

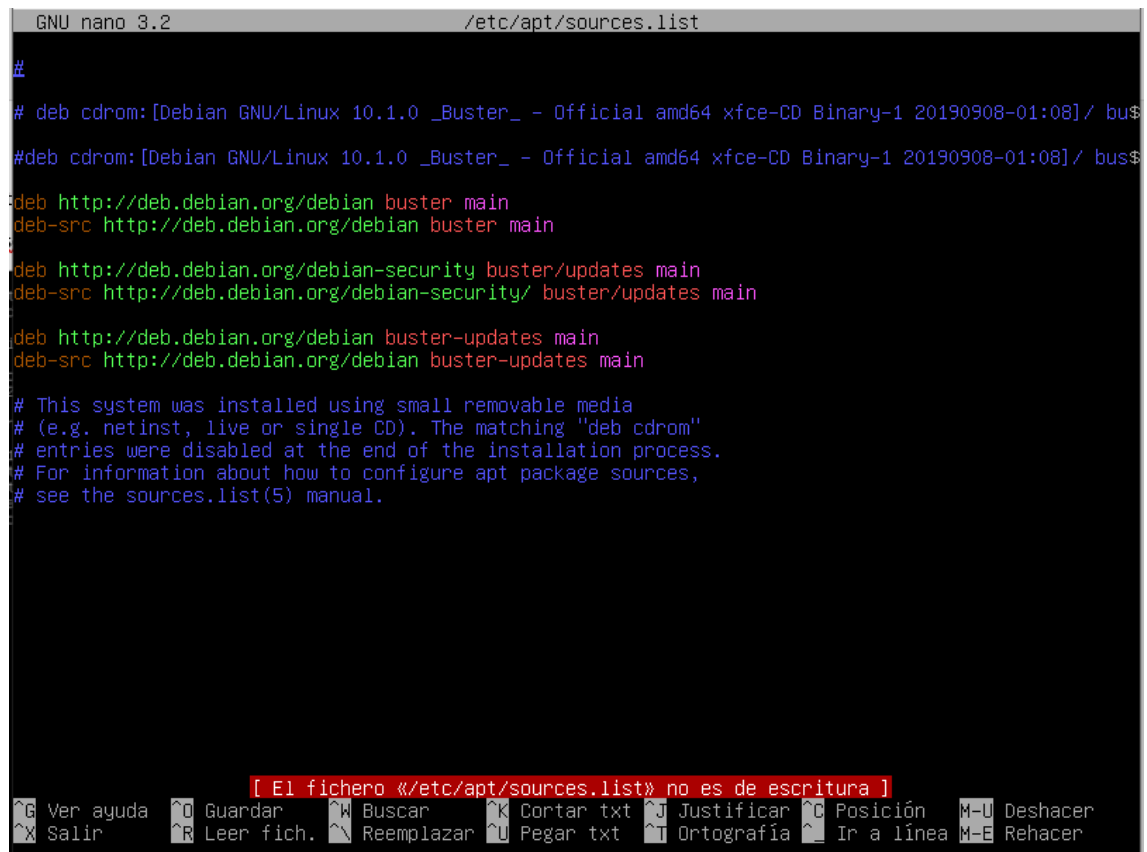
Sistemas Distribuidos

Trabajo Práctico n°1

En el presente trabajo se construyó un cluster compuesto por un nodo maestro y dos esclavos. El nodo maestro compartirá un recurso con los esclavos, en él se encontrará almacenado un código que permite el calculo de numero PI. Este recurso se encontrará montado en cada máquina y mediante MPI, el algoritmo se ejecutará de forma paralela.

A continuación, se detallará del procedimiento necesario, desde la configuración de cada nodo hasta la ejecución del algoritmo.

