

Taller de servidores Linux



CentOS 8
The Community ENTERprise Operating System



Noc / Julio 2021

Rodrigo Santomauro N° 199089

Melissa Rossi N° 241893

Virginia Grajales N° 175840

ÍNDICE

Declaración de auditoría.....	2
Resumen ejecutivo.....	3
Introducción	4
Servidor de Centos.....	5
Servidor de Ubuntu	8
Servidor Bastión.....	10
GitHub.....	12
Ansible.....	14
Roles	15
Common.....	17
Rol DB.....	22
Web	27
Bibliografía	30

DECLARACIÓN DE AUDITORÍA

Nosotros, Rodrigo Santomauro, Virginia Grajales y Melissa Rossi, declaramos que el trabajo que se presenta en esta obra es de nuestra propia mano.

Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos la materia Taller de Servidores Linux, de la carrera Analista en Infraestructuras de la universidad ORT.
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hacemos acuse de recibo de las ayudas recibidas por parte de Sebastián Orrego, Roberto Wagner y Andrés Tarallo.
- Cuando la obra se basa en trabajo realizado juntamente con otros, hemos explicado claramente qué fue contribuido por otros.
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

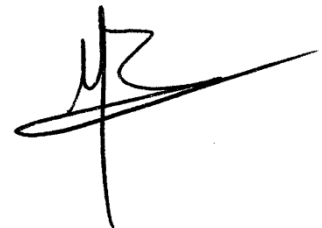
Martes 10 de agosto, 2021



Rodrigo Santomauro



Virginia Grajales



Melissa Rossi

RESUMEN EJECUTIVO

Este documento contendrá información técnica referente a la instalación y configuración de un stack LAMP. En esta instalación se utilizaron diferentes sistemas operativos basados en Linux.

La información será distribuida en diferentes apartados que abarcaran la instalaciones y configuraciones de cada uno de los componentes.

INTRODUCCIÓN

Primeramente, para cumplir con la finalidad del objetivo planteado, se implementan tres máquinas virtuales en VirtualBox con ciertas particularidades en las configuraciones de red, que serán detalladas posteriormente.










Con el objetivo de realizar una instalación de diferentes componentes se decide utilizar Ansible como aprovisionamiento del software necesario para la instalación del stack LAMP. Se debe tener en cuenta que las dos familias de distribuciones solicitadas son RedHat y Debian, donde estos serán creados con esquema de particionamientos de LVM.

La base de este proyecto surge de un repositorio público en GitHub brindado por los docentes de la materia donde en el sugieren una implementación del playbooks de Ansible para CentOS 7, donde se debe realizar un fork del mismo y realizar los cambios pertinentes para que su ejecución sea correcta sobre CentOS 8 y Debian.

Para realizar el mantenimiento del código ansible se utilizó la herramienta de Microsoft Visual Studio Code con las siguientes extensiones: remote-ssh, ansible de Tomasz Maciazek, ansible lenguaje de RedHat.

SERVIDOR DE CENTOS

Se crea una máquina virtual con el objetivo de aprovisionar el software desde ansible basada en la distribución CentOS 8 con las siguientes configuraciones.

	General
Nombre:	Oblig-CentOS test
Sistema operativo:	Red Hat (64-bit)
<hr/>	
	Sistema
Memoria base:	2048 MB
Procesadores:	2
Orden de arranque:	Óptica, Disco duro
Aceleración:	VT-x/AMD-V, Paginación anidada, PAE/NX, Paravirtualización KVM
<hr/>	
	Pantalla
Memoria de vídeo:	16 MB
Controlador gráfico:	VMSVGA
Servidor de escritorio remoto:	Inhabilitado
Grabación:	Inhabilitado
<hr/>	
	Almacenamiento
Controlador:	IDE
IDE secundario maestro:	[Unidad óptica] Vacío
Controlador:	SATA
Puerto SATA 0:	Oblig-CentOS test-disk1.vdi (Normal, 20.00 GB)
<hr/>	
	Audio
Controlador de anfitrión:	Windows DirectSound
Controlador:	ICH AC97
<hr/>	
	Red
Adaptador 1:	Intel PRO/1000 MT Desktop (Adaptador puente, «Intel(R) Dual Band Wireless-AC 3160»)
Adaptador 2:	Intel PRO/1000 MT Desktop (Adaptador solo anfitrión, «VirtualBox Host-Only Ethernet Adapter #4»)
<hr/>	
	USB
Controlador USB:	OHCI, EHCI
Filtros de dispositivos:	0 (0 activo)
<hr/>	
	Carpetas compartidas
Ninguno	
<hr/>	
	Descripción
Ninguno	

Donde podemos destacar que la misma contará con 2GB RAM para evitar falta de recursos, también se utilizó un disco incremental de 20GB.

