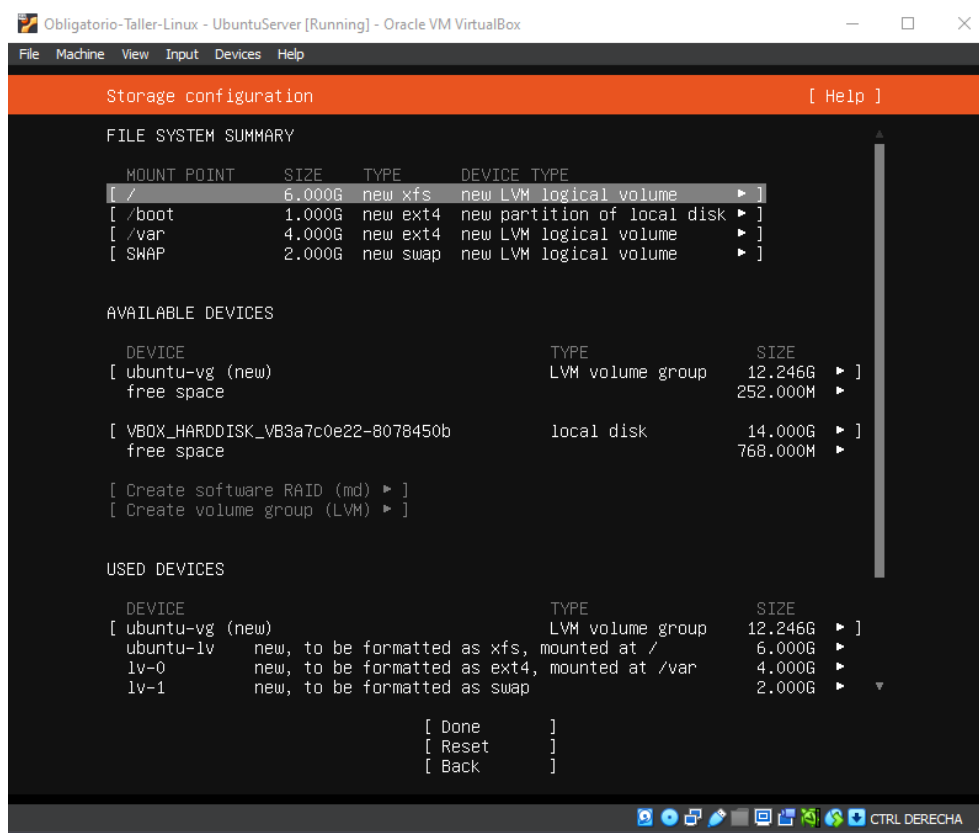
## Particionado de disco

El proyecto indica que los servidores deben cumplir con las siguientes características de particionado de disco:

Rocky Server



## Ubuntu Server



## Configuración de servidores

Generamos las claves en el Bastion las cuales utilizaremos para acceder a los servers y al repositorio en github.

```
[sysadmin@bastion ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/sysadmin/.ssh/id_rsa):
Created directory '/home/sysadmin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sysadmin/.ssh/id_rsa.
Your public key has been saved in /home/sysadmin/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:wZRpjWMJDyP7aH1h+SofwcbvG005o6yW6V/vSVcgLI sysadmin@bastion.ejemplo.com.uy
The key's randomart image is:
+---[RSA 3072]-----+
|
|..0+
|+++ +
|Oo....
|.B..o
|+SOE.
|.oo0..
|.o.B.o
|o.o0o.o
|o=0*++
+---[SHA256]-----+
[sysadmin@bastion ~]$
```

Agregamos al usuario Ansible en los servidores Rockyserver y UbuntuServer, el cual utilizaremos para acceder remotamente con Ansible.

Agregaremos este usuarios al grupo “sudo” y le daremos permiso de ejecutar “sudo” sin contraseña.

