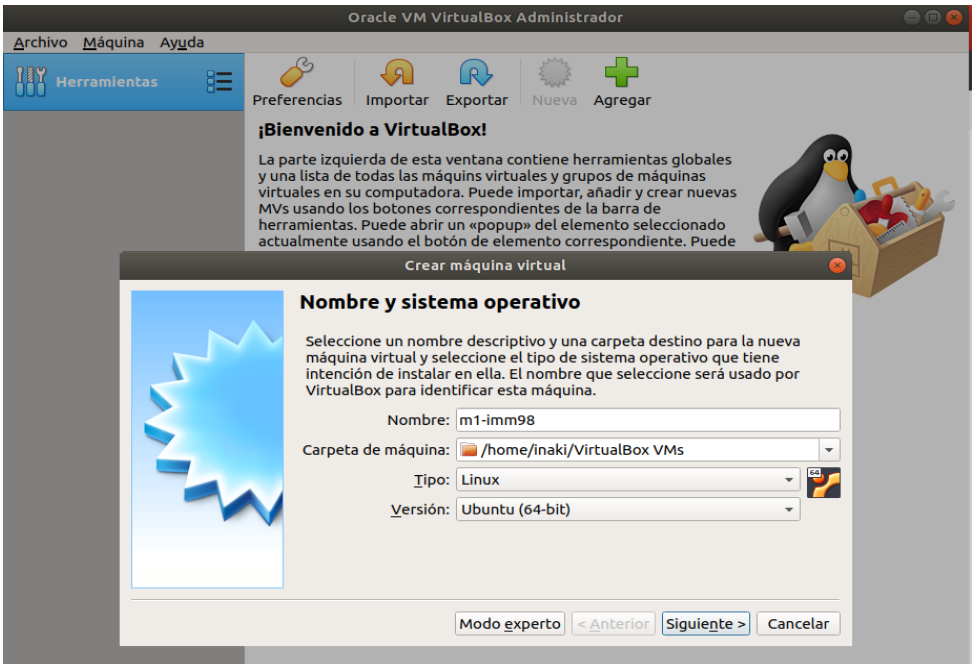
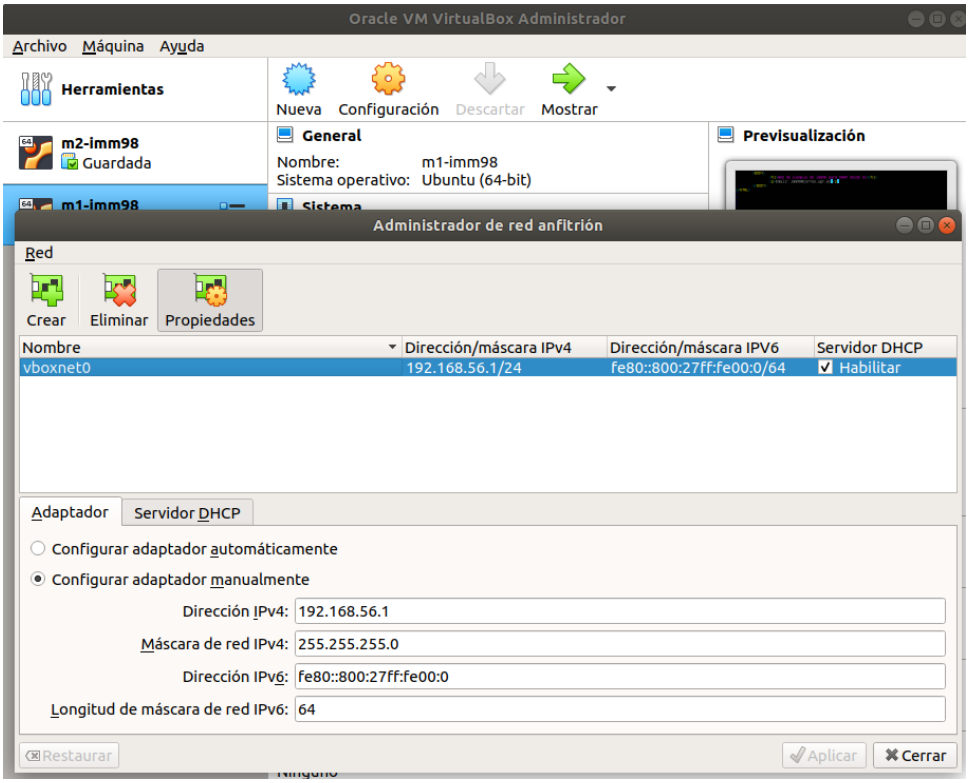
PRÁCTICA 1: PREPARACIÓN DE LAS HERRAMIENTAS

Iñaki Melguizo Marcos

Para la realización de la práctica he utilizado el software de virtualización VirtualBox y he creado dos máquinas virtuales m1-imm98 y m2-imm98 con idénticas características y con UbuntuServer18.04.5

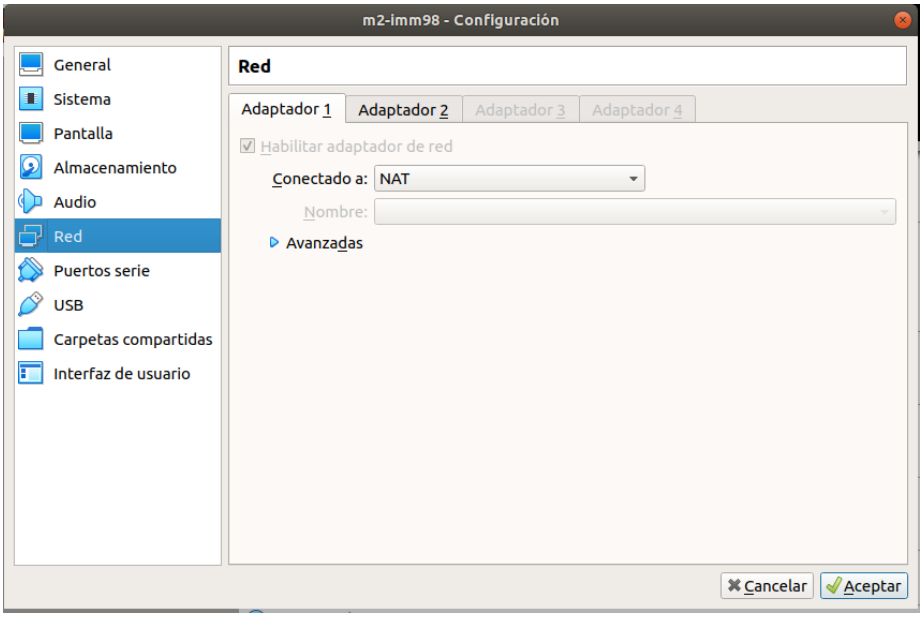


Para la existencia de comunicación y visibilidad entre ambas máquinas virtuales he creado una red sólo anfitrión con la siguiente configuración:

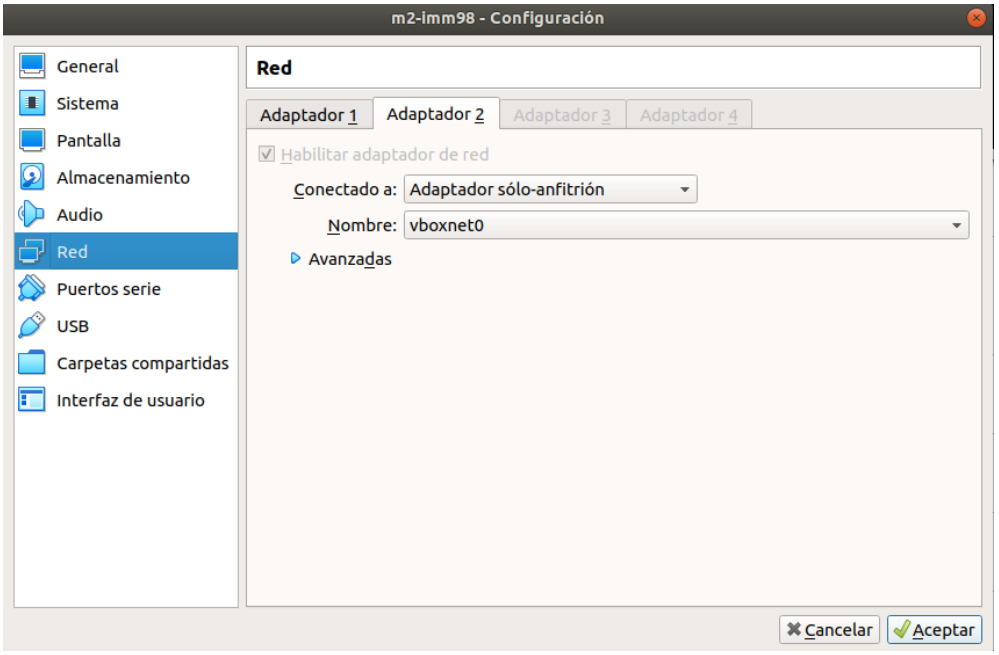


Por lo tanto tendremos dos adaptadores de red:

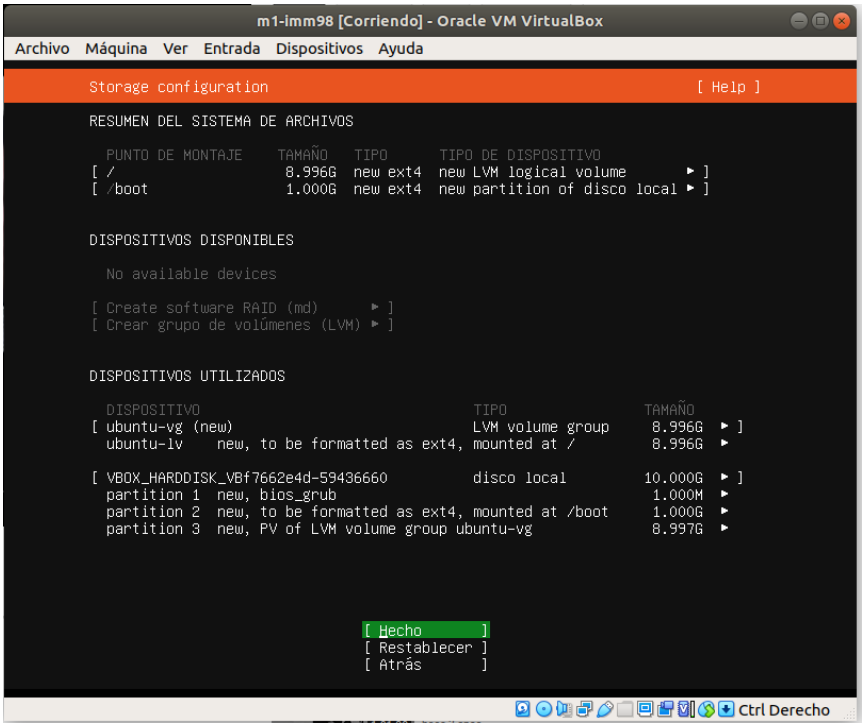
- Un adaptador de tipo **NAT** que permite comunicarnos con todas las máquinas virtuales a Internet.



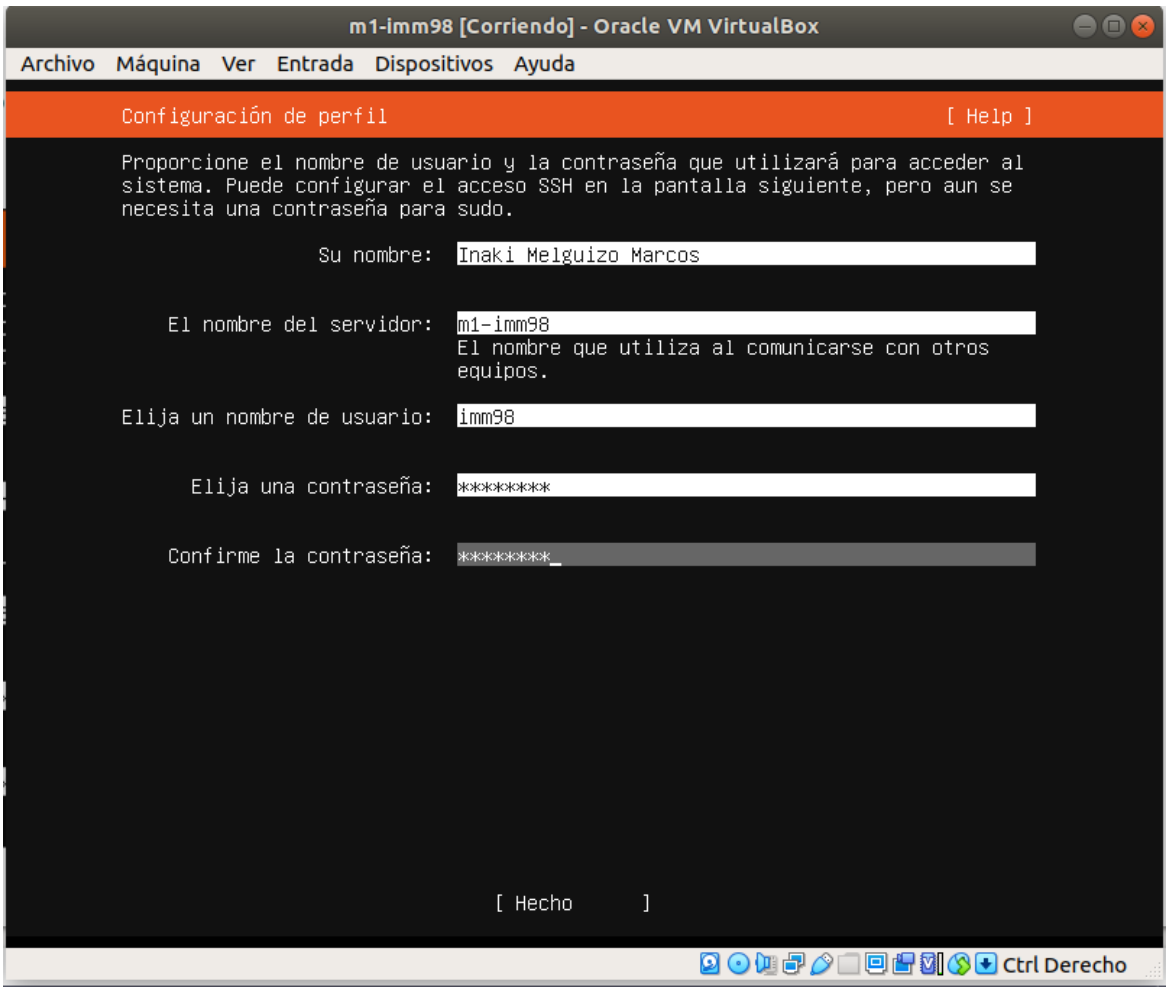
- Un adaptador de tipo **sólo-anfitrión** para poder establecer una comunicación entre las diferentes máquinas virtuales y entre las máquinas virtuales y el host como veremos más adelante.