Primero, se debe verificar que el repositorio este configurado correctamente, para ello se empleó el comando `nano /etc/apt/sources.list` y se validó que se emplee el repositorio original de Linux.



```
GNU nano 3.2 /etc/apt/sources.list

#
# deb cdrom:[Debian GNU/Linux 10.1.0 _Buster_ - Official amd64 xfce-CD Binary-1 20190908-01:08]/ bus$
#deb cdrom:[Debian GNU/Linux 10.1.0 _Buster_ - Official amd64 xfce-CD Binary-1 20190908-01:08]/ bus$
deb http://deb.debian.org/debian buster main
deb-src http://deb.debian.org/debian buster main
deb http://deb.debian.org/debian-security buster/updates main
deb-src http://deb.debian.org/debian-security/ buster/updates main
deb http://deb.debian.org/debian buster-updates main
deb-src http://deb.debian.org/debian buster-updates main

# This system was installed using small removable media
# (e.g. netinst, live or single CD). The matching "deb cdrom"
# entries were disabled at the end of the installation process.
# For information about how to configure apt package sources,
# see the sources.list(5) manual.

[ El fichero «/etc/apt/sources.list» no es de escritura ]
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición M-U Deshacer
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea M-E Rehacer
```

Instalación de paquetes

Para la instalación es necesario loguearse como root con el comando `su -` y posteriormente se debe introducir la contraseña. Luego, con el comando `sudo apt-get install` se realizó la instalación de múltiples paquetes, que permiten lograr el propósito del trabajo, los cuales son:

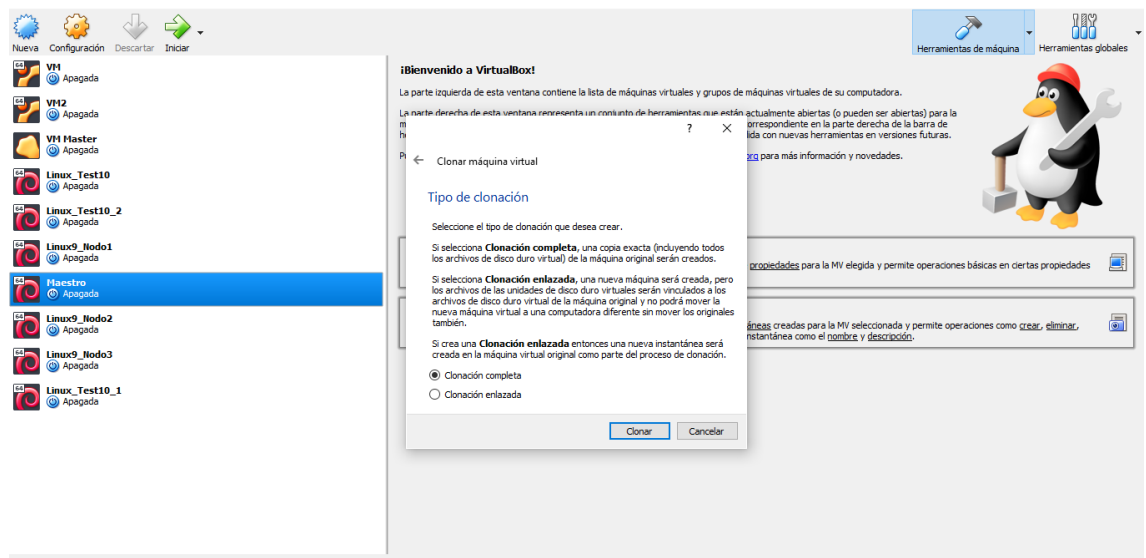
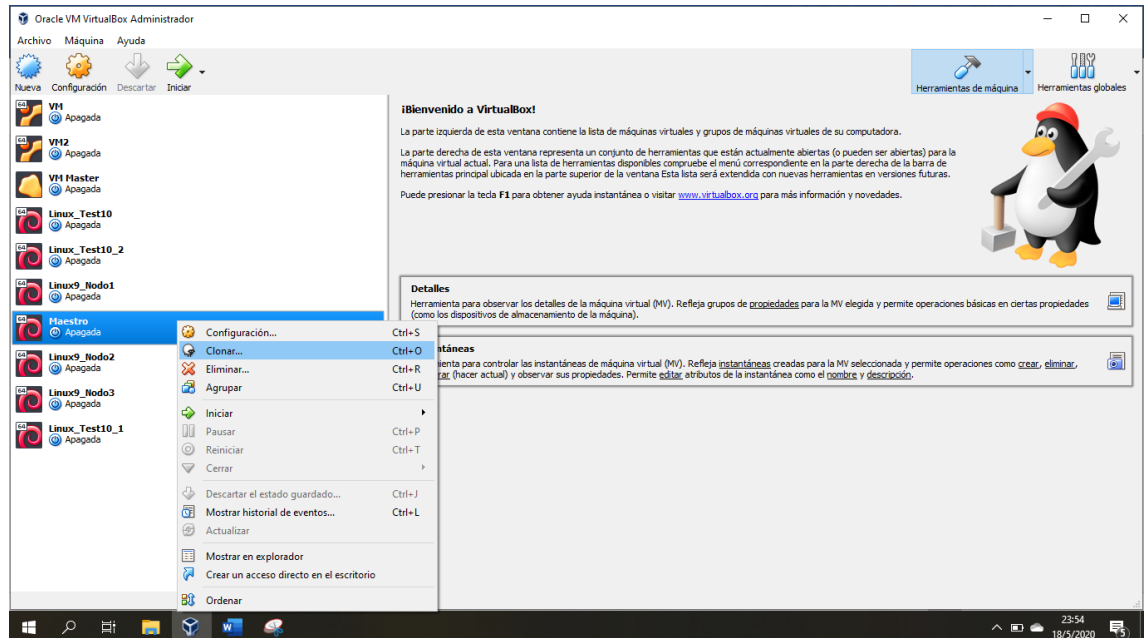
Sudo `apt-get install mpich`