También se debe crear un adaptador puente que será el enlace por el cual nos conectaremos al servidor, se opta por esta opción para facilitar la conexión evitando así la configuración de reenvío de puertos que se realizaría en caso de NAT

Una vez instalada nuestra máquina virtual, procedemos a configurarla con el siguiente esquema de particionamiento como nos pide en la letra del obligatorio

PARTICIONADO MANUAL
INSTALACIÓN DE CENTOS LIN

Hecho
latam
Ayl

New CentOS Linux 8 Installation

DATOS

/home 3 GiB >

SISTEMA

/ 5 GiB

cl_bastion-root

/var 4 GiB

cl_bastion-var

/boot 1024 MiB

sda1

swap 2 GiB

cl_bastion-swap

+ - ↺

ESPACIO DISPONIBLE
4,99 GiB

ESPACIO TOTAL
20 GiB

[1 dispositivo de almacenamiento seleccionado](#)

cl_bastion-home

Punto de montaje:

/home

Dispositivo(s):

ATA VBOX HARDDISK (sda)

Modificar...

Capacidad deseada:

3 GiB

Tipo de dispositivo:

LVM

☐ Cifrar

Grupo Del Volumen:

cl_bastion (4 MiB free)

Modificar...

Sistema de archivos:

xfs

☒ Reformatear

Etiqueta:

Nombre:

home

Restablecer to

Una vez realizada la instalación y la configuración del esquema de particiones lo mostraremos en pantalla.

```
lsblk & df -h
```

```
[root@bastion ~]# df -h
Filesystem                Size      Used Avail Use% Mounted on
devtmpfs                   890M         0   890M   0% /dev
tmpfs                      909M         0   909M   0% /dev/shm
tmpfs                      909M      8.5M   900M   1% /run
tmpfs                      909M         0   909M   0% /sys/fs/cgroup
/dev/mapper/cl_bastion-root 5.0G      1.6G   3.5G  32% /
/dev/sda1                  1014M     177M   838M  18% /boot
/dev/mapper/cl_bastion-home 3.0G       54M   3.0G   2% /home
/dev/mapper/cl_bastion-var  4.0G     240M   3.8G   6% /var
tmpfs                      182M         0   182M   0% /run/user/0

[root@bastion ~]# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                   8:0    0   20G  0 disk
├─sda1                 8:1    0    1G  0 part /boot
└─sda2                 8:2    0   14G  0 part
   ├─cl_bastion-root 253:0    0    5G  0 lvm  /
   ├─cl_bastion-swap 253:1    0    2G  0 lvm  [SWAP]
   ├─cl_bastion-var  253:2    0    4G  0 lvm  /var
   └─cl_bastion-home 253:3    0    3G  0 lvm  /home
sr0                   11:0    1 1024M  0 rom
```

6

Validamos la memoria RAM

```
free -g
```

```
[root@bastion ~]# free -g
              total        used         free       shared    buff/cache   available
Mem:           1           0           1           0           0           1
Swap:          1           0           1
```

Se creó un usuario ansible para poder realizar la conexión desde el Ansible que ejecutará las instrucciones sobre esta máquina.

Este usuario no debe poseer contraseña y debe tener los privilegios de super usuario.

```
## Allow root to run any commands anywhere
root    ALL=(ALL)        ALL
ansible ALL=(ALL) NOPASSWD:ALL
```

Se configura la red externa enp0s3 (192.168.0.71) y otra red interna enp0s8 (172.16.0.4)

```
[root@centos ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:74:1a:df brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.71/24 brd 192.168.0.255 scope global noprefixroute enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::b97:73aa:64d8:dd81/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:53:3a:04 brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.4/16 brd 172.16.255.255 scope global noprefixroute enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::ed0c:bc70:cb72:facc/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
[root@centos ~]#
```


SERVIDOR DE UBUNTU

Se crea una máquina virtual con el objetivo de aprovisionar el software desde ansible basada en la distribución Ubuntu 20.04 LTS con las siguientes configuraciones.



General

Nombre: Oblig-Ubuntu test
Sistema operativo: Ubuntu (64-bit)



Sistema

Memoria base: 2048 MB
Procesadores: 2
Orden de arranque: Óptica, Disco duro
Aceleración: VT-x/AMD-V, Paginación anidada, Paravirtualización KVM



Pantalla

Memoria de vídeo: 16 MB
Controlador gráfico: VMSVGA
Servidor de escritorio remoto: Inhabilitado
Grabación: Inhabilitado



Almacenamiento

Controlador: IDE
IDE secundario maestro: [Unidad óptica] Vacío
Controlador: SATA
Puerto SATA 0: Oblig-Ubuntu test-disk1.vdi (Normal, 20.00 GB)



