

# Galera Cluster, Alta Disponibilidad Multimaster para MariaDB (...y MySQL) sobre CentOS 7... ...con soporte de réplicas geo distribuidas.

---



## Introducción

La alta disponibilidad es una necesidad en el mundo de los servicios y tecnologías asociadas a Internet, y las bases de datos no están ajena a esta necesidad. Hay muchos esquemas y soluciones para dotar de sistemas tolerantes a fallos a los motores de bases de datos basados en MySQL y su fork, MariaDB, y luego de una larga investigación y profundas reflexiones respecto a pros y contras de las soluciones Master-Slave (Asíncronas, Semi-síncronas) y Master-Master o Multi Master, he decidido implementar la solución basada en Galera Cluster ([www.galeracluster.com](http://www.galeracluster.com)) ya que es una real solución de alta disponibilidad Multimaster para MySQL y MariaDB elegida por grandes portales web para dotar a sus sistemas de una

---

---

geodistribucion tolerante a fallos ([www.mercadolibre.com](http://www.mercadolibre.com)), mejorando la seguridad y réplica de los datos de misión crítica. Otro factor no menos importante que me hizo elegir la solución de Galera, es que MariaDB incluye las librerías de Galera como parte de su distribución desde los últimos releases, sin que debamos parchar los fuentes de la base de datos ni instalar Galera como software aparte, siendo hoy por hoy la solución de facto de alta disponibilidad de MariaDB, al estar disponible en su instalación sin mayores problemas y sólo deberemos configurar, sin instalar nada adicional si queremos tener una BBDD replicada y en HA con MariaDB.

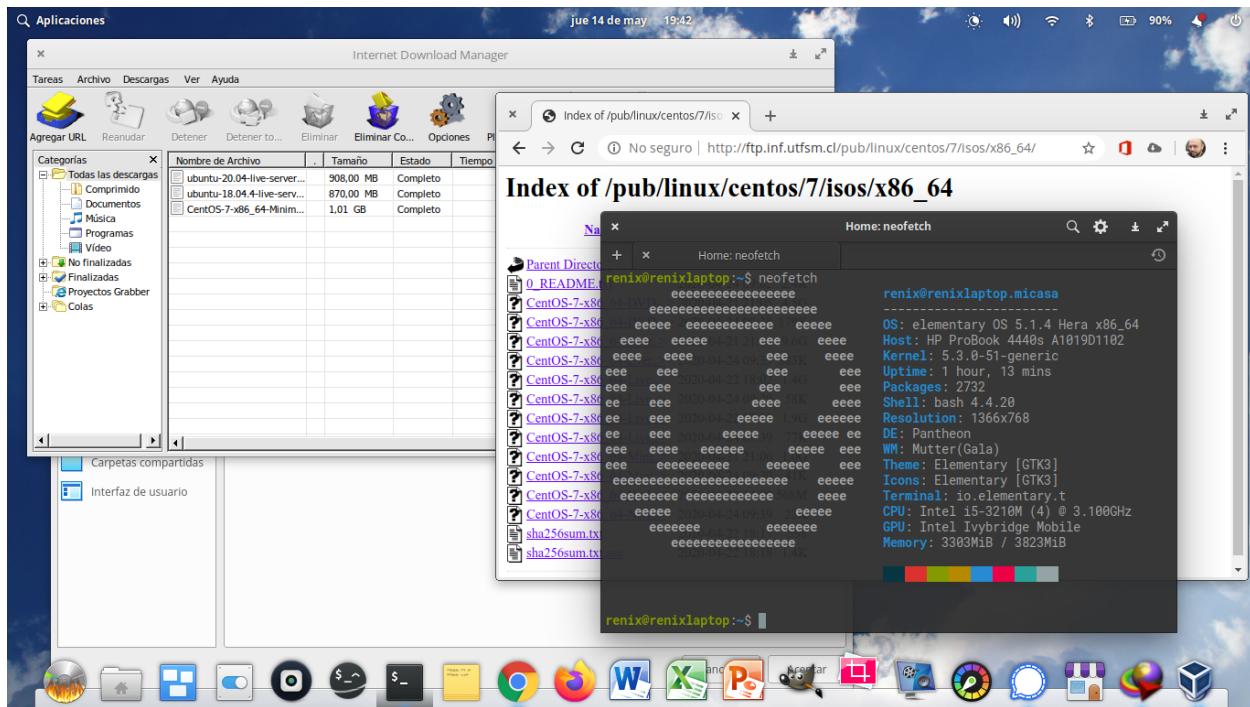
Como MySQL ahora pertenece a Oracle, Galera pone a disposición repositorios con paquetes parchados y compilados de la versión community de MySQL para facilitar el proceso de instalar Galera Cluster con MySQL, lo cual se agradece y se tratará como anexo.

En este documento trataremos el caso de MariaDB en un primer apartado, que nos permitirá familiarizarnos con la configuración relativamente sencilla del Cluster Multimaster de la tecnología de Galera, y en un anexo veremos la instalación de su homólogo para MySQL con los paquetes DEB parchados provistos también por Galera.

Sin más preámbulos, los invito a esta nueva proeza que nos permitirá salir más pronto de problemas y por ende, de las Llamas ;)

## **Requisitos Base**

Para esta misión utilizaremos una distribución Linux basada en RedHat, a saber, CentOS Versión 7, para lo cual descargamos la imagen ISO desde el mirror más cercano, en mi caso el mirror que tienen mi querida UTFSM. Como podemos ver la imagen de la versión Mínima de Centos 7 pesa 1 Gigabyte, por lo que la descargaré ayudado por una muy buena aplicación para Windows funcionando bajo WINE llamada Internet Download Manager, que aunque no viene al caso, quiero jactarme de lo bien que corren muchas muy buenas aplicaciones para Windows en Linux de esta forma, y aquí les dejo a modo de paréntesis mi escritorio híbrido con Elementary OS 5.1 y apps windows funcionando a la perfección, de lo cual podría hacer un documento de las alternativas que existen para tener un sistema Híbrido GNU/Linux más aplicaciones de Windows (Si, el pingüino saltó hace rato por la ventana y la ventana se abrió un buen poco al FLOSS, así que bien por todos.



## Hardware para Galera Cluster

Como requisito mínimo para el Cluster de Galera es necesario tres servidores (físicos o virtuales) que operarán como 3 nodos Master, y se replicarán de forma síncrona y balancearán su carga en operación. Los requisitos mínimos de hardware [según la documentación de Galera](#) para cada servidor son:

- 1 GHz single core CPU;
- 512 MB RAM; y
- 100 Mbps network connectivity

Idealmente la **red de replicación** del Cluster debiese ser una red física aparte de la red que provisionan los datos para que no existan problemas de latencia de tráfico en el proceso de replicación. Para ello, crearemos tres máquinas virtuales en VirtualBox y cada una con dos tarjetas de Red para emular este requisito. Cada servidor tendrá una tarjeta de red mirando a la **red de servicio** (mi red LAN) y una tarjeta de red mirando a la red de replicación.

- La red de servicio tendrá direcciones IP del rango **192.168.100.0/24** autoasignadas por un servidor DHCP.
- La red de replicación tendrá direcciones IP del Rango **10.10.10.0/24** asignadas de manera manual (estática)

Como Sistema Operativo hemos elegido en esta oportunidad CentOS versión 7 Minimal, para asegurarnos de tener instalado solo lo necesario para nuestro Cluster.

## Paso 0: Instalación de CentOS 7 en VirtualBox

**Nota:** Si usted es un usuario avanzado o recurrente de Virtualbox y/o CentOS, y no quiere aburrirse con este apartado de Instalación de CentOS para los SysAdmin más noveles, puede ir a la página XX para ver el proceso desde la instalación de MariaDB

**Nota:** Este paso debe realizarse tres veces, ya que aún no descubro la razón de porque no funciona la replicación clonando la primera instalación dos veces, pero es cosa de tiempo.

Arrancamos VirtualBox y para crear una nueva VM con CentOS pulsamos el botón **Nueva** como se aprecia en la siguiente figura:



Aparecerá el cuadro de diálogo que nos pedirá el nombre de la nueva VM, y colocaremos el nombre **CentosMaster01**, y Virtualbox adivinará el tipo de Sistemas Operativo y pre seleccionará una VM para Linux y una distribución Red Hat de 64 Bits



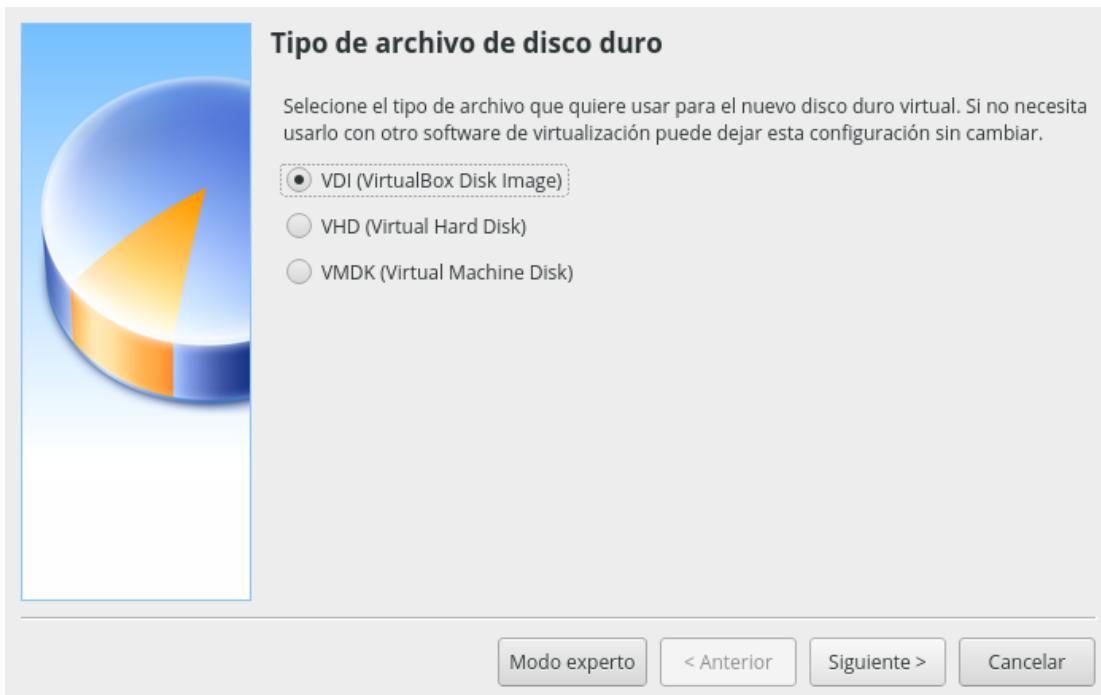
Pulsamos en Siguiente y se nos pedirá configurar la cantidad de memoria RAM que asignaremos a dicha VM. En mi caso y para efectos prácticos sólo se asignan 700 Mbytes de memoria RAM, ya que con menos de 500 Mbytes MySQL y MariaDB no arrancan.