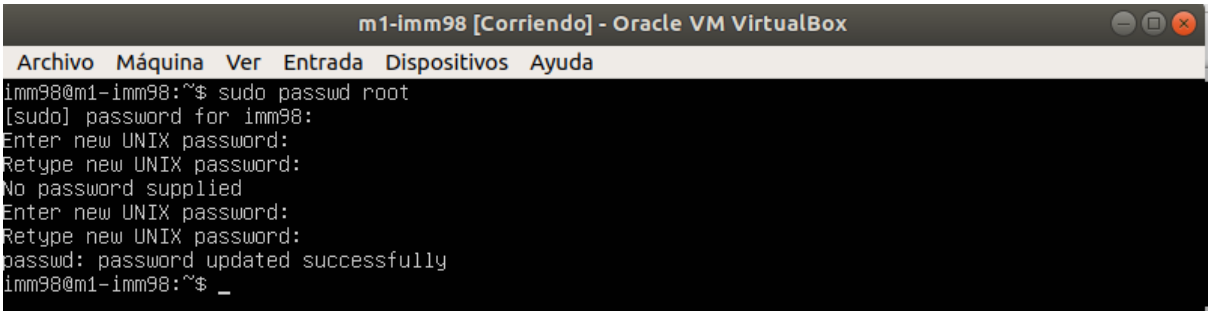
Ya podemos por tanto proceder a iniciar ambas máquinas virtuales y a instalar US en estas:



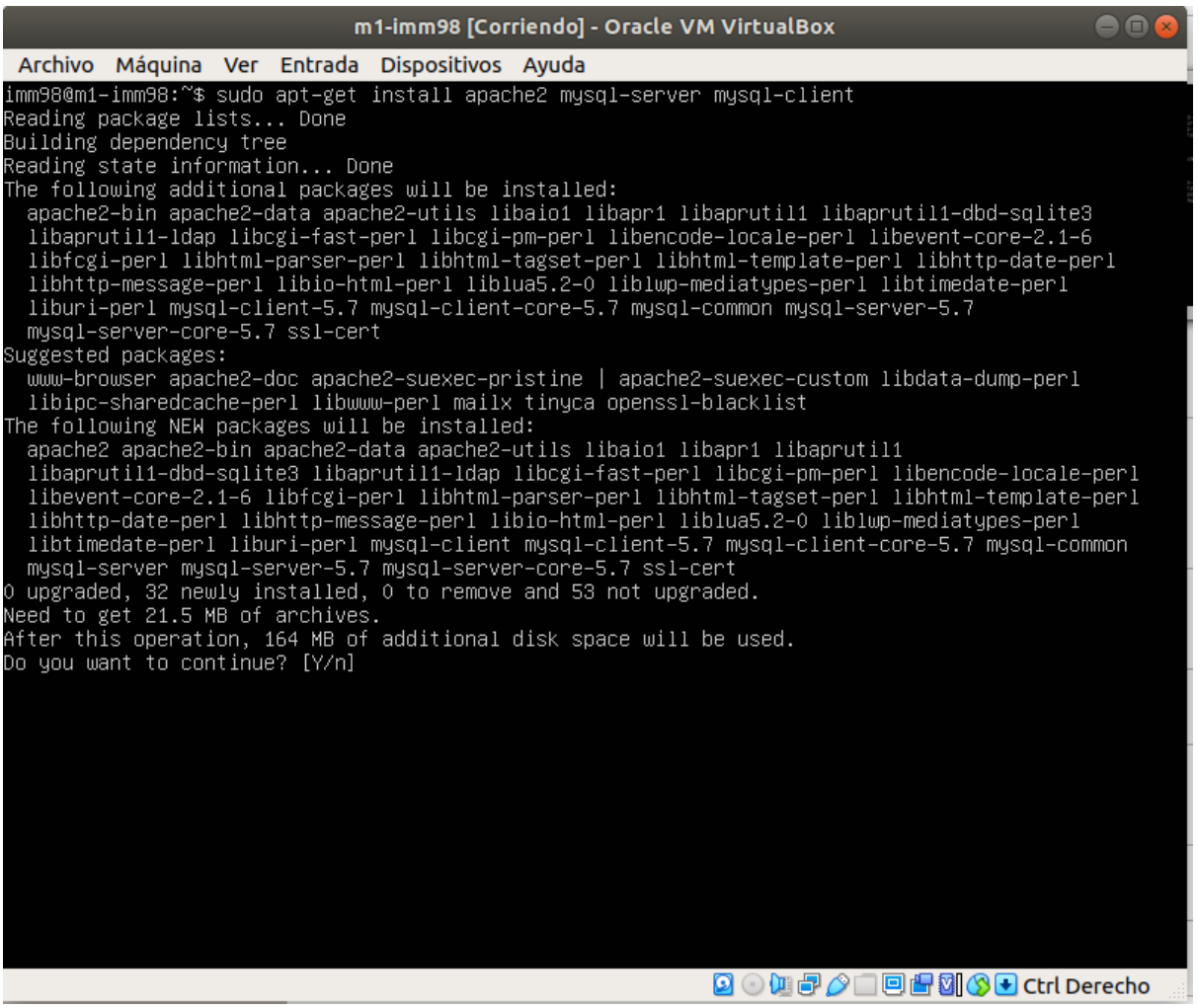
Introduzco como se indica en el guión de prácticas como usuario mi usuario de la UGR y como contraseña *Swap1234*



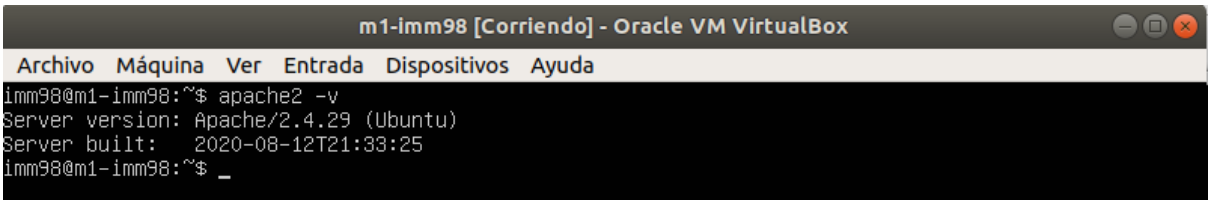
Una vez hemos hecho un *reboot* procedemos a la **activación de la cuenta de root** con la orden
sudo passwd root