Audio

Controlador de anfitrión: Windows DirectSound
Controlador: ICH AC97



Red

Adaptador 1: Intel PRO/1000 MT Desktop (Adaptador puente, «Intel(R) Dual Band Wireless-AC 3160»)
Adaptador 2: Intel PRO/1000 MT Desktop (Adaptador solo anfitrión, «VirtualBox Host-Only Ethernet Adapter #4»)



USB

Controlador USB: OHCI, EHCI
Filtros de dispositivos: 0 (0 activo)



Carpetas compartidas

Ninguno



Descripción

Ninguno

Se configura la red externa enp0s3 (192.168.0.70) y otra red interna enp0s8 (172.16.0.3)

```
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:14:fa:4a brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.70/24 brd 192.168.0.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe14:fa4a/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c8:d7:e0 brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.3/24 brd 172.16.0.255 scope global enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fec8:d7e0/64 scope link
        valid_lft forever preferred_lft forever
$
```

Se creó un usuario ansible para poder realizar la conexión desde el Ansible que ejecutará las instrucciones sobre esta máquina.

```
rvm@ubuntu:~$ sudo useradd ansible -m -c "Usuario de Ansible"
[sudo] password for rvm:
rvm@ubuntu:~$
```

Este usuario no debe poseer contraseña y debe tener los privilegios de super usuario.

```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults    env_reset
Defaults    mail_badpass
Defaults    secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
ansible ALL=(ALL) NOPASSWD:ALL

# Members of the admin group may gain root privileges
%admin   ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:
```

SERVIDOR BASTIÓN

Se crea una máquina virtual con CentOS 8 con el objetivo de centralizar las conexiones hacia los otros servidores, a la cual denominaremos bastión. En ella estarán instalados los paquetes de Ansible, también tendrá una copia del repositorio de GitHub del obligatorio brindado por los docentes y las claves públicas-privadas para realizar las conexiones mediante claves.

Creamos una clave pública para que se pueda conectar desde el bastión

```
ssh-keygen
```

```
[root@bastion ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:9K93u2ll9NvT3vXfrCPH4auhkScT0KuL5dQXjQymU9Y root@bastion.taller
The key's randomart image is:
+----[RSA 3072]-----+
|
|      . .
|     o * E
|    . B + o .
|   S + + ...
|  + + .. +
| + * =o +=
| = . Oo.B=B
| . o o..**BX|
+-----[SHA256]-----+
[root@bastion ~]#
```

Se realiza la copia de la clave pública hacia los servidores de CentOS 8 y Debian para realizar una autenticación instantánea. Por ejemplo, vemos el caso de CentOS

```
[root@bastion lamp]# ssh-copy-id ansible@172.16.0.4
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/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: WARNING: All keys were skipped because they already exist on the remote system.
(if you think this is a mistake, you may want to use -f option)
[root@bastion lamp]# ssh-copy-id ansible@172.16.0.4
```

Una vez copiada la clave, tratamos de conectarnos desde el bastión al servidor de Ubuntu

```
ssh ansible@172.16.0.3
```

```
[root@bastion ~]# ssh ansible@172.16.0.3
The authenticity of host '172.16.0.3 (172.16.0.3)' can't be established.
ECDSA key fingerprint is SHA256:azr6SwDdbbVpgGSBh/sHyBtB/ha3r5y8CS2FJyXJD3M.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.16.0.3' (ECDSA) to the list of known hosts.
}Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-65-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Aug  3 23:27:32 UTC 2021

System load:  0.3               Processes:            148
Usage of /home: 1.8% of 2.99GB   Users logged in:     1
Memory usage:  11%             IPv4 address for enp0s3: 172.16.0.3
Swap usage:    0%

0 updates can be installed immediately.
0 of these updates are security updates.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Tue Aug  3 23:25:41 2021 from 172.16.0.2
}$ ls
-sh: 1: }ls: not found
$ |
```

GITHUB

Se instala el paquete git en el bastión y se inicializa un directorio con el objetivo de almacenar una copia del repositorio anteriormente mencionado.

```
[root@bastion obligatorio]# git init
Initialized empty Git repository in /root/obligatorio/.git/
```

Creamos un clon de la bifurcación del repositorio

```
git clone https://github.com/rsantomauro/obligatorio_2021_08
```

```
[root@bastion obligatorio]# git clone https://github.com/rsantomauro/obligatorio_2021_08
Cloning into 'obligatorio_2021_08'...
remote: Enumerating objects: 35, done.
remote: Counting objects: 100% (35/35), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 35 (delta 0), reused 32 (delta 0), pack-reused 0
Unpacking objects: 100% (35/35), 5.54 KiB | 61.00 KiB/s, done.
[root@bastion obligatorio]#
```