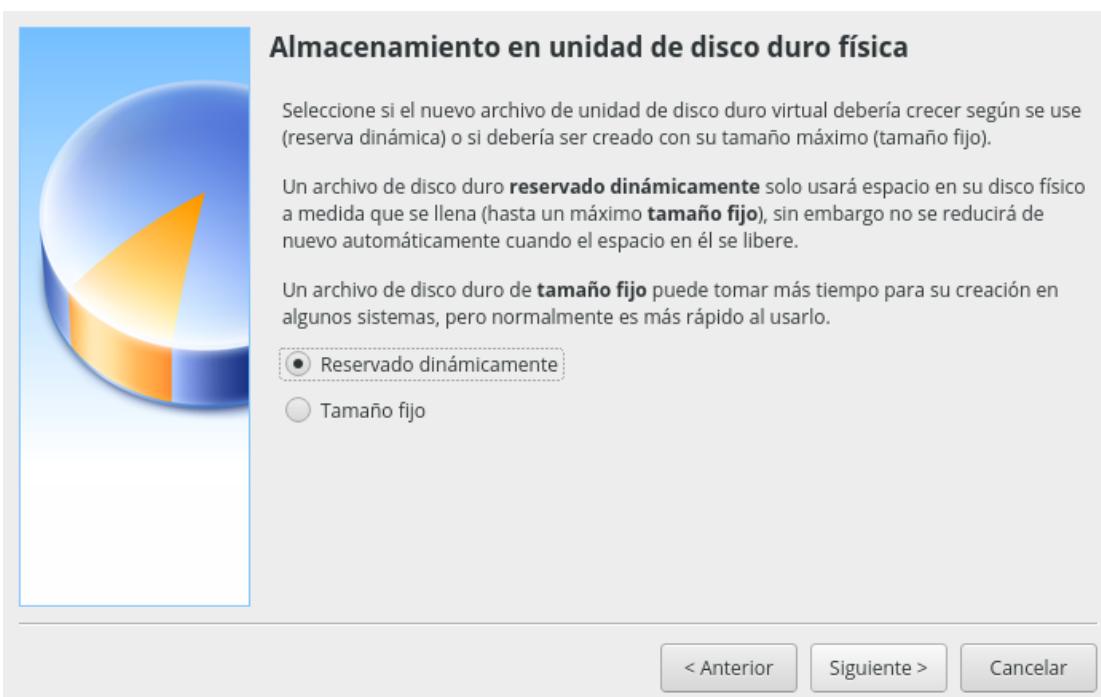
Pulsamos nuevamente en Siguiente y deberemos configurar un disco duro virtual de tipo **Thin Provision** de aproximadamente 30 Gigabytes para la VM, como se aprecia en las siguientes figuras:



Seleccionamos la opción de disco tipo VDI



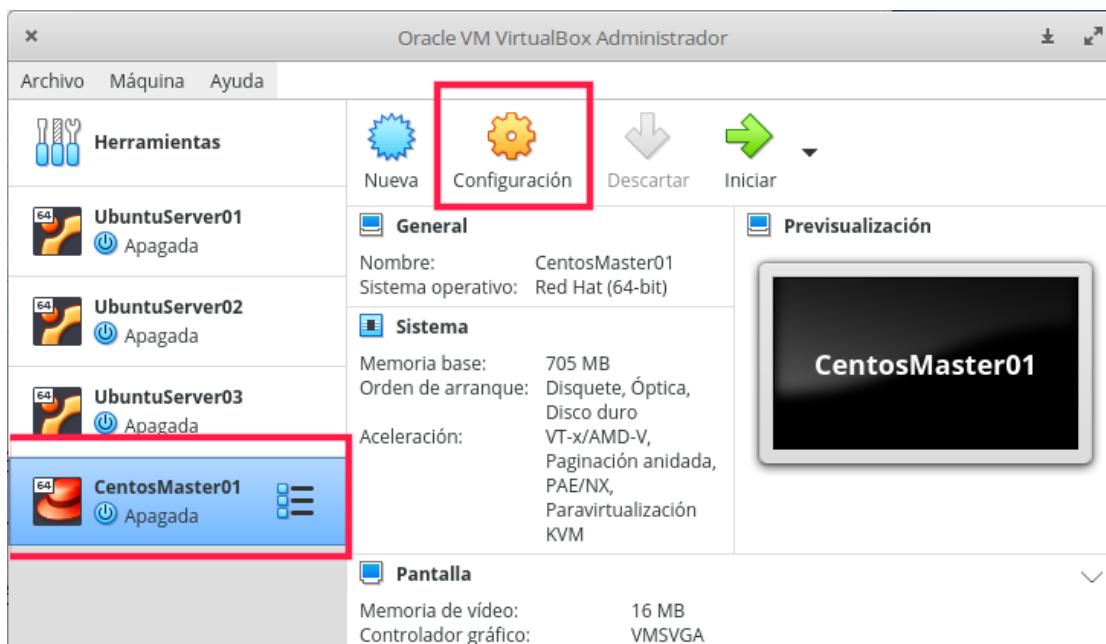
y Luego la Opción de Thin Provision que aca se llama Reservado Dinámicamente



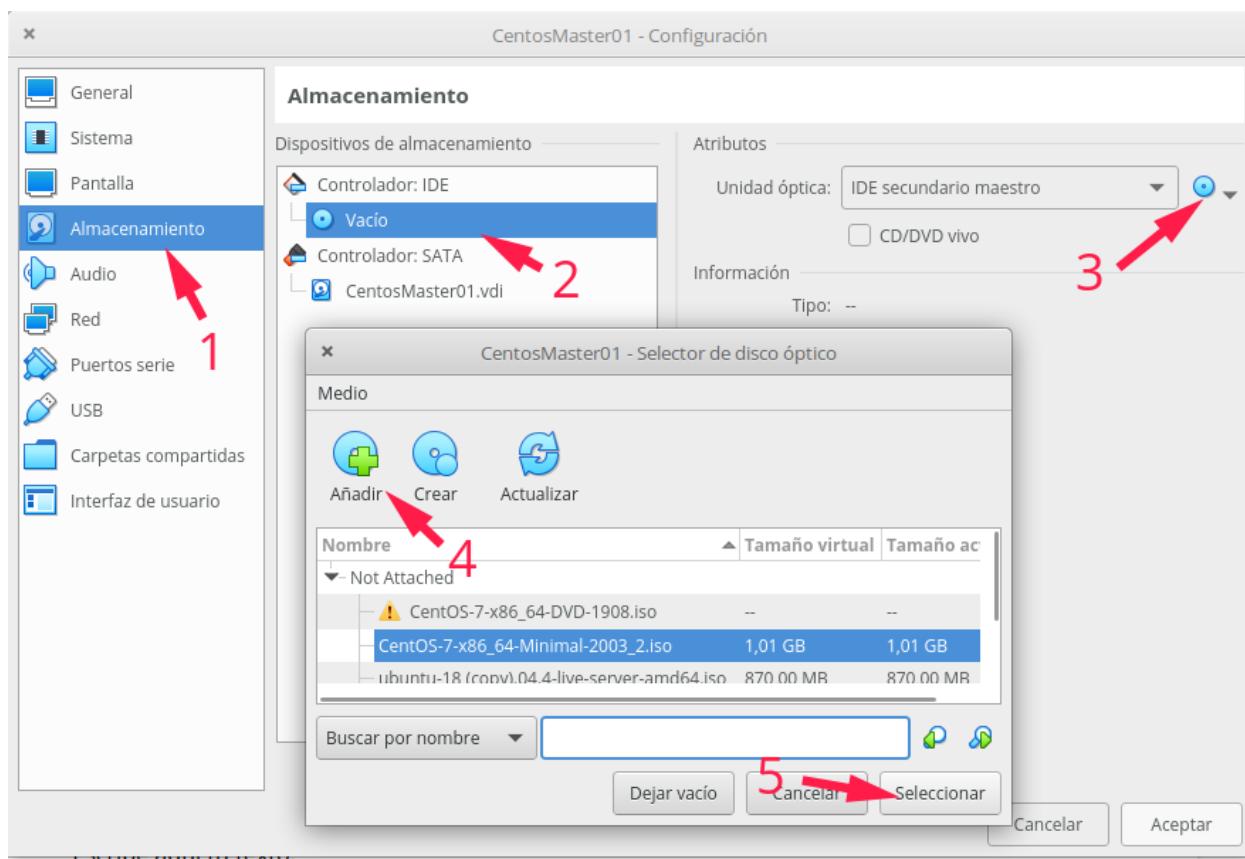
y por último asignamos un tamaño cercano a los 30 Gbytes, que al ser un Disco de tipo Thin sólo ocupará espacio según se vaya necesitando, sin perjuicio de que el Sistema Operativo Guest creerá que tiene un disco real de ese tamaño.



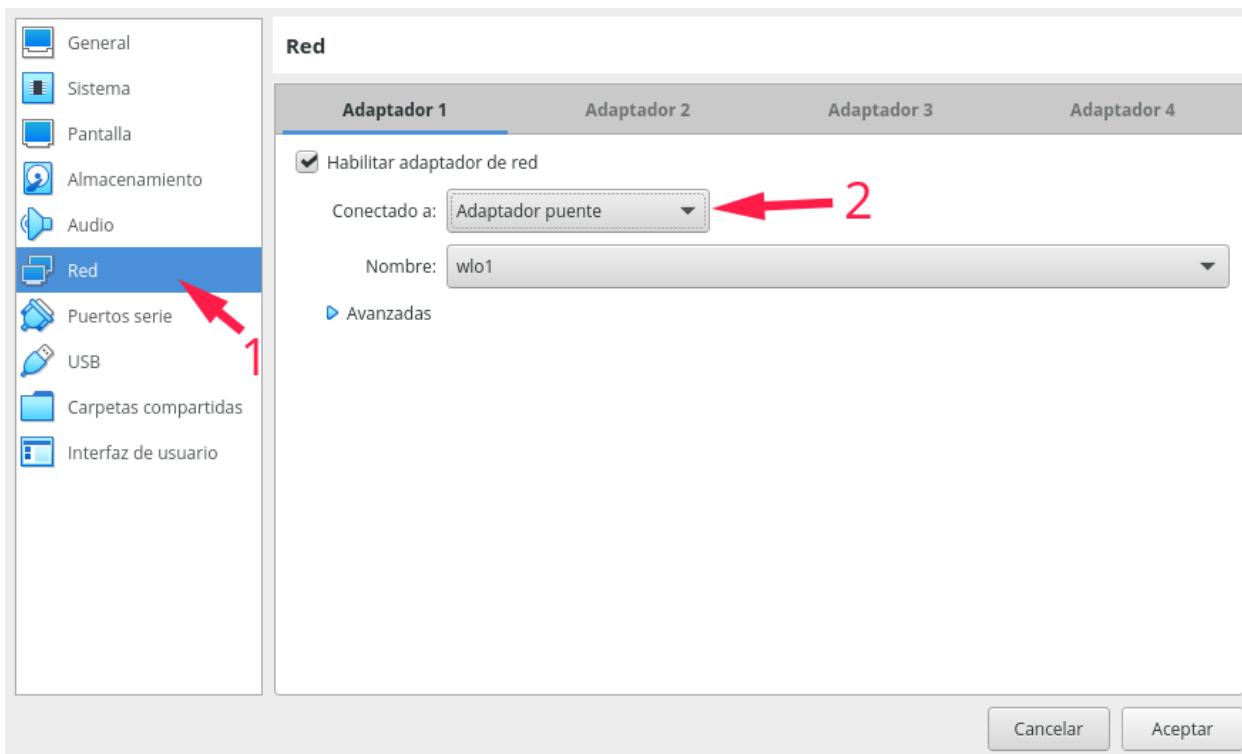
Ahora que se ha creado la VM para CentOS 7, pulsamos en el botón **Configuración** para colocar la ISO de CentOS como unidad óptica virtual y después configurar la red de esta VM



Se nos abre el panel de configuración de la máquina virtual creada y hacemos click en Almacenamiento (1) luego hacemos click en la unidad de disco óptico (2), luego en el icono de disco óptico seleccionamos la Opción de Agregar imagen de disco óptico (3) y se abrirá el almacén de imágenes ISO de Virtualbox, hacemos click en Añadir (4) y debemos seleccionar la imagen iso de Centos 7 que bajamos, y luego de que figure resaltada en el inventario, hacemos click en el botón Seleccionar (5)