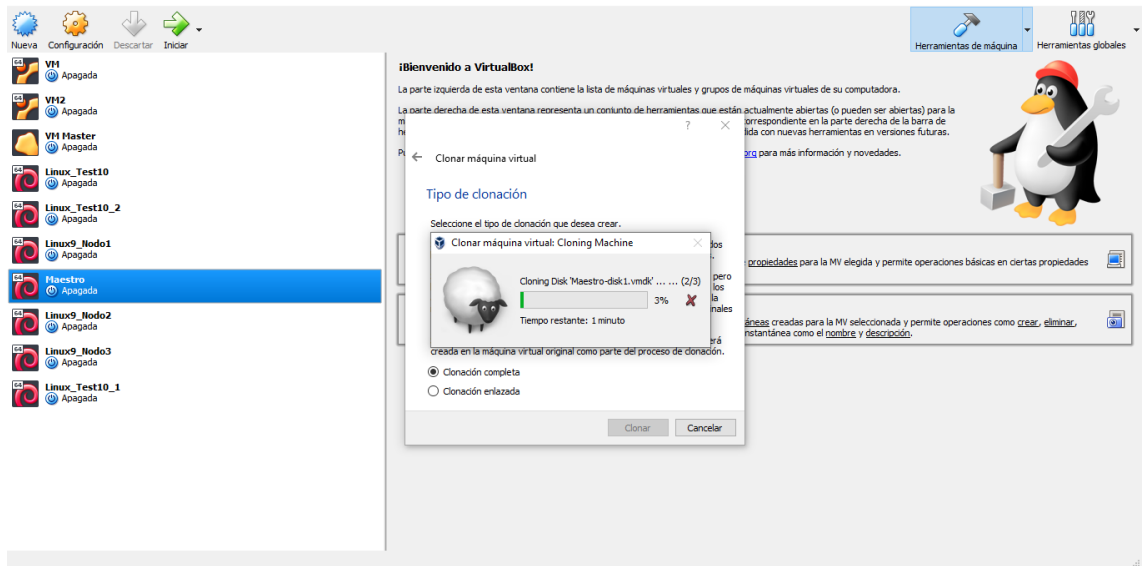
Sudo `ap—get install uml-utilities bridge-utils`