#### En Rockyserver

```
[sysadmin@rockyserver ~]$ sudo adduser ansible
[sudo] password for sysadmin:
[sysadmin@rockyserver ~]$ sudo usermod -aG wheel ansible
[sysadmin@rockyserver ~]$ echo "ansible ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/ansible
ansible ALL=(ALL) NOPASSWD: ALL
[sysadmin@rockyserver ~]$
```

#### En UbuntuServer

```
sysadmin@ubuntuServer:~$ sudo adduser ansible
[sudo] password for sysadmin:
Adding user `ansible' ...
Adding new group `ansible' (1001) ...
Adding new user `ansible' (1001) with group `ansible' ...
Creating home directory `/home/ansible' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ansible
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
sysadmin@ubuntuServer:~$ sudo usermod -aG sudo ansible
sysadmin@ubuntuServer:~$ echo "ansible ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/ansible
ansible ALL=(ALL) NOPASSWD: ALL
```

Luego copiamos la llave pública desde el bastión hacia los servidores, para que el usuario ansible pueda conectarse sin utilizar contraseña.

```
[sysadmin@bastion ~]$ ssh-copy-id ansible@192.168.0.11
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ansible@192.168.0.11's password:

Number of key(s) added: 1

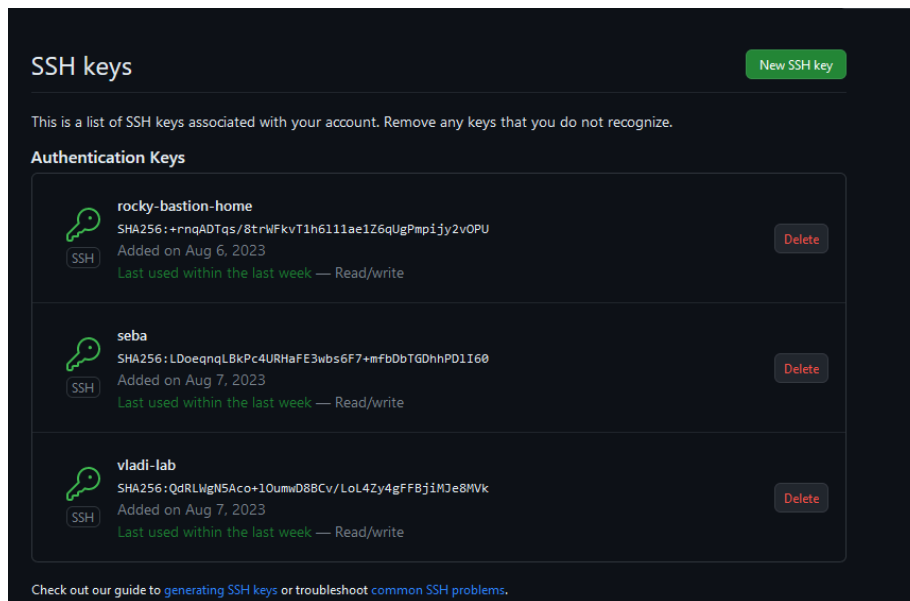
Now try logging into the machine, with:  "ssh 'ansible@192.168.0.11'"
and check to make sure that only the key(s) you wanted were added.
```

```
[sysadmin@bastion ~]$ ssh-copy-id ansible@192.168.0.12
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ansible@192.168.0.12's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'ansible@192.168.0.12'"
and check to make sure that only the key(s) you wanted were added.
```

## Keys en Github



Luego realizamos la instalación de Git y Ansible. En el caso de Git ya lo teníamos instalado en nuestro Rocky bastion.

```
[sysadmin@bastion ~]$ sudo dnf install git
[sudo] password for sysadmin:
Last metadata expiration check: 21:55:20 ago on Fri 04 Aug 2023 12:35:09 AM -03.
Package git-2.39.3-1.el8_8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

```
[sysadmin@bastion ~]$ sudo dnf install ansible
[sudo] password for sysadmin:
Last metadata expiration check: 11:42:37 ago on Fri 04 Aug 2023 11:16:42 PM -03.
Dependencies resolved.
=====
Package                                     Architecture
=====
Installing:
ansible                                     noarch
Installing dependencies:
ansible-core                               x86_64
mpdecimal                                  x86_64
python3-11                                  x86_64
```

Clonamos nuestro repositorio previamente creado en Github.