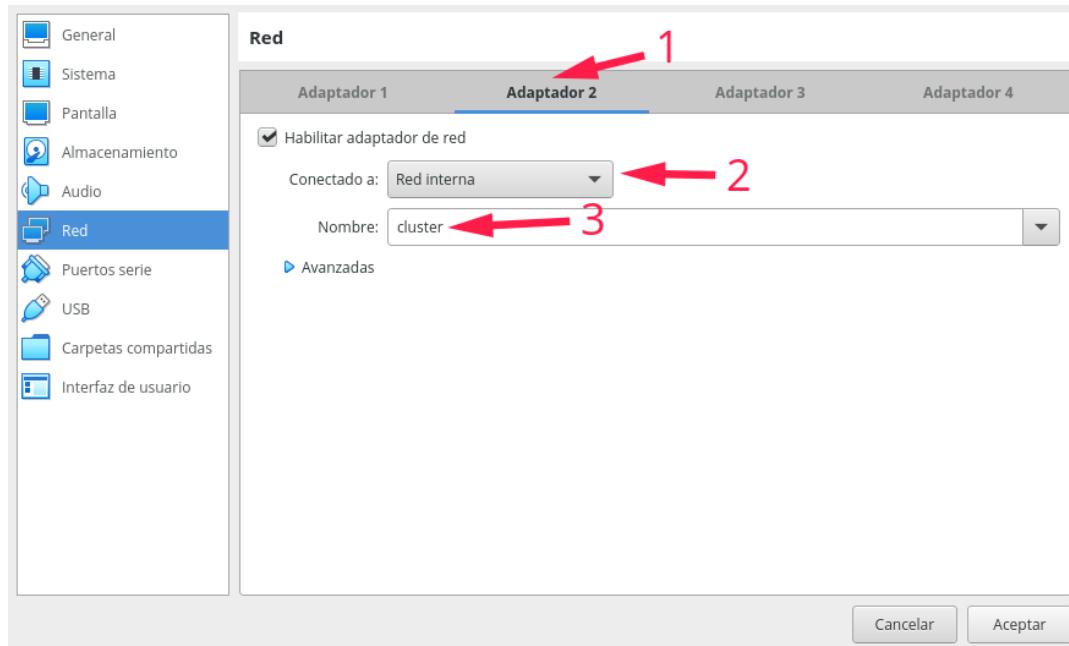


Luego vamos a configuración de Red (1) y seleccionamos la opción de Adaptador Puente (2) (en mi caso lo hice así para poder ver los servicios de la VM desde mi red LAN) creando un bridge hacia mi interfaz wifi wlo1

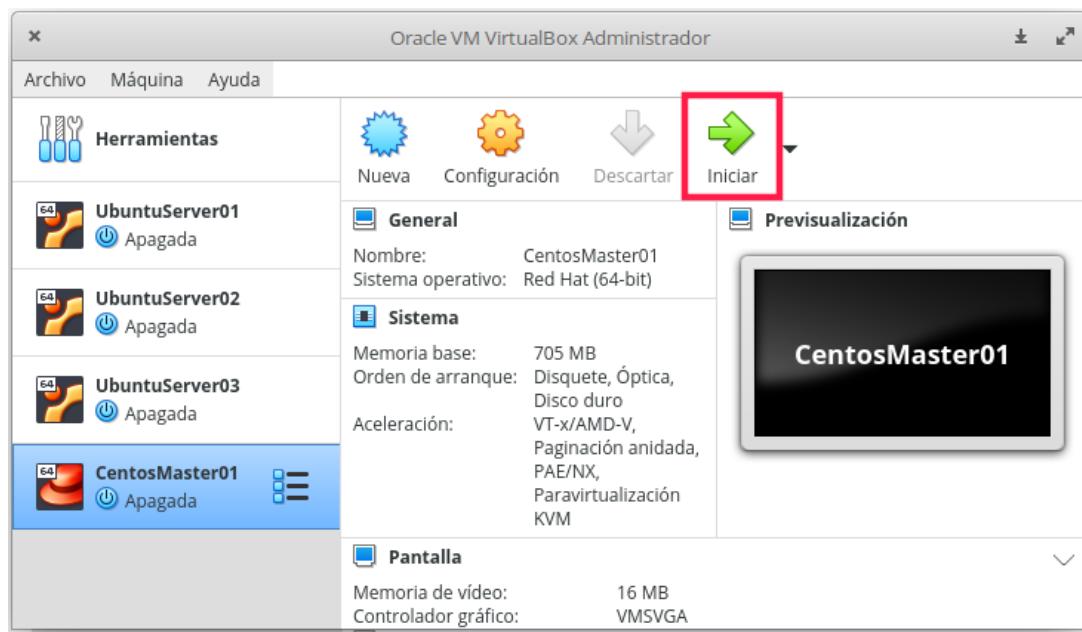


Luego hacemos click en Adaptador 2 (1) , y en el listbox “Conectado a” seleccionamos la opción Red Interna (2) y damos un nombre a esta red interna (3), que en este caso he elegido el nombre “cluster” para simular la red de replicación. Este nombre de la Red Interna “cluster” deberá ser el mismo en las tres VM que vamos a crear para que exista conectividad entre todas las VM de esa red dedicada con las direcciones IP del rango **10.10.10.0/24** que asignaremos a cada una.

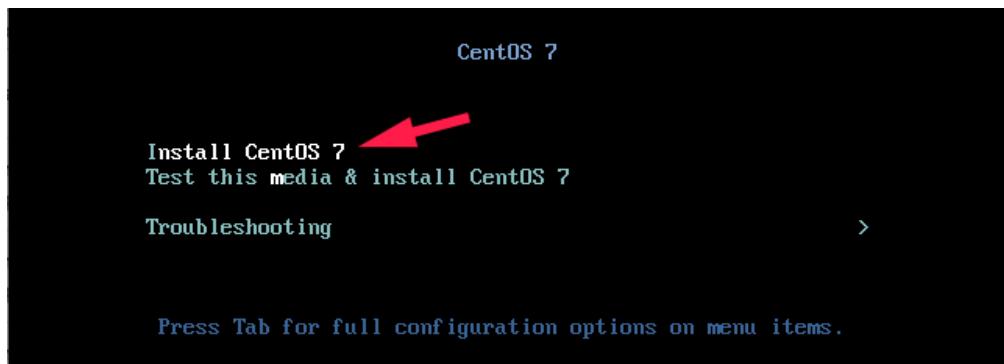
Finalmente apretamos el botón Aceptar para terminar la configuración.



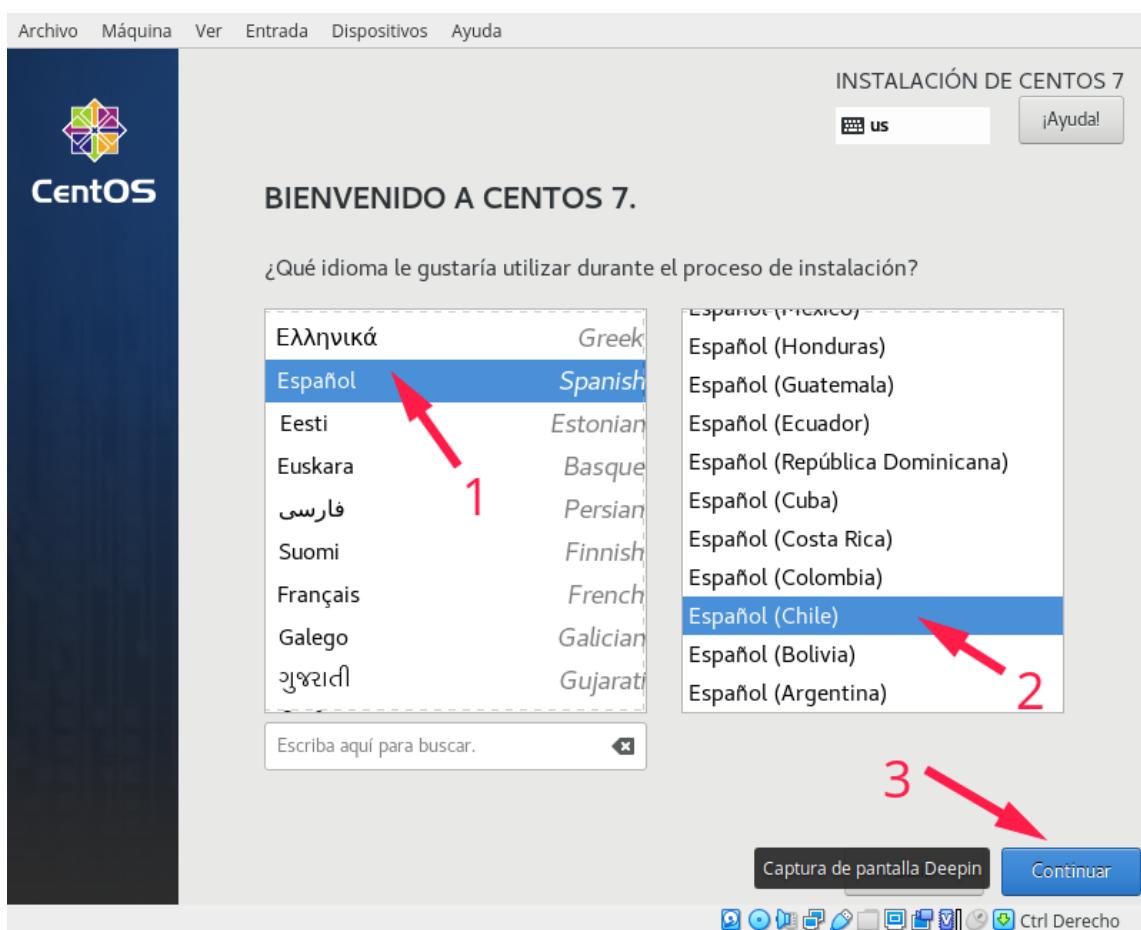
Ahora deberemos presionar en el botón iniciar como se muestra en la siguiente figura:



Vemos que la VM booteó correctamente desde la unidad óptica virtual sobre la cual montamos la imagen de CentOS 7 Minimal y debemos seleccionar (obviamente) la Opción *Install CentOS 7* como se muestra en la siguiente figura:



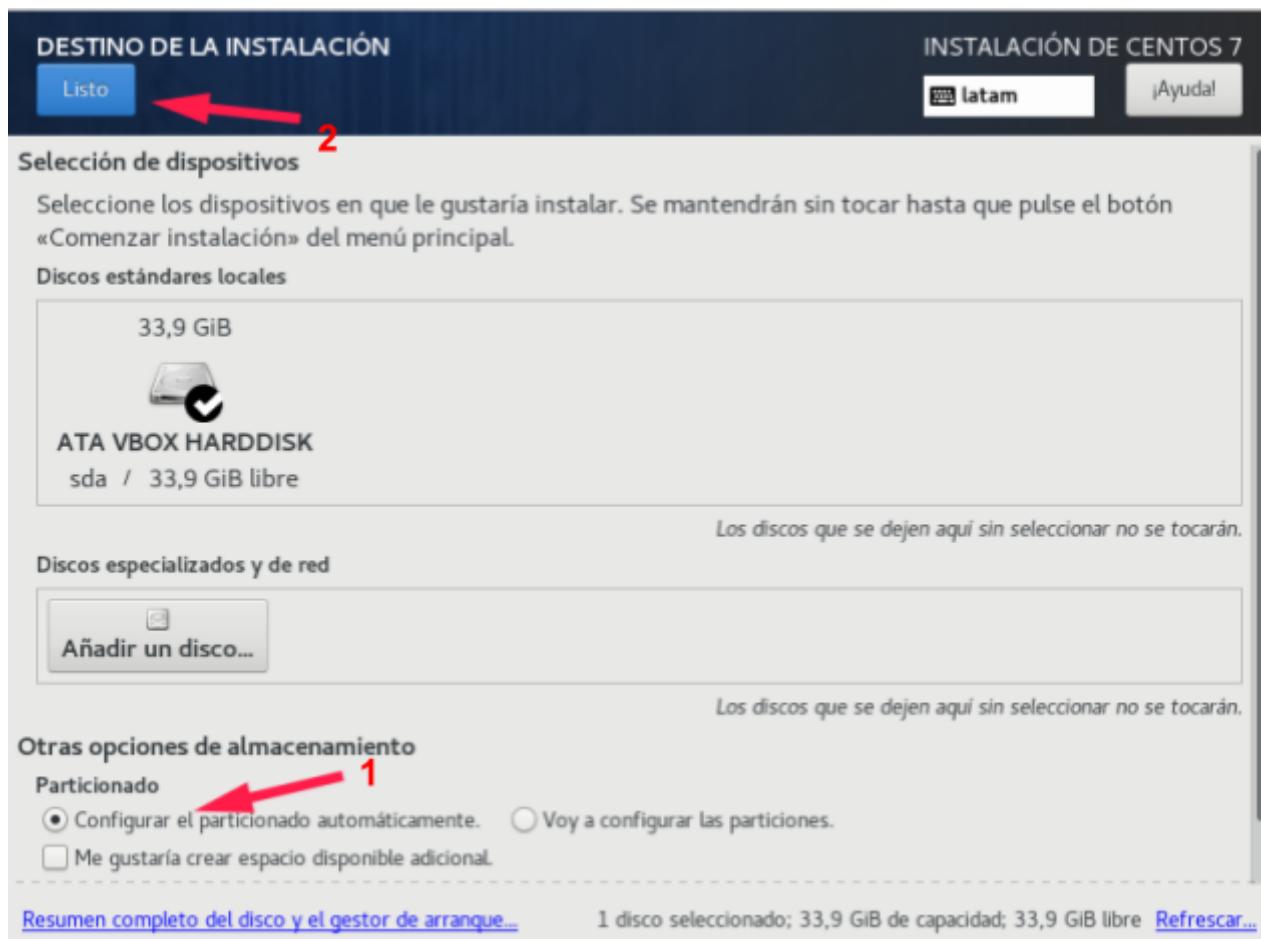
Se nos presenta la pantalla gráfica de instalación y elegimos idioma Español (1) y Español Chile (2) y luego pulsamos Continuar (3)



Luego se nos presenta un resumen de la configuración que por defecto tendrá nuestra instalación y solo deberemos ver el tema del particionado de discos pulsando en el ícono Destino de la Instalación.