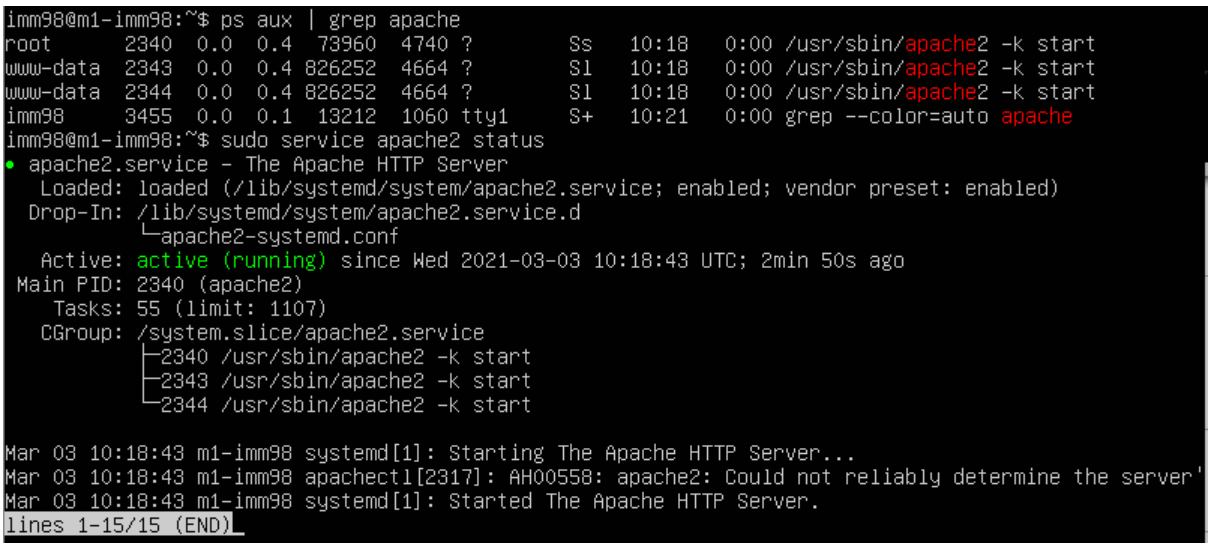
Durante el proceso de instalación no aparecía la opción “LAMP” por lo que para instalar Apache, PHP y MySQL ejecuto el siguiente comando:
sudo apt-get install apache2 mysql-server mysql-client



y tras la instalación vemos la versión del servidor:
apache2 -v



y vemos si está en ejecución:
ps aux | grep apache o *sudo service apache2 status*



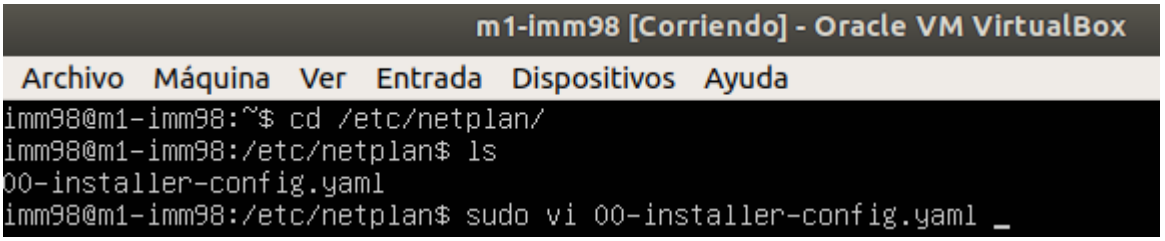
y vemos que efectivamente está en ejecución

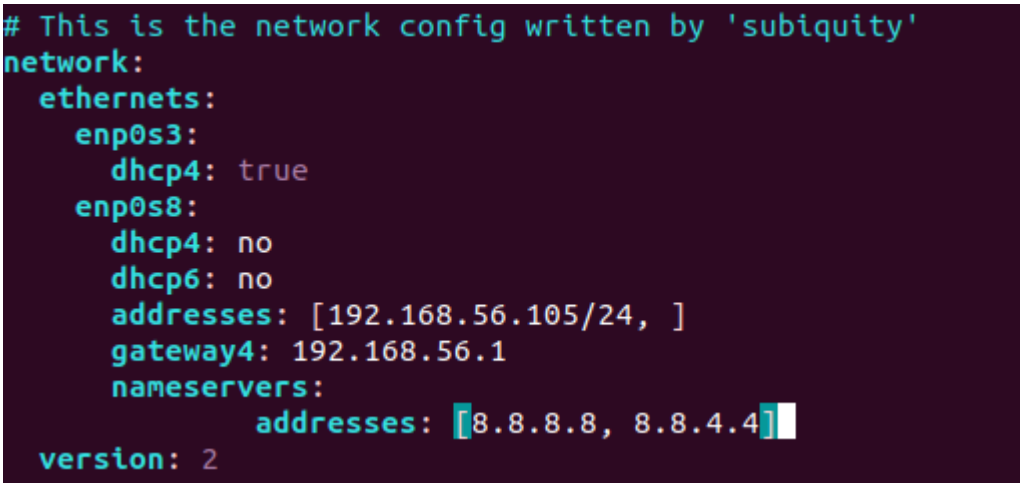
IP DE LAS DOS MÁQUINAS

Vamos primeramente a configurar las interfaces de red, vamos a configurar una

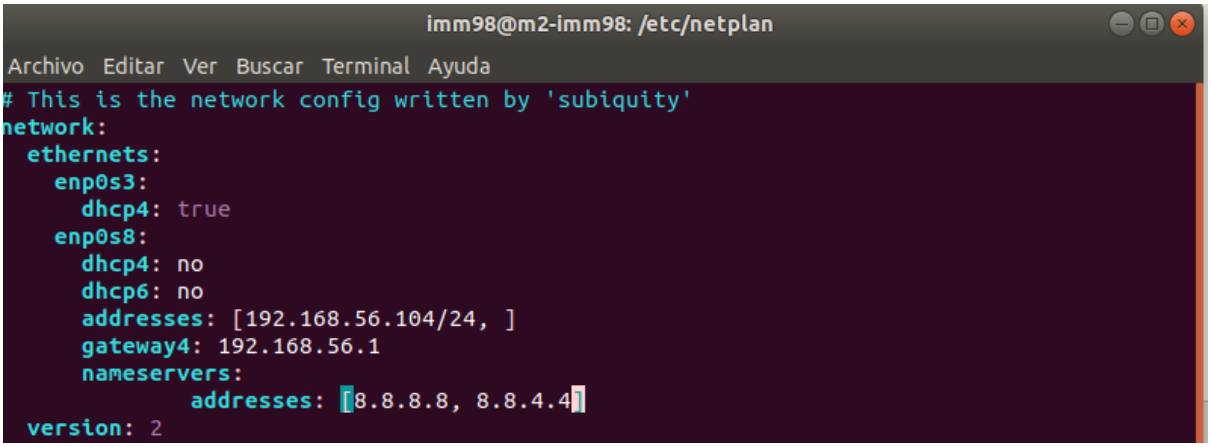
- IP estática para la interfaz de red Ethernet enp0s8
- una IP dinámica para la interfaz de red Ethernet enp0s3

Para ello tenemos que modificar el archivo **/etc/netplan/00-installer-config.yaml**. Primero vamos a configurar dicho fichero de la máquina virtual m1 y le vamos a asociar a la interfaz de red enp0s8 la IP 192.168.56.105:

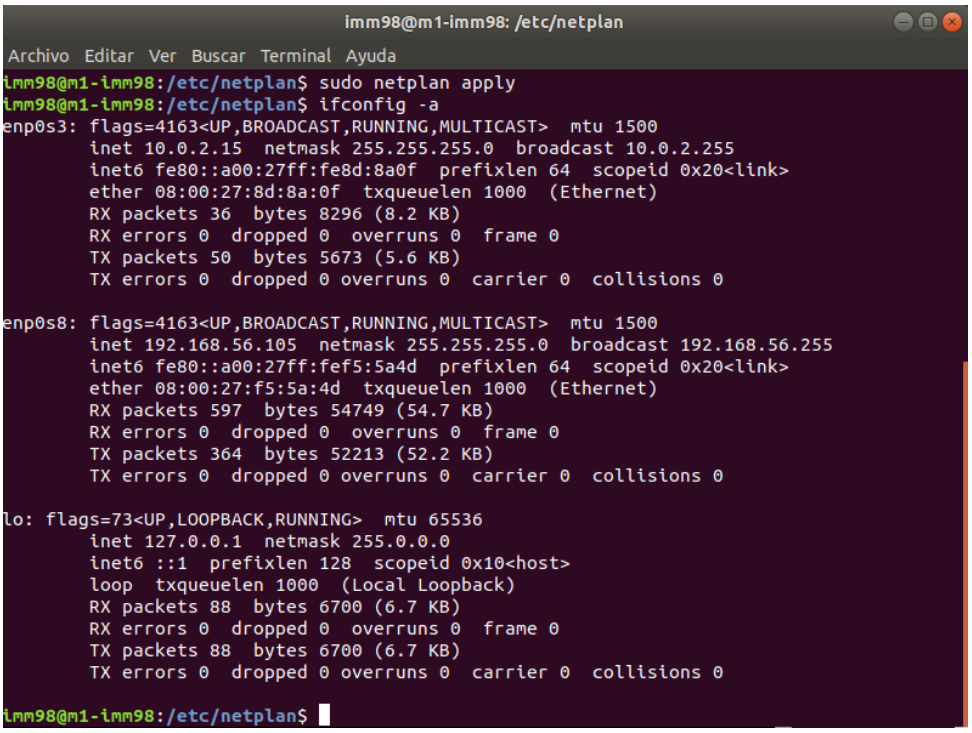




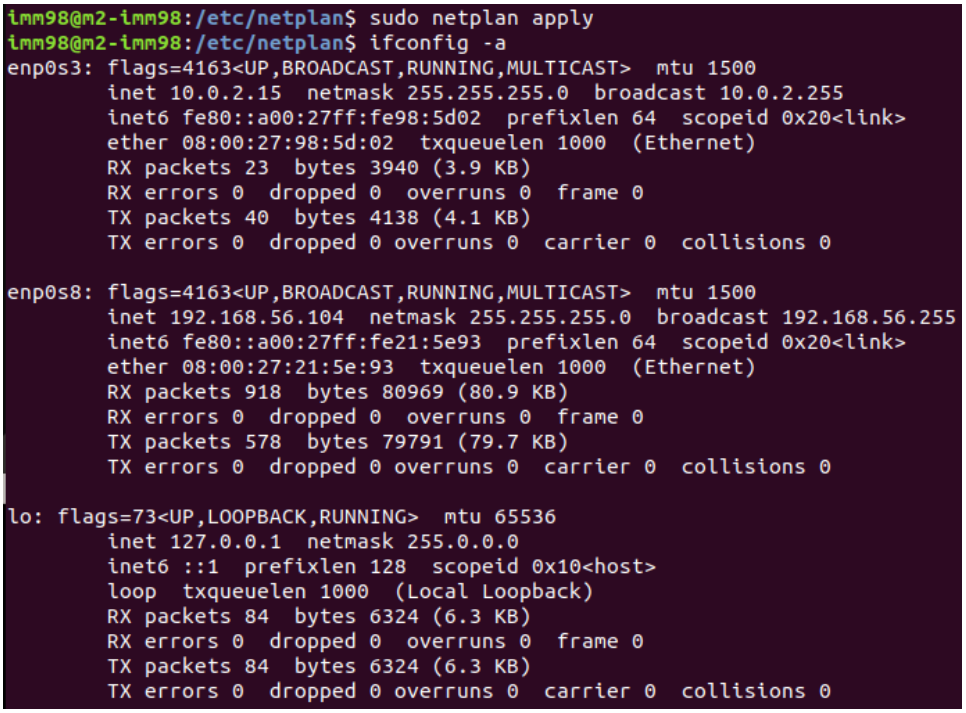
En la segunda máquina volvemos a hacer lo mismo pero asociándole a la interfaz de red enp0s8 la IP 192.168.56.104:



Ejecutamos la orden netplan apply para aplicar los cambios que hemos hecho en las interfaces de red y mediante el comando *ifconfig* vamos a ver la IP de las máquinas m1-imm98



que es la **192.168.56.105**
y de m2-imm98



que es la **192.168.56.104**, es decir:

m1-imm98	192.168.56.105
m2-imm98	192.168.56.104

Probamos a hacer ***ping*** para comprobar que ambas máquinas se “ven”:

- ping de m1 a m2

```
imm98@m1-imm98:~$ ping 192.168.56.104
PING 192.168.56.104 (192.168.56.104) 56(84) bytes of data.
64 bytes from 192.168.56.104: icmp_seq=1 ttl=64 time=0.926 ms
64 bytes from 192.168.56.104: icmp_seq=2 ttl=64 time=1.00 ms
64 bytes from 192.168.56.104: icmp_seq=3 ttl=64 time=0.945 ms
^C
--- 192.168.56.104 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.926/0.958/1.005/0.049 ms
imm98@m1-imm98:~$ _
```

- ping de m2 a m1

```
imm98@m2-imm98:~$ ping 192.168.56.105
PING 192.168.56.105 (192.168.56.105) 56(84) bytes of data.
64 bytes from 192.168.56.105: icmp_seq=1 ttl=64 time=1.65 ms
64 bytes from 192.168.56.105: icmp_seq=2 ttl=64 time=0.958 ms
64 bytes from 192.168.56.105: icmp_seq=3 ttl=64 time=0.930 ms
64 bytes from 192.168.56.105: icmp_seq=4 ttl=64 time=0.858 ms
^C
--- 192.168.56.105 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 0.858/1.099/1.652/0.322 ms
```

Una vez que ya hemos visto que ambas máquinas se pueden comunicar mediante ping vamos a proceder a modificar el puerto de escucha (por defecto el 80) de apache en una de las dos máquinas (m1) por el 8080. Para ello modificamos el archivo **/etc/apache2/ports.conf**