Esto nos mostrara la siguiente estructura de carpetas.

```
[root@bastion obligatorio_2021_08]# pwd
/root/obligatorio/obligatorio_2021_08
[root@bastion obligatorio_2021_08]# ll
total 4
-rw-r--r--. 1 root root  21 Aug  3 19:36 README.md
drwxr-xr-x. 4 root root 101 Aug  3 19:36 lamp
[root@bastion obligatorio_2021_08]#
```

```
[root@bastion lamp]# ll
total 20
-rw-r--r--. 1 root root  237 Aug  3 19:36 LICENSE.md
-rw-r--r--. 1 root root 1163 Aug  3 19:36 README.md
-rw-r--r--. 1 root root   2 Aug  3 19:38 ansible.cfg
drwxr-xr-x. 2 root root  34 Aug  3 19:36 group_vars
-rw-r--r--. 1 root root  59 Aug  3 19:36 hosts
drwxr-xr-x. 5 root root  41 Aug  3 19:36 roles
-rw-r--r--. 1 root root  409 Aug  3 19:36 site.yml
[root@bastion lamp]#
```

Dentro del directorio de repositorio, se modifica la variable ntpserver dentro del archivo all ubicado dentro del directorio group_vars.

```
---  
# Variables listed here are applicable to all host groups  
  
httpd_port: 80  
ntpserver: 0.south-america.pool.ntp.org  
repository: https://github.com/bennojoy/mywebapp.git
```

Movimos el archivo hosts a un nuevo directorio llamado inventario, y modificamos este archivo con las IPs correspondientes de los servidores que destinarán tal funcionalidad.

```
[root@bastion lamp]# cat inventario/hosts  
[webservers]  
172.16.0.4  
  
[dbservers]  
172.16.0.3  
[root@bastion lamp]#
```

Se crea un archivo ansible.cfg en el cual se crean los parámetros básicos para el funcionamiento, como lo es el inventario, el usuario remoto para las ejecuciones, la variable de logs y el directorio el cual contendrá las claves privadas.

También dentro del directorio LAMP se agrega la opción become para otorgar los permisos de super usuario en la ejecución de cada rol.

```
roles:  
  - web  
  
- name: deploy MySQL and configure the databases  
  hosts: dbservers  
  remote_user: ansible  
  become: yes  
  
roles:  
  - db
```

ANSIBLE

Instalamos el lenguaje ansible en nuestras máquinas virtuales con los siguientes comandos

```
dnf install epel-release
dnf update
dnf install ansible
```

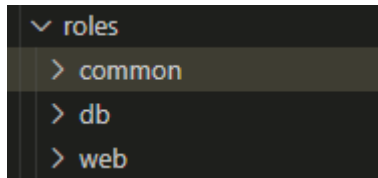
```
[root@bastion lamp]# ansible --version
ansible 2.9.23
  config file = /root/obligatorio/obligatorio_2021_08/lamp/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.6/site-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.6.8 (default, Aug 24 2020, 17:57:11) [GCC 8.3.1 20191121 (Red Hat 8.3.1-5)]
[root@bastion lamp]# ansible -i 172.16.0.3, all -m ping
172.16.0.3 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

```
[root@ubuntu ansible]# ll
total 0
[root@ubuntu ansible]# ll
total 0
[root@ubuntu ansible]# mkdir .ssh
[root@ubuntu ansible]# cd .ssh/
[root@ubuntu .ssh]# vim authorized_keys
[root@ubuntu .ssh]# chown -R ansible: /root/
```

```
[root@bastion lamp]# ansible -i 172.16.0.4, all -m ping
172.16.0.4 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/libexec/platform-python"
  },
  "changed": false,
  "ping": "pong"
}
[root@bastion lamp]# ansible -i 172.16.0.3, all -m ping
172.16.0.3 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
[root@bastion lamp]# |
```

ROLES

En la estructura de los roles como mostramos en la siguiente imagen, se separan diferentes instalaciones de diferentes servicios.



Se detallarán los contenidos modificados en cada uno de ellos en los siguientes apartados.

Se modifica el archivo site.yml para que utilice el usuario ansible

```
lamp > ! site.yml > {} 2 > [abc] remote_user
1  ---
2  # This playbook deploys the whole application stack in this site.
3
4  - name: apply common configuration to all nodes
5    hosts: all
6    remote_user: ansible
7
8    roles:
9      - common
10
11 - name: configure and deploy the webserver and application code
12   hosts: webserver
13   remote_user: ansible
14
15   roles:
16     - web
17
18 - name: deploy MySQL and configure the databases
19   hosts: dbserver
20   remote_user: ansible
21
22   roles:
23     - db
24
```


Se debe asignar los permisos, debido al [issue](#) 71200 de ansible, para esto se modificaron las siguientes líneas,

~/roles/common/tasks/main.yml