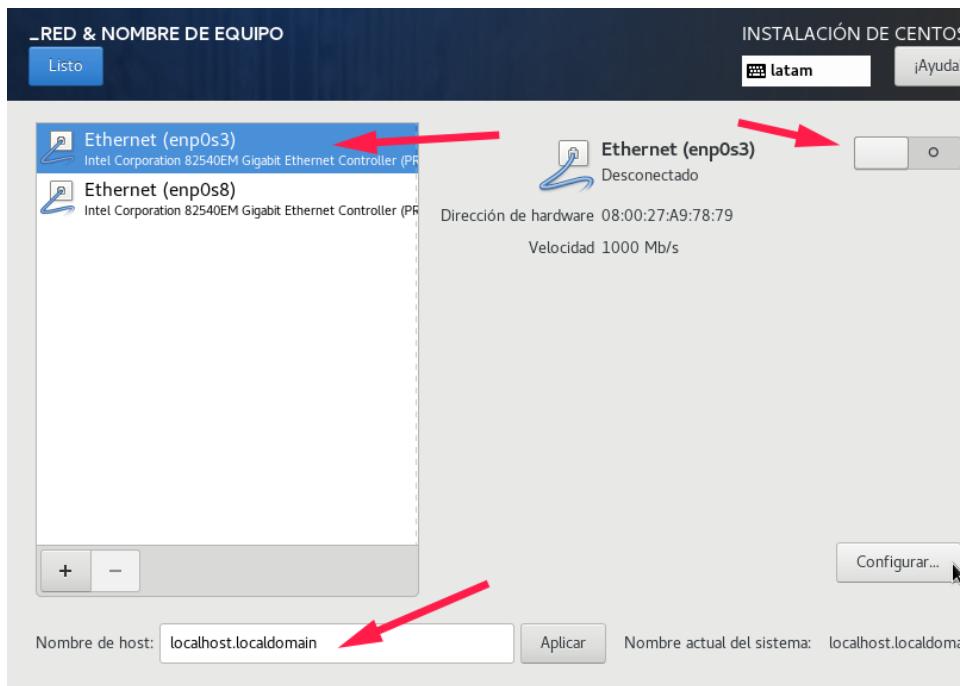
Debe estar seleccionada la opción “Configurar el particionado automáticamente” (1) y luego presionar el botón listo de la parte superior (2).



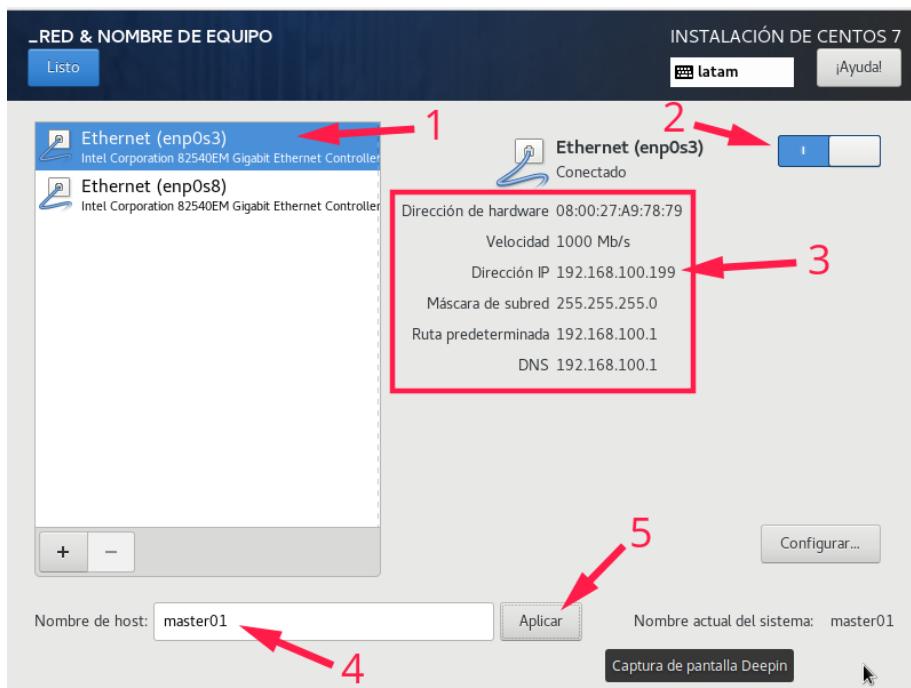
Y ahora toca el turno de configurar la red de servicios 192.168.100.0/24 por DHCP y la red de replicación 10.10.10.0/24 manualmente (Static) haciendo click en “Red y Nombre de Equipo”



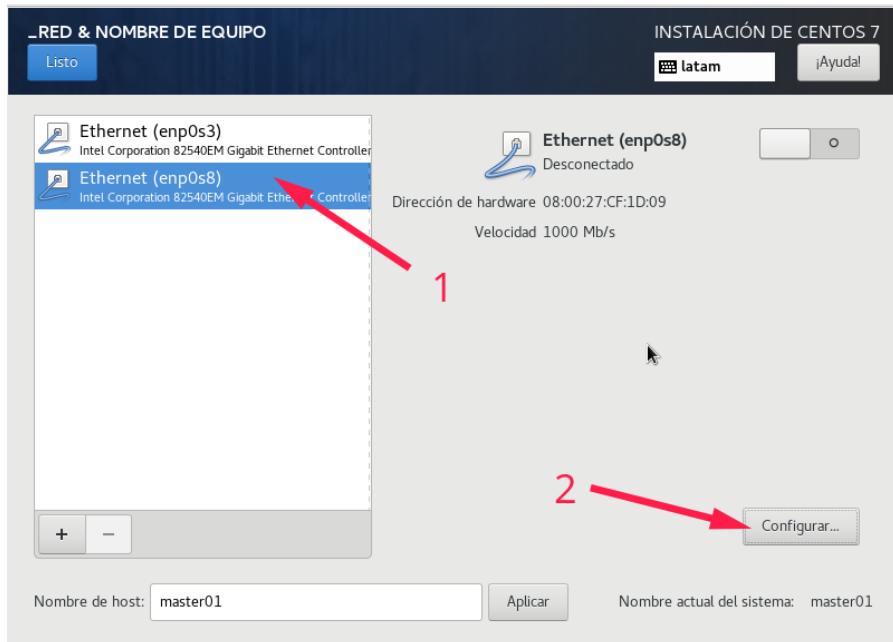
Luego seleccionamos el adaptador de red **enp0s3** y encendemos la interfaz con el interruptor a la derecha, y cambiamos el nombre del host abajo



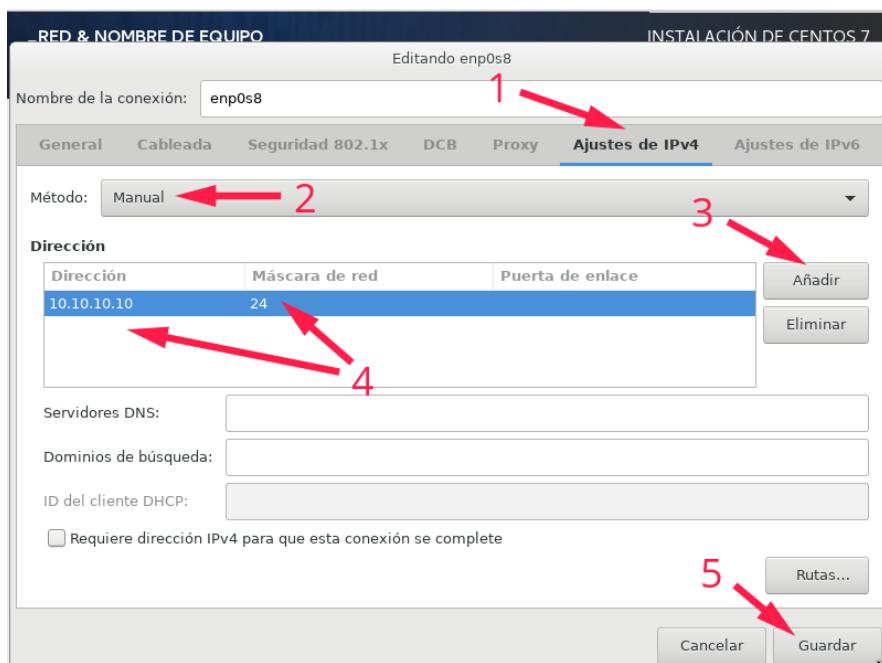
Al Encender el conmutador (2) se levanta la interfaz **enp0s3** (1) y recibe la dirección IP 192.168.100.199/24 del servidor DHCP de mi red LAN (3) y luego cambiamos el nombre de host a **master01** (4) y pulsamos el botón Aplicar (5)



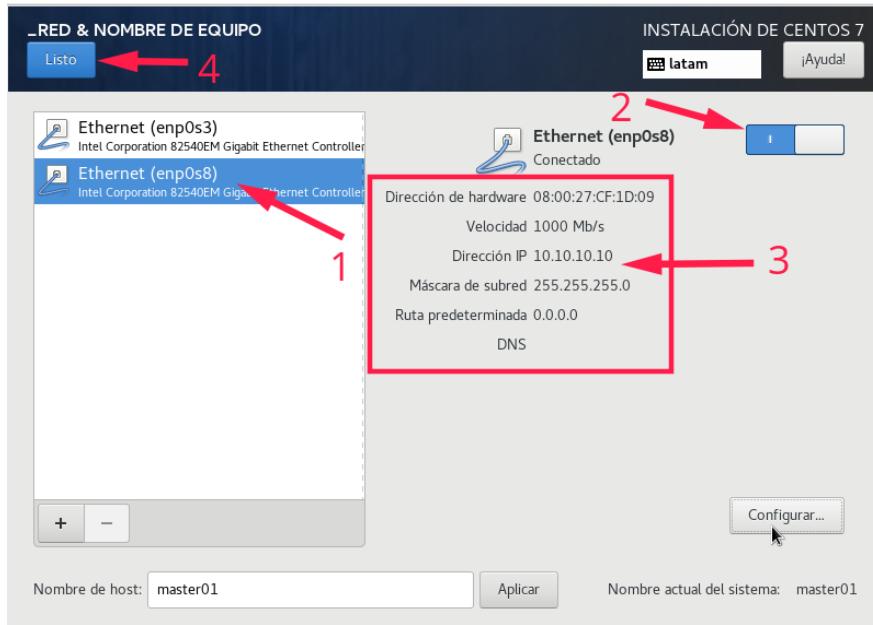
Para la interfaz de la red de replicación seleccionamos la interfaz **enp0s8** (1) y luego pulsamos el botón “Configurar” (2)



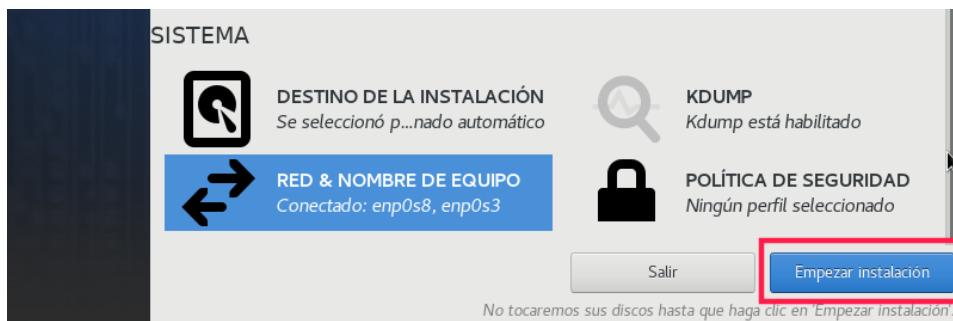
Se nos abrirá el cuadro de diálogo de configuración manual de la interfaz, y debemos pinchar en Ajustes de IPv4 (1), Luego elegir Método “Manual” (2), luego pulsar el botón “Añadir” (3) e ingresar la dirección IP 10.10.10.10 y la mascara de subred de 24 bits (4) y finalmente pulsar el botón “Guardar” (5) como se aprecia en la siguiente figura.