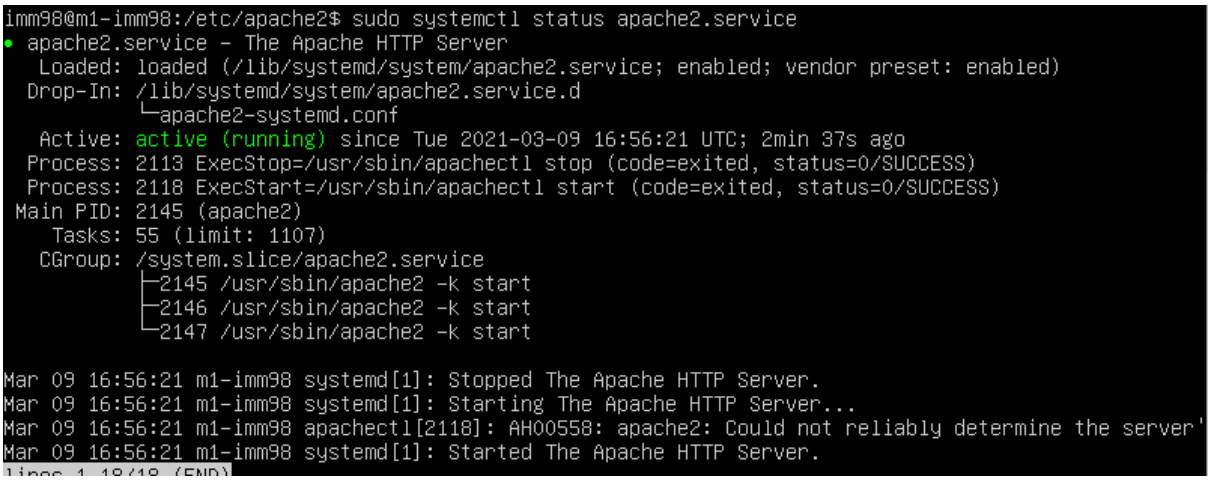
```
imm98@m1-imm98:/etc/apache2$ sudo vi ports.conf
```

```
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8080_

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Como hemos modificado el archivo de configuración reiniciamos el servicio de apache con systemctl restart apache2

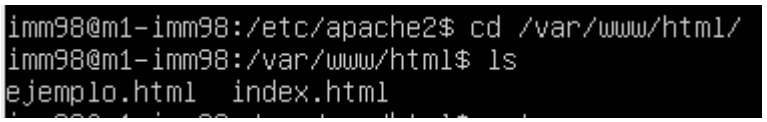
```
imm98@m1-imm98:/etc/apache2$ sudo systemctl restart apache2
```

Vemos que está activo el servicio de apache:

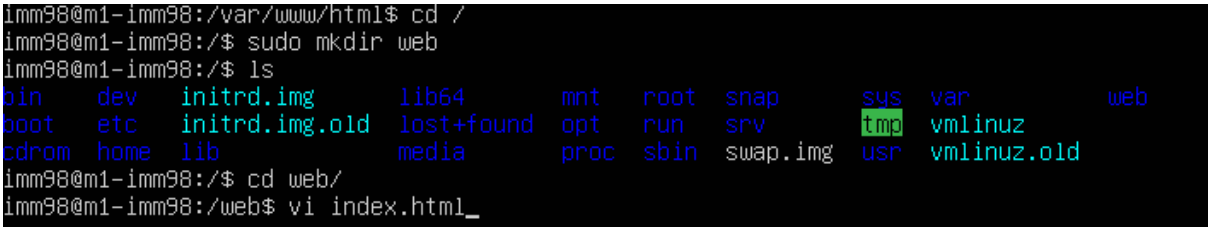


Vemos que está escuchando por el puerto 8080 con la opción
sudo netstat -anp | grep apache2

y tras crear un archivo html en el directorio **/var/www/html** llamado ejemplo.html veo que puedo acceder a él:



Ahora voy a proceder a crear un **directorio virtual.** Voy a crear un directorio llamado web/ y dentro de ese directorio voy a crear un fichero html

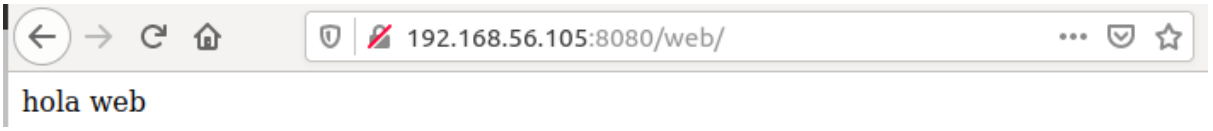


Ahora modifico el archivo de configuración que hay en el directorio **/etc/apache2/sites-enabled** y añado el directorio virtual que he creado asociándole un alias :

```
imm98@m1-imm98:/web$ cd /etc/apache2/sites-enabled/
imm98@m1-imm98:/etc/apache2/sites-enabled$ sudo vi 000-default.conf
```

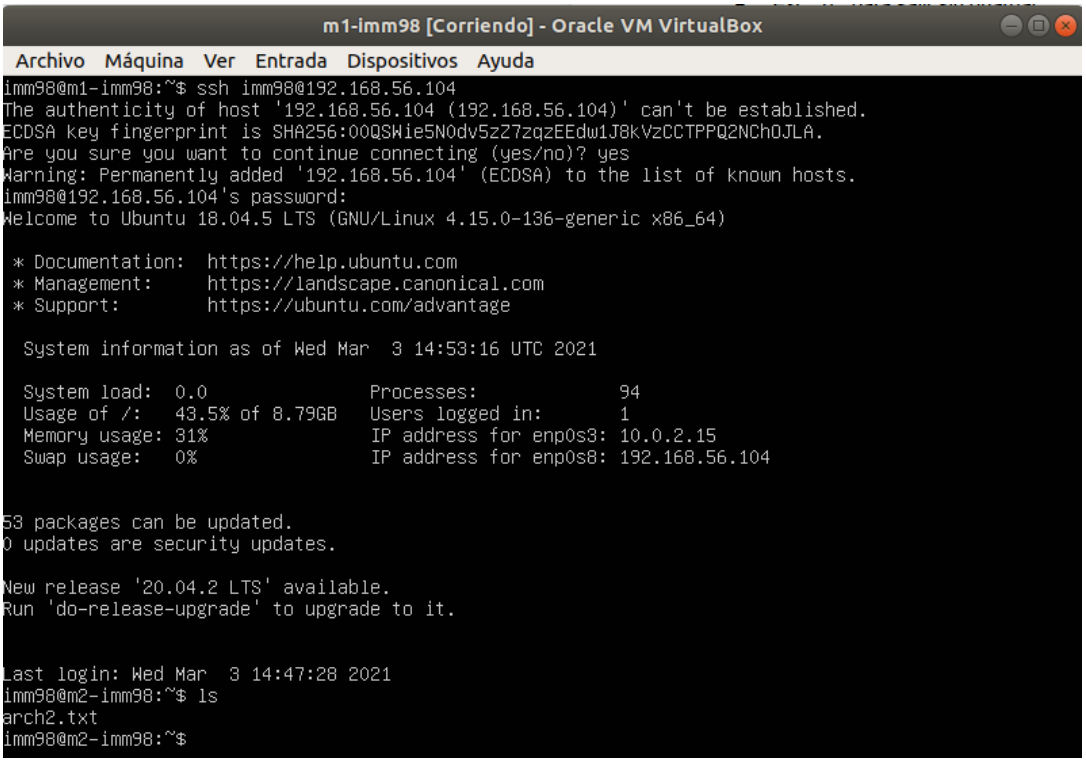
```
Alias /web /web

<Directory /web>
AllowOverride None
Require all granted
</Directory>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet


"000-default.conf" 40L, 1421C written
imm98@m1-imm98:/etc/apache2/sites-enabled$ sudo systemctl restart apache2
```

Voy a probar si efectivamente puedo acceder al archivo index.html que he creado en este directorio:

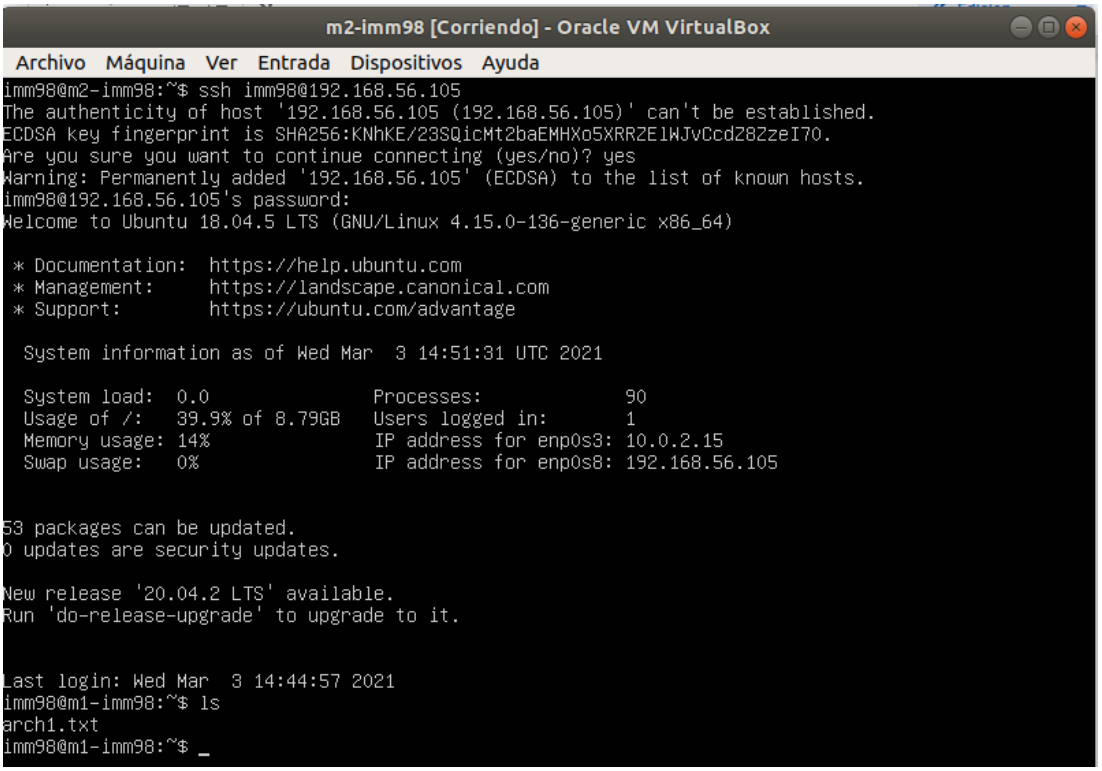


hola web

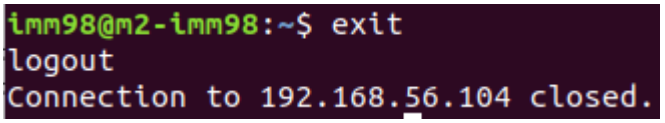
- ssh de m1 a m2:



- ssh de m2 a m1:



y veo que claramente estoy conectado a la otra máquina por lo que cerramos la conexión para terminar lo que queda de la práctica



Ahora vamos a cambiar el puerto de ssh. Para ello tendremos que modificar el archivo de configuración **/etc/ssh/sshd_config** y vamos a poner por ejemplo el puerto de ssh al 2222:

```
imm98@m1-imm98:/etc/ssh$ sudo vi sshd_config
```

```
Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```

y probamos a conectarnos desde la otra máquina indicando con la opción **-p** el nuevo puerto (después de reiniciar el servicio sshd):

```
imm98@m2-imm98:~$ ssh -p 2222 imm98@192.168.56.105
imm98@192.168.56.105's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-136-generic x86_64)

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

  System information as of Tue Mar  9 17:56:02 UTC 2021

  System load:  0.0               Processes:             94
  Usage of /:   43.8% of 8.79GB   Users logged in:       1
  Memory usage: 32%               IP address for enp0s3: 10.0.2.15
  Swap usage:   0%                IP address for enp0s8: 192.168.56.105

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.

     https://microk8s.io/high-availability

54 packages can be updated.
0 updates are security updates.

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Tue Mar  9 17:46:13 2021 from 192.168.56.104
```

Para conectarnos mediante clave pública tenemos que generar una clave pública por lo que ejecutaremos el comando ssh-keygen:

```
imm98@m2-imm98:~$ ssh-key
ssh-keygen   ssh-keyscan
imm98@m2-imm98:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/imm98/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/imm98/.ssh/id_rsa.
Your public key has been saved in /home/imm98/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:NqEThxu8V4Dbl/TYRQoJWwhLpIYXfhl4mQ+cGz7SerU imm98@m2-imm98
The key's randomart image is:
+---[RSA 2048]----+
|     .+=*ooo. ..  |
|   o.=X=.+o. ..   |
|  . =+OB+..=..    |
|   o..*B+o+ o     |
|     o=oSo        |
|    . .+E.        |
|     .            |
|                  |
|                  |
+----[SHA256]-----+
```

que nos crea una clave pública y una privada y nos las mete en los archivos **~/.ssh/id_rsa.pub** y **~/.ssh/id_rsa** respectivamente. Ahora tendremos que pasarle al servidor donde deseamos conectarnos dicha clave pública y esto se hace con el comando ssh-copy-id que nos guarda dicha clave pública en un archivo llamado **~/.ssh/authorized_keys**

```
imm98@m1-imm98:~/.ssh$ ssh-copy-id imm98@192.168.56.104
```

Una vez que ya tenga guardada esa clave en authorized_keys ya me podré conectar sin contraseña de m1 a m2:

```
id_rsa  id_rsa.pub  known_hosts
imm98@m1-imm98:~/.ssh$ ssh imm98@192.168.56.104
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-136-generic x86_64)

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

  System information as of Wed Mar 10 10:19:46 UTC 2021

  System load:  0.0               Processes:             95
  Usage of /:   43.8% of 8.79GB   Users logged in:       1
  Memory usage: 32%               IP address for enp0s3: 10.0.2.15
  Swap usage:   0%                IP address for enp0s8: 192.168.56.104

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.
```


```
imm98@m2-imm98:/etc/ssh$ cd ~/.ssh/
imm98@m2-imm98:~/.ssh$ ls
authorized_keys  id_rsa  id_rsa.pub  known_hosts
imm98@m2-imm98:~/.ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDGQfZpJdhcSvvsajvvUlzWo6x55cZ3QH2MJ2VjQUXWPFH/XL4y2BY3hp2YoDEH
saSIy19gx3MrCGdxjlHkSWGXTSB5PsR6ZBLkcV6Rt2V3lYbnlHgHjlSd4qIezCrMVVOozaXRR6m4Xnh82OFc/oAiQEHmtmGt107F
FEPDB6QbnJyMlWn8cOOOCME5mnCqQ88vshSOYweCnYOUuPiFCvOLxp+accbMbz28dt6KIb1xdObNnKF7nYhjIhw9mU141h2TKJ9a
RU99bod3CZT/uWs8tmVNpY2L1BWL3161TdGgz5XChKnxyDUBGO4/Gu818YQCXWFY1QyX5ShaMYjJFzmj imm98@m1-imm98
```

Ahora vamos a comprobar que apache está funcionando para ello creamos dos archivos HTML en el directorio */var/www/html/* en cuyo contenido viene en que máquina (m1 o m2) están localizados y vamos a acceder a ellos desde la otra máquina usando cURL. Para ello utilizo el comando :
curl *direcciónIP/ejemplo.html*

- m1 a m2

```
imm98@m1-imm98:/etc/ssh$ curl 192.168.56.104/ejemplo.html
<HTML>
        <BODY>
                Web de ejemplo de imm98 para SWAP en m2
                Email: imm98@correo.ugr.es
        </BODY>
</HTML>
```

- m2 a m1

```
imm98@m2-imm98:~/.ssh$ curl 192.168.56.105:8080/ejemplo.html
<HTML>
        <BODY>
                <h1>Web de ejemplo de imm98 para SWAP desde m1</h1>
                <p>Email: imm98@correo.ugr.es</p>
        </BODY>
</HTML>
```