```
[sysadmin@bastion ~]$ git clone git@github.com:vladimirdemari/Obligatorio-ASLX.git
Cloning into 'Obligatorio-ASLX'...
```

Luego creamos los archivos de configuración para nuestro proyecto

- ansible.cfg

```
$ ansible-config init --disabled > ansible.cfg
```

```
# (pathlist) Comma separated list of Ansible inventory sources
inventory=/home/sysadmin/Obligatorio-ASLX/inventario
```

- inventario

```
[redhat]
rockyserver    ansible_host=192.168.0.11

[debian]
ubuntuserver   ansible_host=192.168.0.12

[linux:children]
redhat
debian
```

- requeriment

```
# Roles requeridos
- name: geerlingguy.apache
  src: https://github.com/geerlingguy/ansible-role-apache
```

Instalamos el este rol

```
[sysadmin@bastion Obligatorio-ASLX]$ vim requirements.yml
[sysadmin@bastion Obligatorio-ASLX]$ ansible-galaxy install -r requirements.yml -p roles/
Starting galaxy role install process
- extracting geerlingguy.apache to /home/sysadmin/Obligatorio-ASLX/roles/geerlingguy.apache
- geerlingguy.apache was installed successfully
```

## Playbook en Ansible

Decidimos crear una estructura de 3 bloques en nuestro main.yml principal. En cada estructura aplicaremos tareas a los diferentes Hosts definidos en el inventario.

Los bloques serán los siguientes:

- Linux: correrán tareas genéricas para todos los Linux
- RedHat: tareas para los servers Redhat
- Debian: tareas para los servers Debian



## Linux:

```
1  ---
2  - name: Tareas Generales en Servidores Linux
3    hosts: linux
4    become: yes
5    remote_user: ansible
6
7    tasks:
8      - name: Update RedHat Servers
9        yum:
10          name: "*"
11          state: latest
12          notify: Restart
13          when: ansible_os_family == "RedHat"
14
15      - name: Update Debian Servers
16        apt:
17          name: "*"
18          state: latest
19          update_cache: yes
20          notify: Restart
21          when: ansible_os_family == "Debian"
22
23    handlers:
24      - name: Restart
25        reboot:
```

## RedHat:

```
27  - name: Configuración de Servidores RedHat
28    hosts: redhat
29    become: yes
30    remote_user: ansible
31
32    roles:
33      - geerlingguy.apache
34      - container_tomcat
35
36    tasks:
37
38      - name: Open firewall ports
39        firewallld:
40          service: "{{ item }}"
41          state: enabled
42          permanent: true
43          immediate: true
44        loop:
45          - http
46          - https
```

## Debian:

```
47
48  - name: Configuración de Servidores Debian
49    hosts: debian
50    become: yes
51    remote_user: ansible
52
53    roles:
54      - mariadb_verdes
55      - create_database
```

## Roles Ansible Galaxy

En la estructura de carpetas contamos con 4 roles donde definiremos los roles y servicios que necesitamos instalar y configurar.

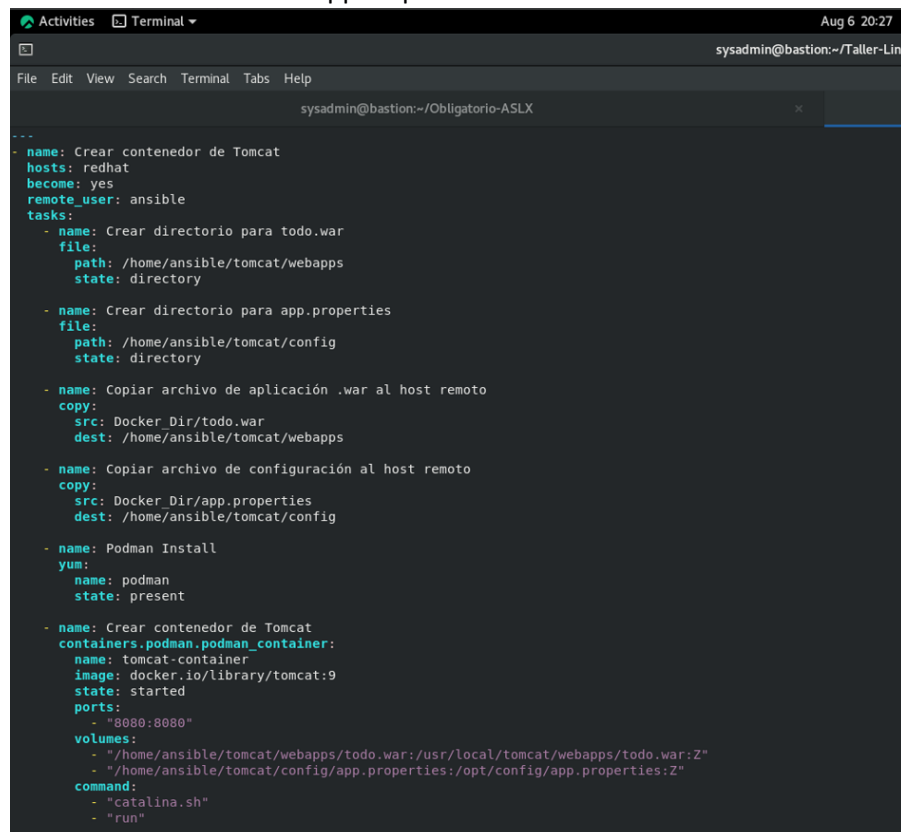
### geerlingguy.apache

Se trata de un rol creado por geerlingguy.apache el cual instala y realiza la configuración básica de un servidor apache. Modificamos una parte del código para incluir nuestra configuración de vhost para el reverse proxy.

### container tomcat

Se trata de un Rol creado por nosotros para implementar un contenedor tomcat con una la aplicación todo.war corriendo en el mismo.

Las tareas que realizan son, copiar archivos necesarios para nuestra app al server anfitrión, instalar podman en el servidor Redhat y correr el módulo “containers.podman.podman\_container” para la creación de la imagen del contenedor y la ejecución del mismo. Este modulo en lugar de copiar los archivos necesarios de la app mapea los mismos desde el anfitrión.