```
- name: Configure ntp file
  template:
    src: "ntp.conf.j2"
    dest: "/etc/ntp.conf"
    mode: 0600
  tags: ntp
  notify: restart ntp
```

~/roles/db/tasks/main.yml

```
- name: Create Mysql configuration file
  template:
    src: "my.cnf.j2"
    dest: "/etc/my.cnf"
    mode: 0600
  notify:
    - restart mariadb
```

~/roles/web/tasks/copy_code.yml

```
- name: Creates the index.php file
  template:
    src: "index.php.j2"
    dest: "/var/www/html/index.php"
    mode: 0600
```

COMMON

Debido a que el paquete ntp no está disponible en CentOS 8 y en las últimas versiones de Ubuntu, se debe realizar ciertas modificaciones al código de ansible del archivo task/main.yml. Se implantará Ansible facts para validar la familia de la distribución para poder elegir qué paquete utilizar.

```
# Se debe modificar el código porque ntp fue removido de la familia RedHat. Se agregan Facts
#- name: Install ntp
#  yum: name=ntp state=present
#  tags: ntp
```

Validamos los campos que queremos evaluar:
Para CentOS:

```
[root@bastion lamp]# ansible dbbservers -m ansible.builtin.setup | egrep 'ansible_distribution|ansible_distribu
tion_major_version'
    "ansible_distribution": "CentOS",
    "ansible_distribution_major_version": "8",
```

Quedando la configuración de Facts para la familia RedHat así

```
# Se debe modificar el código porque ntp fue removido de la version CentOS 8. Se agregan Facts
- name: Install chrony for Centos 8
  dnf:
    name: chrony
    state: present
  tags: chrony
  when: ansible_distribution == "CentOS" or
        ansible_distribution_major_version == "8"
```

Para los otros sistemas Debian también se modifica el paquete a chrony:

```
# Para versiones Debian instalacion de chrony
- name: Install Chrony on Debian
  apt: name=chrony state=present
  tags: chrony
  when: ansible_os_family == "Debian"
```

Se instalan las dependencias para las diferentes familias quedando el código de la siguiente manera:

```
# Se instalan las dependencias en CentOS 8
- name: Install common dependencies on CentOS 8
  dnf:
    name: "{{ dependencies }}"
    state: installed
  vars:
    dependencies:
      - python3-libselinux
      - python3-libsemanage
# El paquete firewalld ya esta instalado, quizas hay que usar este?
#   - python3-firewall
  when: ansible_distribution == "CentOS" or
        ansible_distribution_major_version == "8"

# Se instalan las dependencias en Debian
- name: Install common dependencies on Debian
  apt:
    name: "{{ dependencies }}"
    state: present
  vars:
    dependencies:
      - python3-selinux
      - python3-semanage
      - policycoreutils
      - selinux-utils
      - selinux-basics
# El paquete firewalld ya esta instalado, quizas hay que usar este?
#   - python3-firewall
  when: ansible_os_family == "Debian"
```

Se copia el archivo de configuración hacia los servidores

```
# Configura el archivo chrony
- name: Configure Chrony file
  template:
    src: "/root/obligatorio/obligatorio_2021_08/lamp/roles/common/templates/chrony.conf.j2"
    dest: "/etc/chrony.conf"
    mode: 0600
    tags: chrony
    notify: restart chrony
```

Para corresponder las variables de Jinja, se modifica la variable ntpserver, para que apunte al proyecto ntp.org de Uruguay <https://www.pool.ntp.org/zone/uy>

```
---
# Variables listed here are applicable to all host groups

httpd_port: 80
ntpserver: 0.south-america.pool.ntp.org
repository: https://github.com/bennojoy/mywebapp.git
```

Se crea un archivo de configuración para chronyd en el folder
~/roles/common/templates/ que contemple los parámetros aceptados

```
chrony.conf.j2 M X
lamp > roles > common > templates > chrony.conf.j2
1
2 driftfile /var/lib/chrony/drift
3
4 server 0.south-america.pool.ntp.org
5
6
```

Se agrega la inicialización del servicio de chronyd para CentOS 8

```
# Iniciar servicio de chronyd
- name: Start the chrony service for CentOS 8
  service:
    name: chronyd
    state: started
    enabled: yes
  tags: chrony
```

En Runtime del playbook common/task/main.yml despliega el siguiente error:

```
TASK [Install common dependencies] *****
[DEPRECATION WARNING]: Invoking "yum" only once while using a loop via squash_actions is deprecated. Instead of using a loop to supply multiple items and specifying `name: "{{ item }}"`, please use `name: ['libselinux-python', 'libsemanage-python', 'firewalld']` and remove the loop. This feature will be removed in version 2.11. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
[DEPRECATION WARNING]: Invoking "yum" only once while using a loop via squash_actions is deprecated. Instead of using a loop to supply multiple items and specifying `name: "{{ item }}"`, please use `name: ['libselinux-python', 'libsemanage-python', 'firewalld']` and remove the loop. This feature will be removed in version 2.11. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
failed: [172.16.0.3] (item=['libselinux-python', 'libsemanage-python', 'firewalld']) => {"ansible_facts": {"pkg_mgr": "apt"}, "ansible_loop_var": "item", "changed": false, "item": ["libselinux-python", "libsemanage-python", "firewalld"], "msg": "value of state must be one of: absent, build-dep, fixed, latest, present, got: installed"}
failed: [172.16.0.4] (item=['libselinux-python', 'libsemanage-python', 'firewalld']) => {"ansible_loop_var": "item", "changed": false, "failures": ["libselinux-python Todas las coincidencias se filtraron mediante un filtrado modular del argumento: libselinux-python", "libsemanage-python No hay coincidencias para el argumento: libsemanage-python"], "item": ["libselinux-python", "libsemanage-python", "firewalld"], "msg": "Failed to install some of the specified packages", "rc": 1, "results": []}
```

Se corrige el código deprecated ya que se encuentra obsoleto:

```
- name: Install common dependencies
  yum: name={{ item }} state=installed
  with_items:
    - libselinux-python
    - libsemanage-python
    - firewalld
```

Pasando el loop a variables como es mencionado en este artículo:

<https://www.programmersought.com/article/4888525503/>

Para CentOS 8

```
# Se instalan las dependencias en CentOS 8
- name: Install common dependencies on CentOS 8
  dnf:
    name: "{{dependencies}}"
  vars:
    dependencies:
      - python3-libselinux
      - python3-libsemanage
# El paquete firewalld ya esta instalado, quizas hay que usar este?
#
- python3-firewall
  when: ansible_distribution == "CentOS" or
        ansible_distribution_major_version == "8"
```

Se modifica el archivo ~/roles/common/handlers/main.yml para que tome el nuevo servicio:

```
- name: restart chrony
  service:
    name: chronyd
    state: restarted
```

Validamos la ejecución del playbook

```
[root@bastion lamp]# ansible-playbook -i inventario roles/common/tasks/main.yml

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [172.16.0.4]
ok: [172.16.0.3]

TASK [Install chrony for Centos 8] *****
skipping: [172.16.0.3]
ok: [172.16.0.4]

TASK [Install Chrony on Debian] *****
skipping: [172.16.0.4]
ok: [172.16.0.3]

TASK [Install common dependencies on CentOS 8] *****
skipping: [172.16.0.3]
ok: [172.16.0.4]

TASK [Configure Chrony file] *****
ok: [172.16.0.3]
ok: [172.16.0.4]

TASK [Start the chrony service for CentOS 8] *****
ok: [172.16.0.3]
ok: [172.16.0.4]

PLAY RECAP *****
172.16.0.3      : ok=4   changed=0    unreachable=0    failed=0    skipped=2    rescued=0    ignored=
0
172.16.0.4      : ok=5   changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=
0

[root@bastion lamp]#
```

ROL DB

- Se actualiza el archivo lamp/roles/db/tasks/main.yml
- Se instalan los paquetes para Debian
- Se cambia el valor de state de *installed* a *present*. El estado *installed* no existe

```
TASK [Install MariaDB package for Debian] *****
fatal: [172.16.0.3]: FAILED! => {"changed": false, "msg": "value of state must be one of: absent, build-dep, fixe
d, latest, present, got: installed"}
```

Cambiamos el nombre de las dependencias de CentOS por las de Debian python3-
libselinux por Python-selinux y libsemanage por semanage

Para agilizar la instalación de las dependencias se instalan en el rol common los
paquetes necesarios para realizar la conexión de Python de mysql

```
# Se instalan las dependencias en CentOS 8
- name: Install common dependencies on CentOS 8
  dnf:
    name: "{{ dependencies }}"
    state: installed
  vars:
    dependencies:
      - python3-libselinux
      - python3-libsemanage
# El paquete firewalld ya esta instalado, quizas hay que usar este?
#   - python3-firewall
  when: ansible_distribution == "CentOS" or
        ansible_distribution_major_version == "8"

# Se instalan las dependencias en Debian
- name: Install common dependencies on Debian
  apt:
    name: "{{ dependencies }}"
    state: present
  vars:
    dependencies:
      - python3-selinux
      - python3-semanage
      - policycoreutils
      - selinux-utils
      - selinux-basics
# El paquete firewalld ya esta instalado, quizas hay que usar este?
#   - python3-firewall
  when: ansible_os_family == "Debian"
```

Se instala mariadb para ambos servidores (CentOS y Debian)