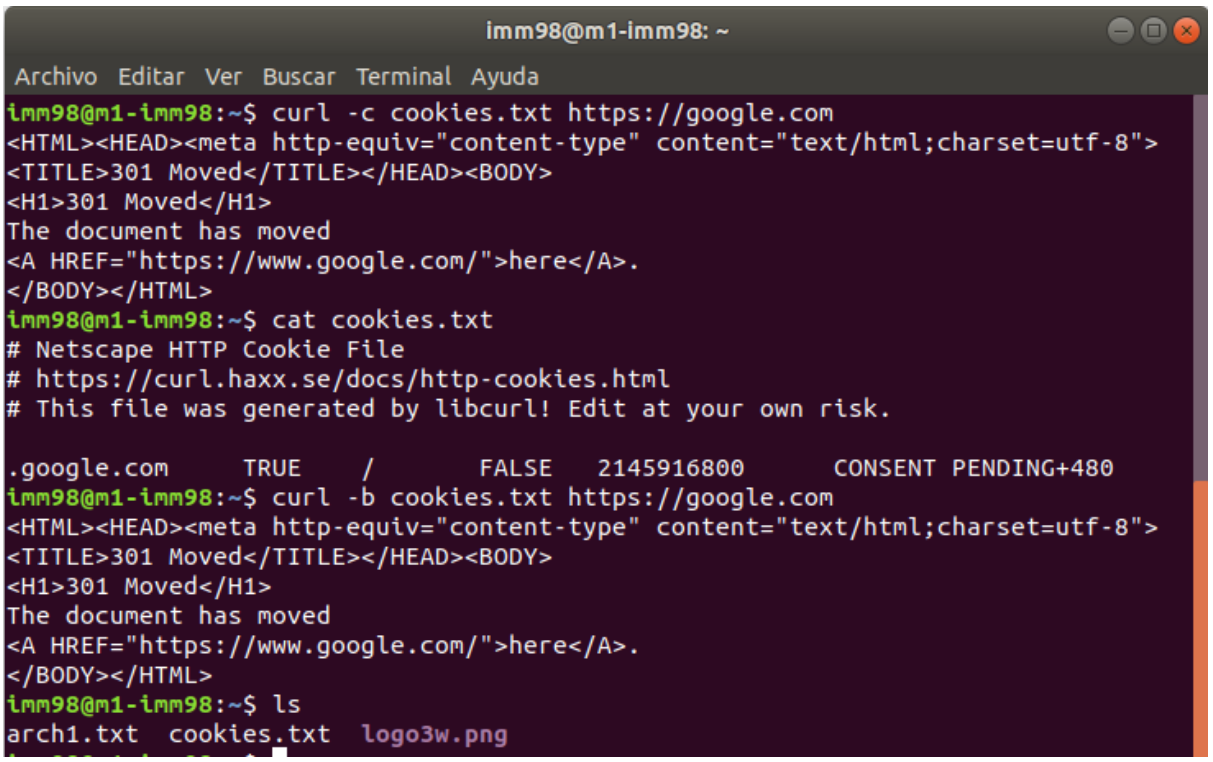
Por último vamos a proceder a descargar un archivo, en lugar de que lo muestre por la salida estándar. Para ello utilizo la orden:
curl -o *imagen.png* *https://www.google.es/images/srpr/logo3w.png*

```
imm98@m1-imm98:~$ sudo curl -o imagen.png https://www.google.es/images/srpr/logo3w.png
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  6748  100  6748    0     0  76681      0 --:--:-- --:--:-- --:--:-- 76681
imm98@m1-imm98:~$ ls
imagen.png
```

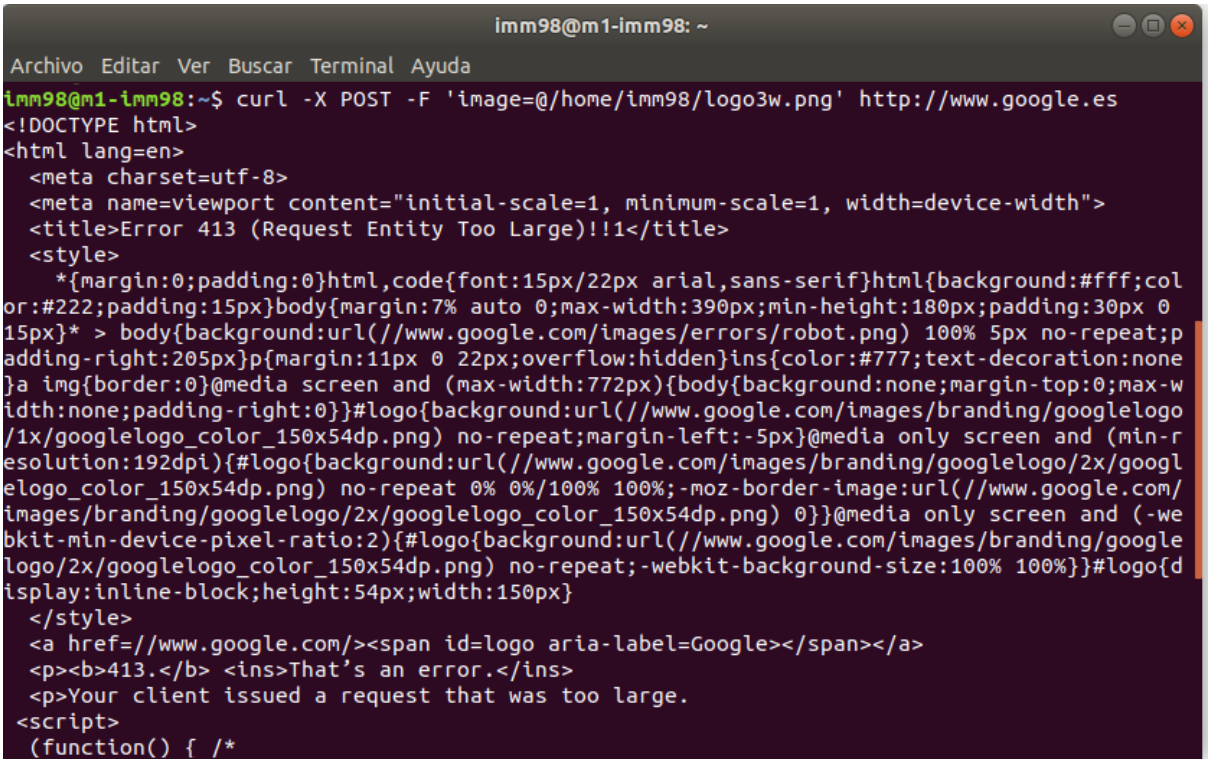
La opción curl -O nos descarga un archivo y le pone el nombre del archivo que estamos descargando:
curl -O *https://www.google.es/images/srpr/logo3w.png*

```
imm98@m1-imm98: ~
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
imm98@m1-imm98:~$ curl -O https://www.google.es/images/srpr/logo3w.png
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  6748  100  6748    0     0  65514      0 --:--:-- --:--:-- --:--:-- 65514
imm98@m1-imm98:~$ ls
arch1.txt  logo3w.png
```

Para el tema de las cookies tenemos dos opciones principales curl -b y curl -c:
curl -c se utiliza para guardar las cookies de respuesta en un determinado fichero. En nuestro ejemplo vamos a guardarlas en el fichero cookies.txt. La opción **curl -b** set a cookie with the values written in a text file in a GET request.



Para realizar una petición POST se hace con la opción **curl -X POST -F** junto con el archivo que desees enviarle al servidor:



Para realizar una petición GET simplemente hay que hacer **curl Dirección IP**