```
---
- name: Crear contenedor de Tomcat
  hosts: redhat
  become: yes
  remote_user: ansible
  tasks:
    - name: Crear directorio para todo.war
      file:
        path: /home/ansible/tomcat/webapps
        state: directory

    - name: Crear directorio para app.properties
      file:
        path: /home/ansible/tomcat/config
        state: directory

    - name: Copiar archivo de aplicación .war al host remoto
      copy:
        src: Docker_Dir/todo.war
        dest: /home/ansible/tomcat/webapps

    - name: Copiar archivo de configuración al host remoto
      copy:
        src: Docker_Dir/app.properties
        dest: /home/ansible/tomcat/config

    - name: Podman Install
      yum:
        name: podman
        state: present

    - name: Crear contenedor de Tomcat
      containers.podman.podman_container:
        name: tomcat-container
        image: docker.io/library/tomcat:9
        state: started
        ports:
          - "8080:8080"
        volumes:
          - "/home/ansible/tomcat/webapps/todo.war:/usr/local/tomcat/webapps/todo.war:Z"
          - "/home/ansible/tomcat/config/app.properties:/opt/config/app.properties:Z"
        command:
          - "catalina.sh"
          - "run"
```

## mariadb verdes

En este rol incluimos el playbook visto en clase, el cual instala Mariadb, Python3, realiza la configuración inicial de Mariadb y configuración de seguridad.

A su vez, realiza la configuración del Firewall de Ubuntu y abriendo el puerto necesario y harderiza la configuración del usuario root.

## create.database

Por un tema de orden decidimos crear este rol para incluir la configuración de nuestra base de datos. En las tareas de este rol creamos la base de datos y sus tablas, mediante un dump de la base de datos del obligatorio anterior. También creamos el usuario todo el cual utilizará la appweb para conectarse.

```
1  ---
2  # tasks file for roles/create.database
3
4  - name: Copiar archivo dump de la base de datos
5    copy:
6      src: roles/create.database/files/todo.sql
7      dest: /home/ansible/todo.sql
8
9
10 - name: Restore database
11   community.mysql.mysql_db:
12     login_host: localhost
13     login_user: root
14     login_password: root
15     name: todo
16     state: import
17     target: /home/ansible/todo.sql
18
19 - name: Creamos usuario todo
20   mysql_user:
21     login_host: localhost
22     login_user: root
23     login_password: root
24     name: todo
25     password: todo1234
26     # encrypted: yes
27     priv: ' *.*:ALL,GRANT'
28     state: present
```

## Files en los roles

Tanto en el rol container\_tomcat como en el rol create.database tienen archivos en sus respectivas carpetas /files.

En el caso de container\_tomcat tenemos el app.properties y el todo.war.

En el create.database tenemos el archivo todo.sql para el dump de la base de datos.