Luego debemos levantar la interfaz **enp0s8** (1) comutando el interruptor (2) y veremos reflejada la configuración manual de la interfaz de replicación en la red IP destinada para este efecto (3) y luego salir de la configuración de red pulsando el botón “Listo” (4).



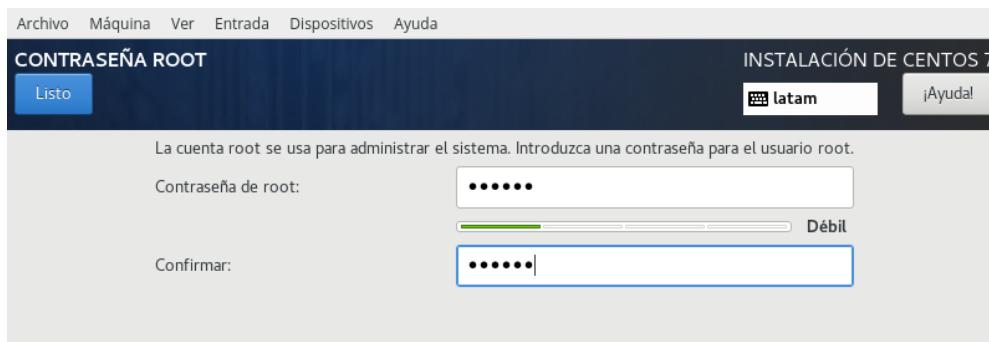
Luego volvemos a la interfaz de instalación de CentOS 7 y pulsamos el botón “Empezar Instalación” como se muestra en la siguiente figura



Ahora debemos configurar la contraseñas de root y crear un nuevo usuario con privilegios de administrador (sudoer). Hacemos click en el ícono “Contraseña de root”



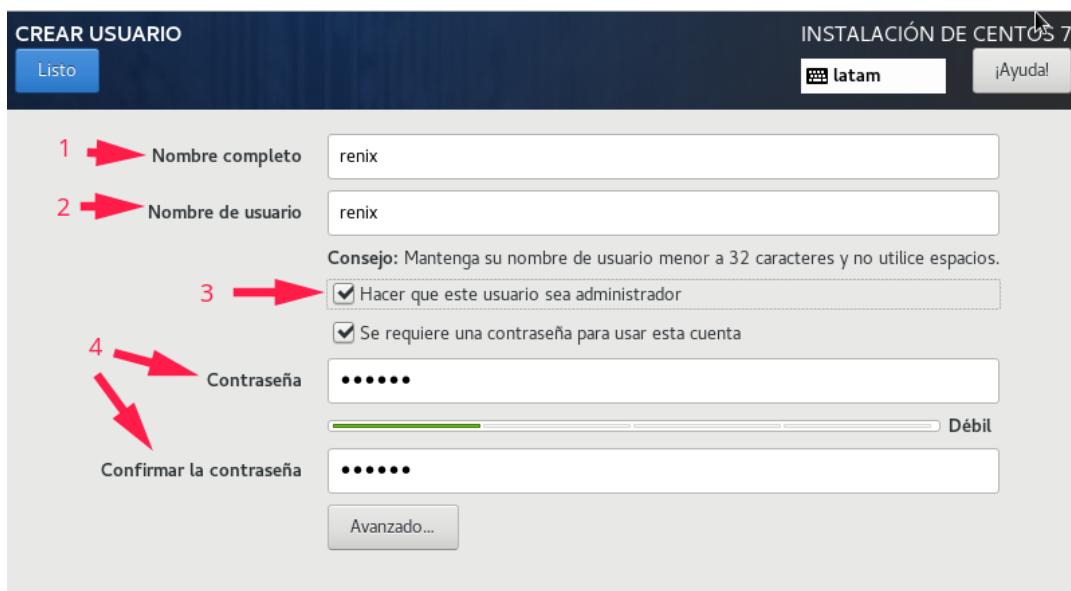
y luego ingresamos una contraseña dos veces.



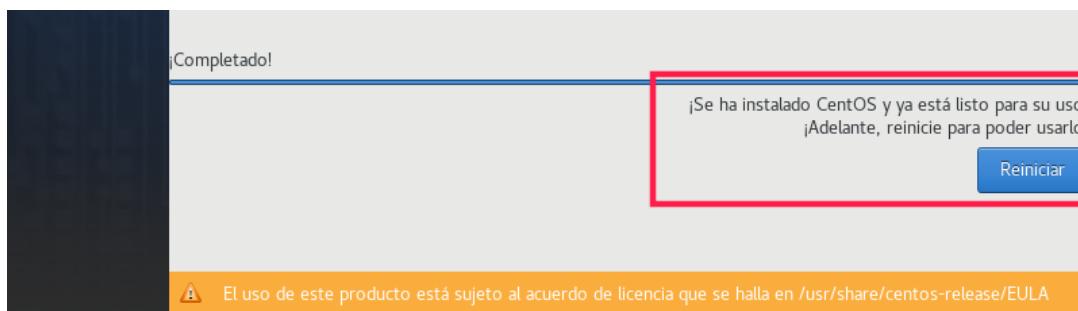
Ahora crearemos un usuario sudoer con privilegios de administrador haciendo click en el icono "Creación de Usuario"



y llenamos los campos de la interfaz de creación de usuario con el nombre de la persona (1), con el nombre de usuario o login (2), Marcamos la casilla que indicará que el usuario a crear sea administrador (sudoer) (3) y colocamos la contraseña (4) para finalmente pulsar en el botón superior izquierdo “Listo”



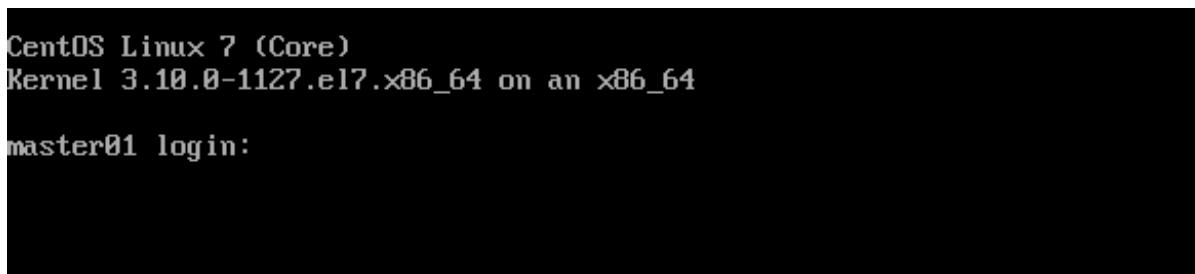
y mientras hacíamos todo lo anterior, en background se instalaba al Sistema Operativo y solo nos queda pulsar el Botón Reiniciar.



**Nota:** Se que muchos podrán pensar sobre el porqué me di el trabajo de documentar la instalación de CentOS 7 sobre Virtualbox, y la respuesta es muy simple: Sysadmin en Llamas busca acercar a talentos noveles a esta hermosa área TI, y busca generar documentación consolidada, con recetas de inicio a fin para laboratorios, y no tener que ir picando por un sitio u otro en busca de parte de los ingredientes; y si eres un usuario que conoce estos tópicos de sobra, simplemente adelántate y disfruta de la siguiente parte.

## Paso 1: Configuración post instalación para cada nodo

La VM con CentOS 7 ha reiniciado mostrando la clásica bienvenida a hacer login. Desde ahora usará un programa de terminal para conectarme por SSH a esta VM y así poder copiar y pegar texto en vez de estar copiando y pegando imágenes.



**Nota:** Galera Cluster puede colgarse ocasionalmente cuando corre en hardware muy limitado, debido a insuficiencia de memoria. Para prevenir esto asegúrese de tener suficiente espacio de Swap en una partición o en un archivo de Swap. Para más información sobre cómo crear más espacio de Swap puede ir a [Configuring Swap Space](#) del sitio de Galera

..y accedemos vía SSH a la VM que ahora tiene la dirección IP 192.168.100.208 y hacemos Login con el usuario root y actualizaremos el Sistema Operativo:

```
# yum -y update
```

---

Luego instalamos el nuevo manejador de paquetes **dnf**, y luego el paquete **net-tools** para poder tener al mítico comando **ifconfig** de vuelta, ya que no viene instalado por defecto en CentOS (si, se puede usar **ip addr** en su lugar pero la costumbre mata :)

```
# yum -y install dnf  
# dnf install -y net-tools mlocate
```

Además, como es una instalación mínima de CentOS, deberemos instalar varias utilidades: **Midnight Commander**, **nmon**, **htop**, **nano**, **nmap** y otras utilidades.

```
# dnf -y install mc nano nmap unzip
```

Para instalar **nmon** necesitamos los repositorios epel de CentOS

```
# dnf -y install epel-release  
# dnf -y install nmon htop
```

Para hacer que funcione la réplica de la BBDD entre nodos necesitamos instalar rsync

```
# dnf -y install rsync
```

Además, para configurar adecuadamente las políticas de SELinux en cada máquina debemos instalar **policycoreutils**

```
# dnf -y install policycoreutils-python
```

...y porqué no , instalar **neofetch** (porque es un chiche ASCII)

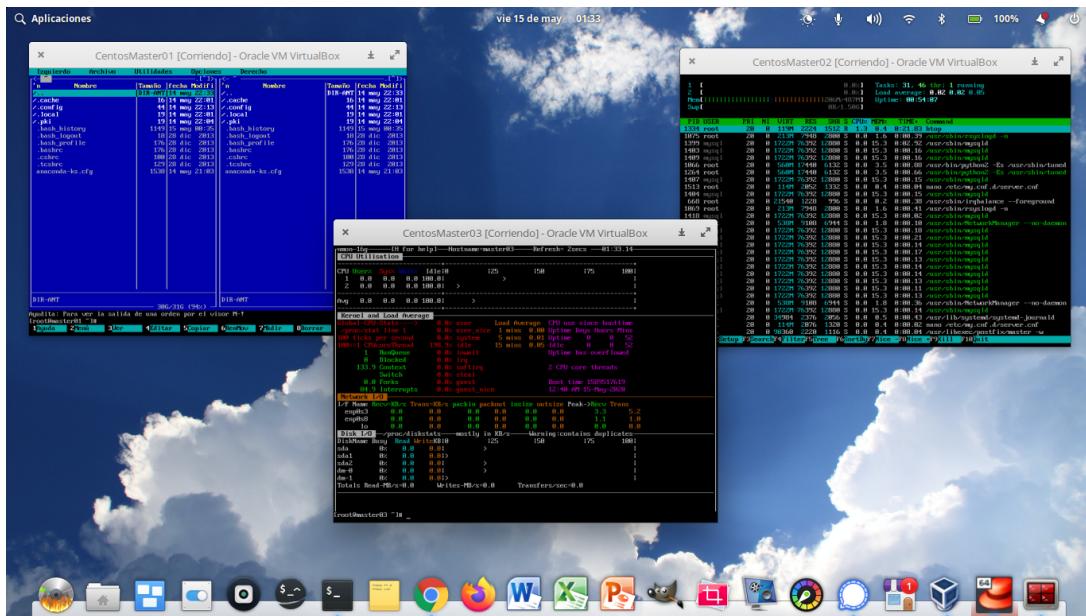
```
# dnf -y install dnf-plugins-core  
# dnf -y copr enable konimex/neofetch  
# dnf -y install neofetch
```

y ejecutamos neofetch ...y tenemos de regalo un arte ASCII con info de nuestra VM (más bonito que el de Elementary OS)

```
[root@master01 ~]# neofetch
Fetch
      .PLTJ.
      <><><><>
      KKSV' 4KKK LJ KKKL.'VSSKK
      fo de KKV' 4KKKKK LJ KKKKAL'VKK el de El
      V' 'VKKKK LJ KKKK' 'V
      .4NA.' 'VKK LJ KKV' '.4Mb.
      .KKKKKA.' 'V LJ V' '.4KKKKK .
      .4D KKKKKKKKA.' LJ '''.4KKKKKKK FA.
      <QDD ++++++ GFD>
      'VD KKKKKKKK' .. LJ ..'KKKKKKK FV
      ogley VKKKKKK' D;4 LJ K. .KKKKKKK FV
      DB no configura KKA. 'KKKK LJ KKKK' .4KK
      el siguiente KKA. 'KKKK LJ KKKK' .4KK
      KKSA. VKK LJ KKKV .4SSKK
      <><><><>
      mariadb.com/Ma'MKKM' mariadb_repo_setup | sudo
      ...
[root@master01 ~]#
```