```
# Se instala los paquetes en CentOS
- name: Install MariaDB package for CentOS
  yum:
    name: "{{ paquetes }}"
    state: present
  vars:
    paquetes:
      - mariadb-server
# Se instalan paquetes de python
      - python3-pip
#
      - python-mysqldb
  when: ansible_os_family == "RedHat"

# Se instalan los paquetes para Debian
- name: Install MariaDB package for Debian
  apt:
    name: "{{ paquetes }}"
    state: present
  vars:
    paquetes:
      - mariadb-server
      - python3-mysqldb
  when: ansible_os_family == "Debian"
```

Se configura SELinux para MySQL en RedHat

```
# Configuración de SELinux
- name: Configure SELinux to start mysql on any port
  seboolean:
    name: mysql_connect_any
    state: true
    persistent: yes
  when: ansible_os_family == "RedHat"
```


Copiamos el archivo de configuración de mariadb en el cual debemos especificar el usuario y grupo

```
#MySQL Debian
- name: Create Mysql configuration file
  template:
    src: /root/obligatorio/obligatorio_2021_08/lamp/roles/db/templates/my.cnf.j2
    dest: /etc/mysql/my.cnf
    owner: mysql
    group: mysql
    mode: 0600
  notify:
    - restart mariadb
```

Se instalan las librerías de Python para MySQL

```
# Instalamos librerias necesarias de Python para MySQL
- name: Instalar MySQL-python Debian
  pip:
    name: mysql-connector-python
    executable: pip3
    state: present
#   when: ansible_facts['os_family'] == "Debian"

- name: Instalar PyMySQL
  pip:
    name: PyMySQL
    executable: pip3
    state: present
#   when: ansible_facts['os_family'] == "Debian"
```

Se crearon las nuevas reglas necesarias para MySQL con los manejadores del firewall de cada familia

```
- name: Start firewalld
  service:
    name: firewalld
    state: started
    enabled: yes
  when: ansible_facts['os_family'] == "RedHat"

- name: insert firewalld rule
  firewalld: port={{ mysql_port }}/tcp permanent=true state=enabled immediate=yes
  when: ansible_facts['os_family'] == "RedHat"

# Crear regla de Firewall Debian
- name: Firewall rule ufw
  ufw:
    rule: allow
    port: "{{ mysql_port }}"
    proto: tcp
  when: ansible_os_family == "Debian"
```

Se crea un usuario para poder hacer la conexión a la base de datos

```
# Crear usuario de conexion a la BD RH
- name: Crear usuario en mariadb in RH
  no_log: false
  mysql_user:
    name: "{{ dbuser }}"
    password: "{{ upassword }}"
    priv: '{{ dbname }}.*:ALL,GRANT'
    state: present
  when: ansible_facts['os_family'] == "RedHat"

# Crear usuario de conexion a la BD Debian
- name: Crear usuario en mariadb in Debian
  no_log: false
  mysql_user:
    login_unix_socket: /var/lib/mysql/mysql.sock
    name: "{{ dbuser }}"
    password: "{{ upassword }}"
    priv: '{{ dbname }}.*:ALL,GRANT'
    state: present
  when: ansible_facts['os_family'] == "Debian"
```

Se crean las bases de datos para ambas familias, pero a la hora de ejecutar las configuraciones, nos muestra en pantalla el siguiente error. Este error se debe a que la ejecución se deberá realizar por socket en vez de usuario y contraseña, ya que no tiene ninguna clave definida.

```
TASK [db : Create Application Database] *****
fatal: [172.16.0.3]: FAILED! => {"changed": false, "msg": "unable to find /root/.my.cnf. Exception message: (1698, \"Access denied for user 'root'@'localhost'\")"}

PLAY RECAP *****
172.16.0.3      : ok=15   changed=1   unreachable=0    failed=1   skipped=7   rescued=0   ignored=
0
172.16.0.4      : ok=15   changed=0   unreachable=0    failed=0   skipped=6   rescued=0   ignored=
0
```

```
# Creacion de BD para RH
- name: Crear BD de aplicacion en RH
  mysql_db:
    name: "{{ dbname }}"
    state: present
  when: ansible_facts['os_family'] == "RedHat"

# Creacion de BD para Debian
- name: Crear BD de aplicacion en Debian
  mysql_db:
    login_unix_socket: /var/lib/mysql/mysql.sock
    name: "{{ dbname }}"
    state: present
  when: ansible_facts['os_family'] == "Debian"
```

A la hora de crear un usuario nos devuelve el mensaje de que requiere setear no_log a true

```
TASK [db : Debian - Crear usuario en DB] *****
*****
[WARNING]: Module did not set no_log for update_password
fatal: [172.16.0.3]: FAILED! => {"changed": false, "msg": "unable to connect to database, check login_user and login_password are correct or /root/.my.cnf has the credentials. Exception message: (2003, \"Can't connect to MySQL server on 'localhost' ([Errno 2] No such file or directory)\")"}
```

Para resolver este inconveniente tuvimos que agregarlo al código