Sudo apt-get install open-ssh

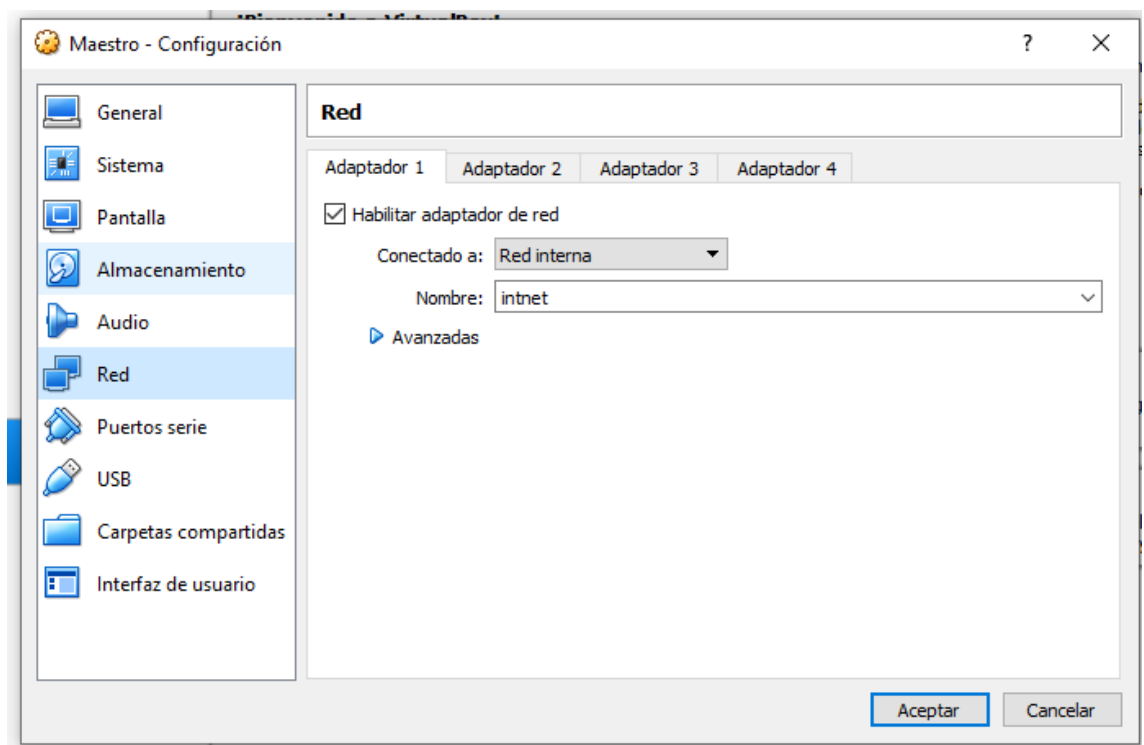
Sudo apt-get install nfs-kernel-server nfs-common

Una vez, que haya concluido satisfactoriamente la instalación se debe clonar la máquina, que serán los nodos esclavos.





Finalmente, es necesario acceder al apartado configuraciones, red y modificar cada nodo a red interna.



Luego se debe configurar en cada nodo (maestro y esclavos), la dirección Ip para que sea fija.

Para ello es necesario ejecutar el siguiente comando `sudo nano /etc/network/interface`.

Configuración del maestro

```
GNU nano 3.2 /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#allow-hotplug enp0s3
#iface enp0s3 inet dhcp
auto enp0s3
iface enp0s3 inet static
address 192.168.0.10
netmask 255.255.255.0
```

[19 líneas leídas]

Ver ayuda	Guardar	Buscar	Cortar txt	Justificar	Posición	Deshacer
Salir	Leer fich.	Reemplazar	Pegar txt	Ortografía	Ir a línea	Rehacer

Configuración de los esclavos

```
GNU nano 3.2 /etc/network/interfaces Modificado

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#allow-hotplug enp0s3
#iface enp0s3 inet dhcp
auto enp0s3
iface enp0s3 inet static
address 192.168.0.11
netmask 255.255.255.0
```

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición M-U Deshacer
 ^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea M-E Rehacer

```
GNU nano 3.2 /etc/network/interfaces Modificado

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#allow-hotplug enp0s3
#iface enp0s3 inet dhcp
auto enp0s3
iface enp0s3 inet static
address 192.168.0.12
netmask 255.255.255.0
```

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición M-U Deshacer
 ^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea M-E Rehacer

Luego, se debe bajar el puerto y volverlo a iniciar para que la configuración sea efectiva, para ello se debe emplear los comandos `ifdown puerto`, para apagarlo, e `ifup puerto` para iniciarlo, en este caso el puerto fue `enp0s3`

```
linuxtest@Linux10:~$ sudo ifdown enp0s3
linuxtest@Linux10:~$ sudo ifup enp0s3
linuxtest@Linux10:~$ _
```

Para comprobar que las configuraciones estén correctas, es decir las IPs sean estáticas, se debe ejecutar el comando `ifconfig`, ya que nos permite conocer el puerto y la dirección IP, entre otros datos de red.