Luego probamos conectividad entre las máquinas en la red de replicación haciendo ping desde master01. Como podemos apreciar, la red virtual con el nombre **cluster** provee conectividad a la red 10.10.10.0/24 como si de una red física aparte se tratara, y son las direcciones IP que utilizaremos para configurar la replicación multimaster en MariaDB

```
[root@master01 ~]# ping 10.10.10.10
PING 10.10.10.10 (10.10.10.10) 56(84) bytes of data.
64 bytes from 10.10.10.10: icmp_seq=1 ttl=64 time=0.070 ms
64 bytes from 10.10.10.10: icmp_seq=2 ttl=64 time=0.075 ms
64 bytes from 10.10.10.10: icmp_seq=3 ttl=64 time=0.086 ms
64 bytes from 10.10.10.10: icmp_seq=4 ttl=64 time=0.109 ms
64 bytes from 10.10.10.10: icmp_seq=5 ttl=64 time=0.097 ms
^C
--- 10.10.10.10 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3998ms
rtt min/avg/max/mdev = 0.070/0.087/0.109/0.016 ms
[root@master01 ~]# ping 10.10.10.20
PING 10.10.10.20 (10.10.10.20) 56(84) bytes of data.
64 bytes from 10.10.10.20: icmp_seq=1 ttl=64 time=0.853 ms
64 bytes from 10.10.10.20: icmp_seq=2 ttl=64 time=0.975 ms
64 bytes from 10.10.10.20: icmp_seq=3 ttl=64 time=0.720 ms
64 bytes from 10.10.10.20: icmp_seq=4 ttl=64 time=0.726 ms
64 bytes from 10.10.10.20: icmp_seq=5 ttl=64 time=0.756 ms
^C
--- 10.10.10.20 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 0.720/0.806/0.975/0.097 ms
[root@master01 ~]# ping 10.10.10.30
PING 10.10.10.30 (10.10.10.30) 56(84) bytes of data.
64 bytes from 10.10.10.30: icmp_seq=1 ttl=64 time=0.780 ms
64 bytes from 10.10.10.30: icmp_seq=2 ttl=64 time=0.722 ms
64 bytes from 10.10.10.30: icmp_seq=3 ttl=64 time=0.724 ms
64 bytes from 10.10.10.30: icmp_seq=4 ttl=64 time=0.739 ms
64 bytes from 10.10.10.30: icmp_seq=5 ttl=64 time=0.734 ms
^C
--- 10.10.10.30 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 0.722/0.739/0.780/0.040 ms
[root@master01 ~]#
```



Y tenemos nuestros tres Nodos de MariaDB, dos de ellos esperando a ser configurados (con un aroma Mac parecido a Güindous 95 ;).

### Paso 3: Instalando MariaDB en cada nodo por medio de dnf/yum

Ahora seguiremos con la instalación de MariaDB desde sus repositorios. Como siempre, vamos a Google y buscamos la documentación oficial de MariaDB, y al revisarla, los desarrolladores de MariaDB nos dan un script shell que nos configurará los repositorios en esta URL:

<https://mariadb.com/kb/en/mariadb-package-repository-setup-and-use/>.

El script de configuración de los repositorios se puede correr con el siguiente comando:

```
# curl -LsS https://downloads.mariadb.com/MariaDB/mariadb_repo_setup | sudo bash
```

y esta debería ser la salida en la consola con el comando ejecutado sin problemas

```
*          root@master01:~*
+ *      root@master01:~*          Home: scp
[root@master01 ~]# curl -LsS https://downloads.mariadb.com/MariaDB/mariadb_re
po_setup | sudo bash
[info] Repository file successfully written to /etc/yum.repos.d/mariadb.repo
[info] Adding trusted package signing keys...
[info] Successfully added trusted package signing keys
[root@master01 ~]#
```

---

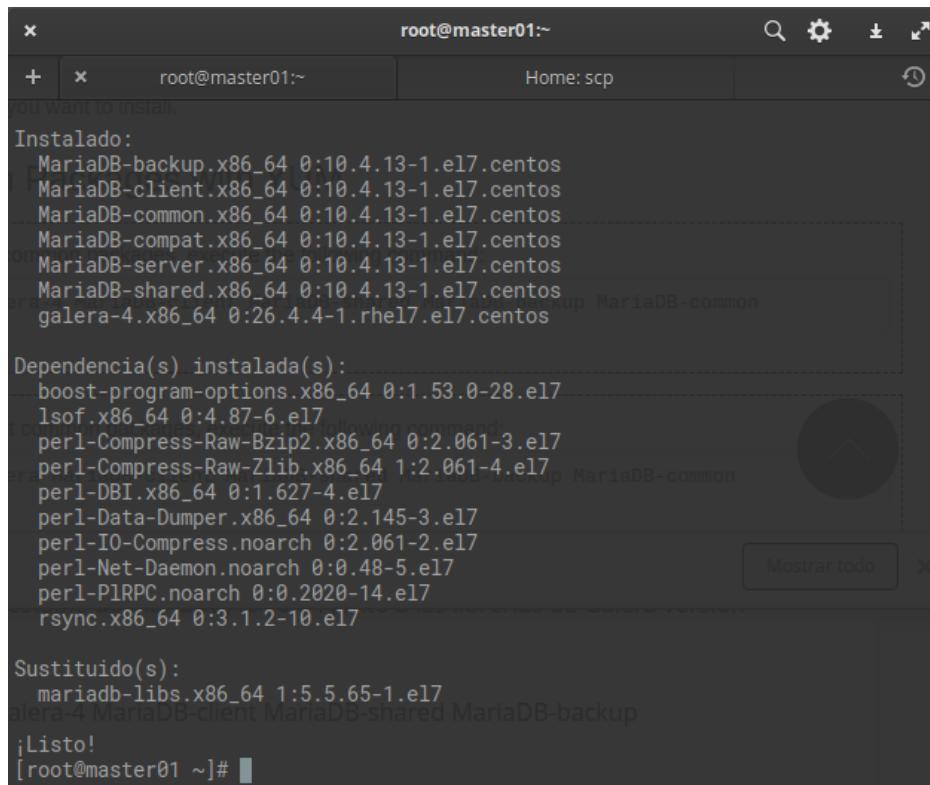
Luego actualizamos los repositorios de MariaDB con yum

```
# yum -y update
```

Ahora Instalaremos la última versión estable de MariaDB 10.4, junto a las librerías de Galera versión 4 utilizando yum con el siguiente comando copiado desde la página.

```
# yum -y install MariaDB-server MariaDB-client galera-4
```

y la salida exitosa del comando debería terminar como se muestra en la siguiente figura:



```
you want to install.

Instalado:
  MariaDB-backup.x86_64 0:10.4.13-1.el7.centos
  MariaDB-client.x86_64 0:10.4.13-1.el7.centos
  MariaDB-common.x86_64 0:10.4.13-1.el7.centos
  MariaDB-compat.x86_64 0:10.4.13-1.el7.centos
  MariaDB-server.x86_64 0:10.4.13-1.el7.centos
  MariaDB-shared.x86_64 0:10.4.13-1.el7.centos
  galera-4.x86_64 0:26.4.4-1.rhel7.el7.centos

Dependencia(s) instalada(s):
  boost-program-options.x86_64 0:1.53.0-28.el7
  lsof.x86_64 0:4.87-6.el7
  perl-Compress-Raw-Bzip2.x86_64 0:2.061-3.el7
  perl-Compress-Raw-Zlib.x86_64 1:2.061-4.el7
  perl-DBI.x86_64 0:1.627-4.el7
  perl-Data-Dumper.x86_64 0:2.145-3.el7
  perl-IO-Compress.noarch 0:2.061-2.el7
  perl-Net-Daemon.noarch 0:0.48-5.el7
  perl-PlRPC.noarch 0:0.2020-14.el7
  rsync.x86_64 0:3.1.2-10.el7

Sustituido(s):
  mariadb-libs.x86_64 1:5.5.65-1.el7
galera-4 MariaDB-client MariaDB-shared MariaDB-backup

¡Listo!
[root@master01 ~]#
```

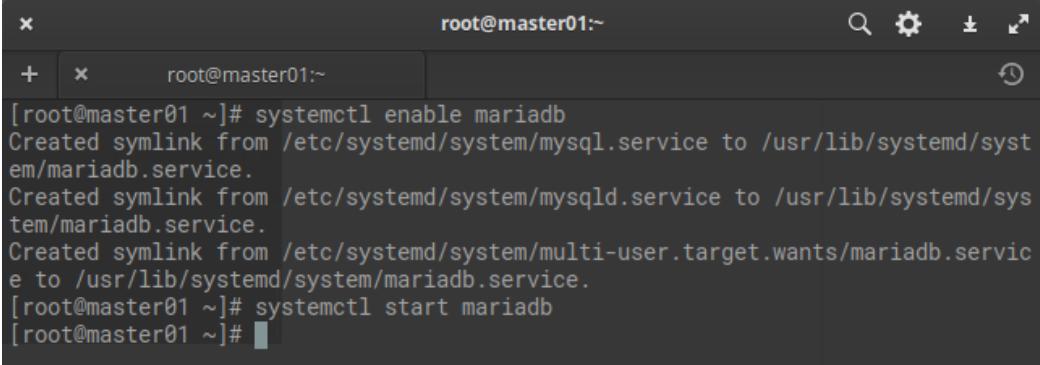
y arrancamos el servicio de MariaDB

```
# systemctl enable mariadb
```

y luego configuramos MariaDB para que arranque como servicio junto con el SO.

```
# systemctl start mariadb
```

...la salida de ambos comandos debería ser como se muestra en la siguiente figura



```
[root@master01 ~]# systemctl enable mariadb
Created symlink from /etc/systemd/system/mysql.service to /usr/lib/systemd/system/mariadb.service.
Created symlink from /etc/systemd/system/mysqld.service to /usr/lib/systemd/system/mariadb.service.
Created symlink from /etc/systemd/system/multi-user.target.wants/mariadb.service to /usr/lib/systemd/system/mariadb.service.
[root@master01 ~]# systemctl start mariadb
[root@master01 ~]#
```

Luego fortalecemos la seguridad de la BBDD con la app destinada para ello

```
# mysql_secure_installation
```

...y su salida, y mis respuestas, son:

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!
```

```
In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.
```

```
Enter current password for root (enter for none): [pulsé Enter]
OK, successfully used password, moving on...
```

```
Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.
```

```
You already have your root account protected, so you can safely answer 'n'.
```

```
Switch to unix_socket authentication [Y/n] y
Enabled successfully!
Reloading privilege tables..
... Success!
```

```
You already have your root account protected, so you can safely answer 'n'.
```

```
Change the root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!
```

```
By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.
```

```
Remove anonymous users? [Y/n] n
... skipping.
```

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

```
Disallow root login remotely? [Y/n] y
... Success!
```

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

```
Remove test database and access to it? [Y/n] n
... skipping.
```

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