```
# Creacion de usuario para conexion a la BD para RH
- name: Crear usuario en BD para RH
  no_log: true
  mysql_user:
    name: "{{ dbuser }}"
    password: "{{ upassword }}"
    priv: '{{ dbname }}.*:ALL,GRANT'
    state: present
  when: ansible_facts['os_family'] == "RedHat"

# Creacion de usuario para conexion a la BD para Debian
- name: Crear usuario en BD para Debian
  no_log: true
  mysql_user:
    login_unix_socket: /var/lib/mysql/mysql.sock
    name: "{{ dbuser }}"
    password: "{{ upassword }}"
    priv: '{{ dbname }}.*:ALL,GRANT'
    state: present
  when: ansible_facts['os_family'] == "Debian"
```

WEB

Se instalan los paquetes de Apache, en este paso, se adaptó el código para evitar la autorización del issue mencionado anteriormente

```
# Se configura la instalacion para CentOS
- name: Install httpd and php on CentOS
  yum:
    name: "{{ paquetes }}"
    state: present
  vars:
    paquetes:
      - httpd
      - php
      - php-mysqlnd
  when: ansible_os_family == "RedHat"

# Se configura la instalacion para Debian
- name: Install httpd and php on Debian
  apt:
    name: "{{ paquetes }}"
    state: present
  vars:
    paquetes:
      - apache2
      - php
      - php-mysql
  when: ansible_os_family == "Debian"
```

Se agrega la instalación de los paquetes de git para Debian

```
# Se instalan los paquetes git
- name: Install web role specific dependencies Debian
  apt:
    name: "{{ paquete_git }}"
    state: present
  vars:
    paquete_git:
      - git
  when: ansible_os_family == "Debian"
```

Crean las reglas necesarias para cada familia de Linux

```
# Levantar CentOS
- name: Start firewalld
  service:
    name: firewalld
    state: started
    enabled: yes
  when: ansible_os_family == "RedHat"

# Crear regla de Firewall Debian
- name: Firewall rule ufw
  ufw:
    rule: allow
    port: "{{ httpd_port }}"
    proto: tcp
  when: ansible_os_family == "Debian"
```

Configuramos el servicio de los paquetes para las diferentes versiones, esto es así porque los paquetes tienen un nombre diferente para cada versión.

```
#Levantar el servicio de Apache en CentOS
- name: http service state
  service: name=httpd state=started enabled=yes
  when: ansible_os_family == "RedHat"

#Levantar el servicio de Apache en Debian
- name: Apache service state
  service: name=apache2 state=started enabled=yes
  when: ansible_os_family == "Debian"
```

Le agregamos el mode como se menciona anteriormente

```
- name: Creates the index.php file
  template:
    src: "index.php.j2"
    dest: "/var/www/html/index.php"
    mode: 0600
```

Y finalmente ejecutamos el playbook para chequear los errores de este

```
TASK [db : Start firewalld] *****
skipping: [172.16.0.3]

TASK [db : insert firewalld rule] *****
skipping: [172.16.0.3]

TASK [db : Firewall rule ufw] *****
[WARNING]: The value 3306 (type int) in a string field was converted to '3306' (type string). If this does not look like what you expect, quote
the entire value to ensure it does not change.
ok: [172.16.0.3]

TASK [Crear usuario en mariadb in RH] *****
skipping: [172.16.0.3]

TASK [Crear usuario en mariadb in Debian] *****
[WARNING]: Module did not set no_log for update_password
ok: [172.16.0.3]

TASK [db : Crear BD de aplicacion en RH] *****
skipping: [172.16.0.3]

TASK [db : Crear BD de aplicacion en Debian] *****
ok: [172.16.0.3]

TASK [db : Crear usuario en BD para RH] *****
skipping: [172.16.0.3]

TASK [db : Crear usuario en BD para Debian] *****
ok: [172.16.0.3]

PLAY RECAP *****
172.16.0.3      : ok=18  changed=1  unreachable=0  failed=0  skipped=10  rescued=0  ignored=0
172.16.0.4      : ok=15  changed=0  unreachable=0  failed=0  skipped=6   rescued=0  ignored=0

[root@bastion lamp]#
```

Validamos que podamos entrar a la url

< > C : http://192.168.0.71/

Hello World! My App deployed via Ansible V6.

BIBLIOGRAFÍA

Ejemplo de ansible.cfg

<https://github.com/ansible/ansible/blob/devel/examples/ansible.cfg>

Ejemplo de playbooks

https://docs.ansible.com/ansible/latest/user_guide/playbooks_conditionals.html

Ejemplo de ansible facts

https://docs.ansible.com/ansible/latest/user_guide/playbooks_vars_facts.html