```
Reload privilege tables now? [Y/n] y
... Success!
```

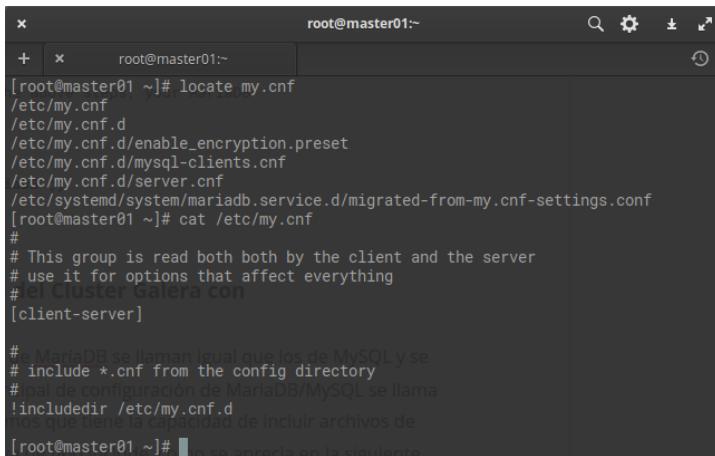
Cleaning up...

All done! If you've completed all of the above steps, your MariaDB installation should now be secure.

Thanks for using MariaDB!

## Paso 4: Configuración del Nodo principal del Cluster Galera con MariaDB

Por defecto los archivos de configuración de MariaDB se llaman igual que los de MySQL y se ubican en el directorio /etc. el archivo principal de configuración de MariaDB/MySQL se llama /etc/my.cnf, y si vemos su contenido veremos que tiene la capacidad de incluir archivos de configuración adicionales con una directiva de tipo include, específicamente la directiva !includedir /etc/my.cnf.d , haciendo que MariaDB tome todos los archivos con configuraciones desagregadas desde ese directorio como se aprecia en la siguiente figura:



```
root@master01:~#
[root@master01 ~]# locate my.cnf
/etc/my.cnf
/etc/my.cnf.d
/etc/my.cnf.d/enable_encryption.preset
/etc/my.cnf.d/mysql-clients.cnf
/etc/my.cnf.d/server.cnf
/etc/systemd/system/mariadb.service.d/migrated-from-my.cnf-settings.conf
[root@master01 ~]# cat /etc/my.cnf
#
# This group is read both by the client and the server
# use it for options that affect everything
#[galera]
[client-server]

# MariaDB se llaman igual que los de MySQL y se
# include *.cnf from the config directory
#pal de configuración de MariaDB/MySQL se llama
!includedir /etc/my.cnf.d
#os que tiene la capacidad de incluir archivos de
[root@master01 ~]#
```

---

para configurar los nodos de Galera Cluster, el archivo que nos interesa es  
`/etc/my.cnf.d/server.cnf`

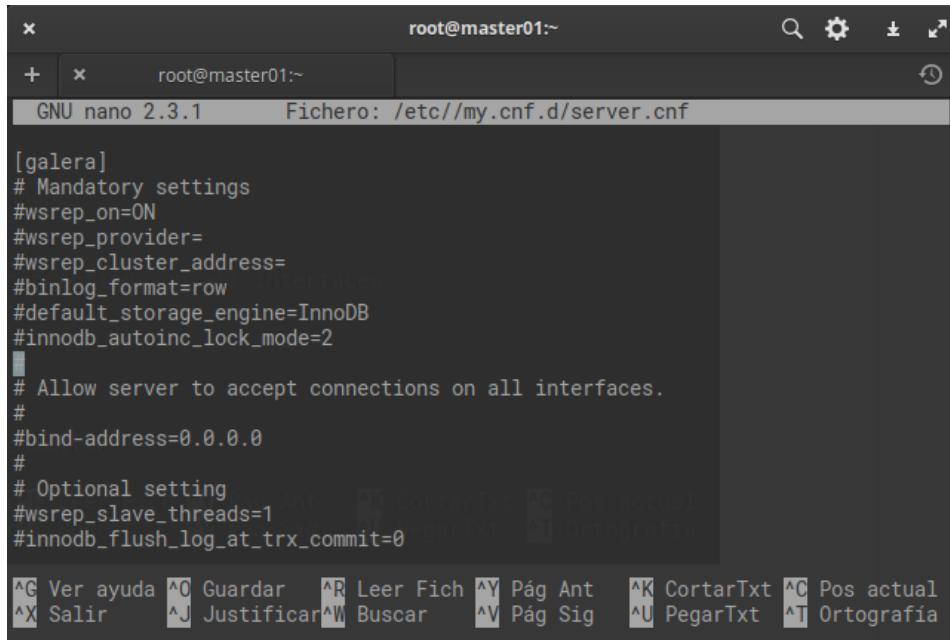
**Nota:** Para asegurarnos de poder volver atrás si es que nos pasa algo con la configuración, creamos una copia de respaldo del archivo `server.cnf`

```
# cp /etc/my.cnf.d/server.cnf /etc/my.cnf.d/server.cnf.bk
```

y luego editamos el archivo `server.cnf`

```
# nano /etc/my.cnf.d/server.cnf
```

...y prestaremos atención al apartado **[galera]** y los parámetros a configurar:



```
root@master01:~ Fichero: /etc/my.cnf.d/server.cnf
[galera]
# Mandatory settings
#wsrep_on=ON
#wsrep_provider=
#wsrep_cluster_address=
#binlog_format=row
#default_storage_engine=InnoDB
#innodb_autoinc_lock_mode=2
#
# Allow server to accept connections on all interfaces.
#
#bind-address=0.0.0.0
#
# Optional setting
#wsrep_slave_threads=1
#innodb_flush_log_at_trx_commit=0
```

Por defecto, Galera en modo cluster viene deshabilitado (`#wsrep_on=ON` comentado), por lo cual debemos descomentar (quitar el carácter `#`) de varias líneas de su configuración, y configurar apoyados por la documentación de Galera:

<https://galeracluster.com/library/training/tutorials/configuration.html>

---

...y procederemos a configurar las directivas necesarias en el nodo principal.

```
[galera]
# Mandatory settings
wsrep_on=ON
wsrep_provider=/usr/lib64/galera-4/libgalera_smm.so
wsrep_cluster_name="galeracluster"
wsrep_cluster_address="gcomm://10.10.10.10,10.10.10.20,10.10.10.30"
binlog_format=row
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
wsrep_node_address="10.10.10.10"
wsrep_node_name="master01"
wsrep_sst_method=rsync
#
# Allow server to accept connections on all interfaces.
#
bind-address=0.0.0.0
#
# Optional setting
#wsrep_slave_threads=1
innodb_flush_log_at_trx_commit=0
```

## Paso 5: Configurar los demás nodos del Cluster

La configuración del archivo `server.cnf` debe repetirse en los demás nodos, conservando el nombre del cluster para que funcione, pero con un nombre de nodo diferente y una dirección IP correspondiente al nodo nuevo, con direcciones IP de la red de replicación del rango **10.10.10.0/24**

Cambiar en el Nodo 2

```
#Nodo 2
wsrep_node_address="10.10.10.20"
wsrep_node_name="master02"
```

Cambiar en el Nodo 3

```
#Nodo 3
wsrep_node_address="10.10.10.30"
wsrep_node_name="master03"
```

---

## Paso 6: Configurando el Firewall de CentOS Linux en todos los nodos

**Nota:** Si el servidor está preconfigurado por un tercero con reglas de cortafuegos local, debemos abrir los puertos necesarios para Galera y María DB en cada nodo para que haya comunicación en el cluster.

CentOS usa la utilidad **firewall-cmd**, los siguientes comandos abrirán puertos y acceso a las direcciones IP de la red de replicación para los servicios de MariaDB y de Galera:

```
# firewall-cmd --permanent --zone=public --add-port=3306/tcp  
# firewall-cmd --permanent --zone=public --add-port=4567/tcp  
# firewall-cmd --permanent --zone=public --add-port=4568/tcp  
# firewall-cmd --permanent --zone=public --add-port=4444/tcp  
# firewall-cmd --permanent --zone=public --add-port=4567/udp
```

Ahora hay que agregar cada nodo a la zona **public** ejecutando los siguientes comandos

```
# firewall-cmd --permanent --zone=public --add-source=10.10.10.10/32  
# firewall-cmd --permanent --zone=public --add-source=10.10.10.20/32  
# firewall-cmd --permanent --zone=public --add-source=10.10.10.30/32
```

y luego aplicamos todas estas reglas de **firewall-cmd**:

```
# firewall-cmd --reload
```

## Paso 7: Configuración de SELinux

Crearemos una política de SELinux que permitirá a todos los nodos en el clúster ser capaces de comunicarse entre sí y realizar las operaciones de replicación.

SELinux es un módulo del kernel que mejora la seguridad del sistema operativo con soporte para control de acceso y políticas de control de acceso prioritario. Viene habilitado por defecto en CentOS 7 y restringe al demonio de MariaDB para realizar muchas actividades necesarias, por lo que crearemos una política realizando una serie de actividades necesarias seteando SELinux en modo permisivo para MariaDB. Luego crearemos una política a partir de eventos en los logs, para

---

finalmente volver a dejar a SELinux en modo seguro (enforcing) una vez que la política definitiva se haya instalado exitosamente.

Primero permitiremos acceso a los puertos necesarios para el cluster Galera ejecutando los siguientes comandos en cada nodo:

```
# semanage port -a -t mysqld_port_t -p tcp 4567  
# semanage port -a -t mysqld_port_t -p udp 4567  
# semanage port -a -t mysqld_port_t -p tcp 4568  
# semanage port -a -t mysqld_port_t -p tcp 4444
```

**Nota:** Es posible que salga el error de tipo **ValueError** cuando se de acceso a algunos de estos puertos, debido a que el estatus de ese puerto en SELinux ya esté configurado, sin afectar el desarrollo de todo este proceso.

En dichos comandos utilizamos la herramienta **semanage** de SELinux con la opción **-a** para agregar puertos específicos que ignoren la actividad del servidor de base de datos.

Luego ejecutaremos el siguiente comando en cada nodo para dejar el dominio de SELinux sobre MySQL SELinux en modo permisivo de manera temporal, para poder trabajar y generar registros de actividad de MariaDB y Galera y así enseñar a SELinux que dicha actividad es normal en cada servidor.

```
# semanage permissive -a mysqld_t
```

Este comando puede que demore un poco en completarse y no arroja salida alguna en la terminal.

Luego, debemos detener el servidor de bbdd en todos los nodos para arrancarlo nuevamente en el nodo primario en modo bootstrap con las políticas de SELinux compartidas en los demás nodos del cluster. Para realizar esto, ejecutaremos el siguiente comando en cada nodo:

```
# systemctl stop mariadb
```

---

Ahora, arrancaremos el primer nodo del cluster en modo **bootstrap** para generar eventos de comunicación entre los nodos que serán añadidas a una política segura y permanente de SELinux. En el nodo primario del cluster arrancaremos MariaDB en modo bootstrap ejecutando:

```
# galera_new_cluster
```

**Nota:** Si no hay errores, no se mostrará ninguna salida al finalizar el inicio de MariaDB en modo bootstrap

Luego crearemos una base de datos en el primer nodo junto a una tabla con el objetivo específico de generar logs de eventos SST de Galera entre los nodos ejecutando el siguiente comando en el nodo primario:

```
# mysql -u root -p
```

...y una vez en la consola de mysql escribimos la siguiente query compuesta:

```
MariaDB [(none)]> CREATE DATABASE selinux;  
CREATE TABLE selinux.selinux_policy (id INT NOT NULL AUTO_INCREMENT,  
PRIMARY KEY(id));  
INSERT INTO selinux.selinux_policy VALUES ();
```

y una vez ejecutada la consulta compuesta abandonamos la consola de mysql con:

```
MariaDB [(none)]> quit;
```

Luego arrancamos el segundo nodo:

```
# systemctl start mariadb
```

...y después arrancamos el tercer nodo:

```
# systemctl start mariadb
```

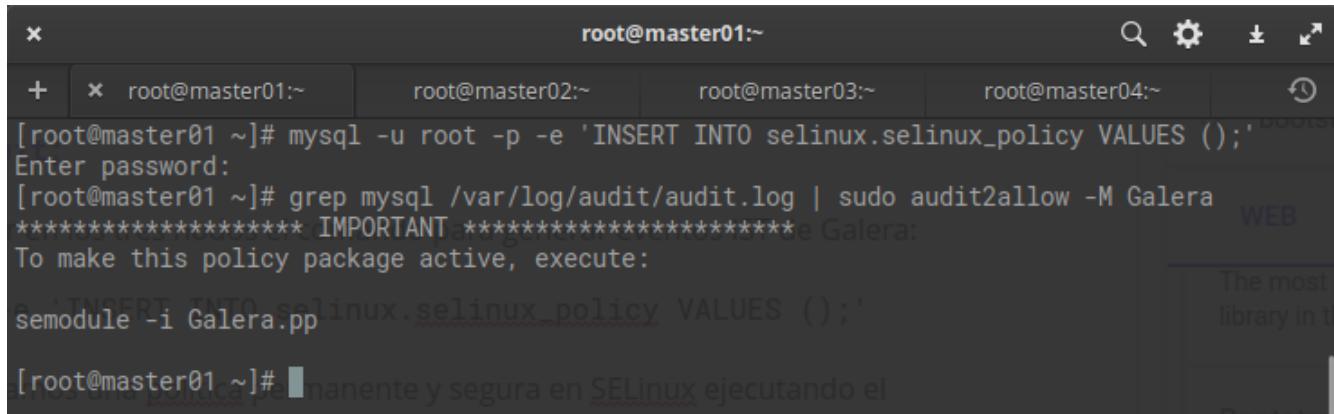
Con todos los nodos operando debemos ejecutar en cada uno el siguiente comando para generar actividades en la BBDD y así generar eventos IST de Galera:

```
# mysql -u root -p -e 'INSERT INTO selinux.selinux_policy VALUES ()';
```

y ahora creamos y habilitamos una política permanente y segura en SELinux ejecutando el siguiente comando en los tres nodos:

```
# grep mysql /var/log/audit/audit.log | sudo audit2allow -M Galera
```

y se nos muestra la salida con una instrucción para crear la política de SELinux...



The screenshot shows a terminal window with four tabs at the top: root@master01:~, root@master02:~, root@master03:~, and root@master04:~. The main pane displays the following command and its output:

```
[root@master01 ~]# mysql -u root -p -e 'INSERT INTO selinux.selinux_policy VALUES ();'
Enter password:
[root@master01 ~]# grep mysql /var/log/audit/audit.log | sudo audit2allow -M Galera
*****IMPORTANT***** To make this policy package active, execute:
semodule -i Galera.pp
[root@master01 ~]#
```

A tooltip "The most library in t" is visible on the right side of the window.

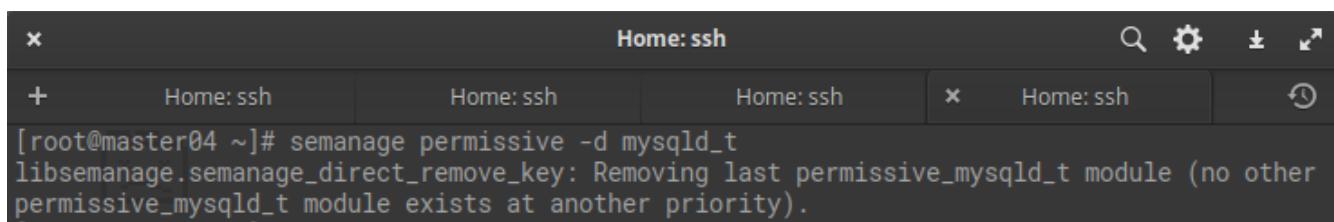
Instrucción que ejecutamos en cada nodo:

```
# semodule -i Galera.pp
```

Ahora que la nueva política de SELinux está activa con conocimiento de las actividades de MariaDB con Galera, deshabilitamos el modo permisivo para MariaDB en los tres nodos, dejando el servidor con SELinux correctamente configurado

```
# semanage permissive -d mysqld_t
```

y se nos muestra la siguiente salida:



The screenshot shows a terminal window titled "Home: ssh" with four tabs at the top: Home: ssh, Home: ssh, Home: ssh, and Home: ssh. The main pane displays the following command and its output:

```
[root@master04 ~]# semanage permissive -d mysqld_t
libsemanage.semanage_direct_remove_key: Removing last permissive_mysqld_t module (no other permissive_mysqld_t module exists at another priority).
```

## Paso 8: Arrancando el Clúster

Lo primero que debemos hacer es detener MariaDB en los tres nodos para poder levantar el Cluster con todo lo anterior configurado., y debe realizarse en un orden adecuado para asegurar que el nodo Primario contiene todas las transacciones del Cluster y no tenga problemas de pérdidas de secuencia y otros dolores de cabeza :)

1. Apagamos el tercer nodo:

```
# systemctl stop mariadb
```

2. Apagamos el segundo nodo:

```
# systemctl stop mariadb
```

3. Apagamos el primer nodo:

```
# systemctl stop mariadb
```

Para arrancar el cluster de forma correcta, encendemos el primer nodo con un comando especial que enciende MAriaDB en modo bootstrap para que no busque otros nodos, con el comando:

```
# galera_new_cluster
```

**Nota:** Si no hay errores, no se mostrará ninguna salida al finalizar el inicio de MariaDB en modo bootstrap

y para verificar su correcto funcionamiento, consultamos la tabla de estado de cluster en MariaDB con el comando;

```
# mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
```

The screenshot shows a terminal window with three tabs: 'root@master01:~', 'root@master02:~', and 'root@master03:~'. The 'root@master01' tab contains the command 'galera\_new\_cluster' followed by the MySQL status output for 'wsrep\_cluster\_size'. The output shows a single row with 'wsrep\_cluster\_size' having a value of 1.

Variable_name	Value	Variable_name	Value
wsrep_cluster_size	1	rep_cluster_size	1

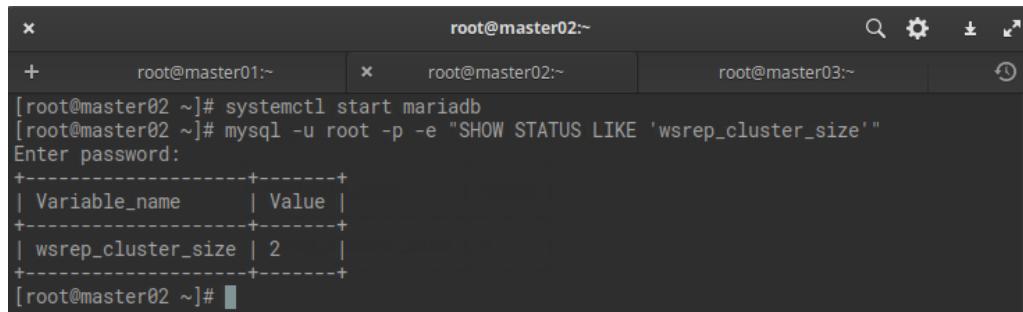
y la salida de la consulta contenida en el comando nos debe mostrar que el cluster tiene sólo un nodo.

Arrancamos el segundo nodo con el comando:

```
# systemctl start mariadb
```

y para verificar su correcto funcionamiento, consultamos la tabla de estado de cluster en MariaDB con el comando:

```
# mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
```



A terminal window titled 'root@master02:~'. It shows three tabs: 'root@master01:~', 'root@master02:~' (active), and 'root@master03:~'. The active tab displays the command 'mysql -u root -p -e "SHOW STATUS LIKE 'wsrep\_cluster\_size''' and its output. The output shows a single row in a table with 'wsrep\_cluster\_size' having a value of 2.

Variable_name	Value
wsrep_cluster_size	2

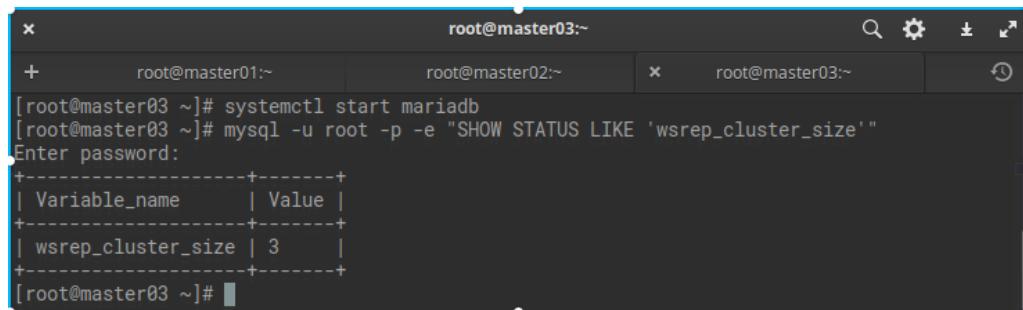
Y podemos apreciar que el cluster ahora contiene dos nodos.

Finalmente arrancamos el tercer nodo con el comando:

```
# systemctl start mariadb
```

y para verificar su correcto funcionamiento, consultamos la tabla de estado de cluster en MariaDB con el comando;

```
# mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
```



A terminal window titled 'root@master03:~'. It shows three tabs: 'root@master01:~', 'root@master02:~', and 'root@master03:~' (active). The active tab displays the command 'mysql -u root -p -e "SHOW STATUS LIKE 'wsrep\_cluster\_size''' and its output. The output shows a single row in a table with 'wsrep\_cluster\_size' having a value of 3.

Variable_name	Value
wsrep_cluster_size	3

Y ejecutamos nuevamente una consulta a las tablas de estado de Galera y vemos que el cluster contiene ahora tres nodos.

## Paso 9: Pruebas de carga y replicación del Cluster Galera con MariaDB

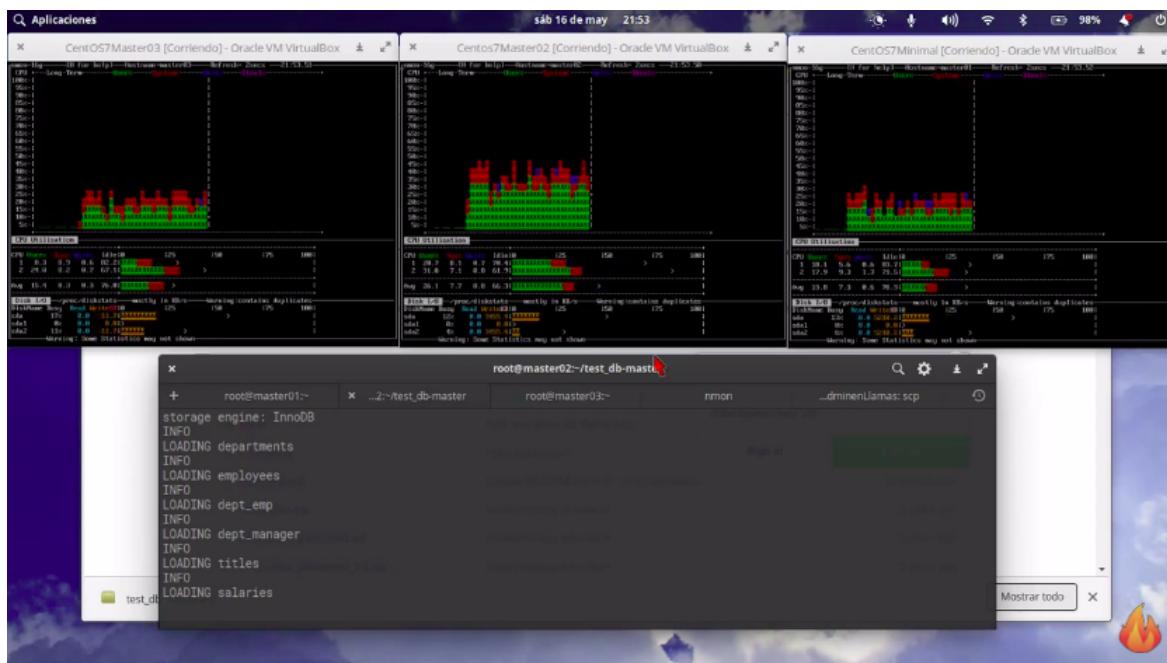
Cargaremos set de datos grande en el nodo 2 y veremos con nmon cómo se replican los demás nodos (1, y, 3). Para ello utilizaremos un set de datos de prueba, que encontré en internet con información falsa de empleados de una empresa, que está disponible públicamente en GitHub para hacer ejercicios de este tipo. La BBDD está disponible en

[https://github.com/datacharmer/test\\_db](https://github.com/datacharmer/test_db)

bajamos el archivo, lo enviamos al nodo 2 con scp y lo descomprimimos con unzip, se generará una carpeta con el set de datos con varios archivos SQL, y volcamos el archivo employes.sql sobre la bbdd del nodo 2 con el comando

```
# mysql -u root -p < employes.sql
```

y podemos ver en esta hermosa imagen como este gran set de datos se replica en los 3 nodos adicionales



y más hermoso aún el video de este proceso en caliente que lo pueden ver en el canal de youtube de SysAdmin en Llamas <https://www.youtube.com/watch?v=b3SENShzRLk>

Hasta una próxima entrega SysAdmins en Llamas...



#sysadminenllamas

Encuentra la última versión de este tutorial en

<https://www.sysadminenllamas.cl>