Maestro

```
linuxtest@Linux10:~$ sudo ifdown enp0s3
linuxtest@Linux10:~$ sudo ifup enp0s3
linuxtest@Linux10:~$ sudo ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.10 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::a00:27ff:fe0c:a0b3 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:0c:a0:b3 txqueuelen 1000 (Ethernet)
    RX packets 171 bytes 18446 (18.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 169 bytes 17788 (17.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

linuxtest@Linux10:~$ _
```

Esclavos

```
linuxtest@Linux10Nodo0:~$ sudo ifconfig
sudo: unable to resolve host Linux10Nodo0: Fallo temporal en la resolución del nombre
[sudo] password for linuxtest:
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.11 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::a00:27ff:fe0c:a0b3 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:0c:a0:b3 txqueuelen 1000 (Ethernet)
    RX packets 154 bytes 16670 (16.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 195 bytes 21152 (20.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

linuxtest@Linux10Nodo0:~$
```



```
linuxtest@Linux10Nodo1:~$ sudo ifconfig
sudo: unable to resolve host Linux10Nodo1: Fallo temporal en la resolución del nombre
[sudo] password for linuxtest:
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.12 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::a00:27ff:fe0c:a0b3 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:0c:a0:b3 txqueuelen 1000 (Ethernet)
    RX packets 36 bytes 7132 (6.9 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 43 bytes 7446 (7.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

linuxtest@Linux10Nodo1:~$ _
```

Para la configuración de ssh se debe realizar los siguientes pasos:

Primero, se debe navegar hasta la carpeta /home/linuxtest, luego con el comando mkdir se debe crear una carpeta de nombre .ssh. Posteriormente, con el comando cd se ingresa a la carpeta.

```
linuxtest@Linux10:~$ cd /home/linuxtest/
linuxtest@Linux10:~$ mkdir .ssh_
```

```
linuxtest@Linux10:~/ssh$
```

Luego, se debe ejecutar el siguiente comando, que permite la creación de claves tanto pública como privada del tipo rsa.

```
linuxtest@Linux10:~/ssh$ ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (/home/linuxtest/.ssh/id_rsa):
/home/linuxtest/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/linuxtest/.ssh/id_rsa.
Your public key has been saved in /home/linuxtest/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:igZ0yHarq7QSnEFgyNAGxY6vVQ+Z2e2Dgrj49x2RH0s linuxtest@Linux10
The key's randomart image is:
+----[RSA 2048]-----+
|0*.   ...          |
|00+    0.          |
|. +    = 0.         |
|00. * 0..          |
|0==+ 0 +E          |
|+=.0.+0            |
|*.0... .           |
|* 0.. .            |
|+=0. 0.            |
+----[SHA256]-----+
linuxtest@Linux10:~/ssh$ _
```

Empleando ls, se puede comprobar que ambas claves se generaron.

Para poder ingresar en las maquina esclavas, sin que solicite contraseña en el inicio de sesion, se debe copiar la clave publica, que se encuentra en el nodo maestro, en cada maquina esclava. Es necesario crear una carpeta .ssh en cada nodo y debe estar ubicada en /home/linuxtest

El siguiente comando permite copiar la clave publica desde el maestro hacia el esclavo, es necesario indicar la carpeta donde debe ser copiado el archivo, en este caso es /home/linuxtest/.ssh

```
linuxtest@Linux10:~/ssh$ scp id_rsa.pub linuxtest@192.168.0.11:/home/linuxtest/.ssh/
The authenticity of host '192.168.0.11 (192.168.0.11)' can't be established.
ECDSA key fingerprint is SHA256:2QDqELDip2xMmqyDcz25pioVdfglbIeoHeDFLwbeS5c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.11' (ECDSA) to the list of known hosts.
linuxtest@192.168.0.11's password:
id_rsa.pub                                100% 399      9.8KB/s   00:00
linuxtest@Linux10:~/ssh$
```

Luego, en el nodo esclavo se necesario comprobar si s se copió el archivo para ello se empleó el comando ls, ya que nos listara todos los documentos contenidos en la carpeta .ssh. Finalmente, se debe copiar el contenido de la clave publica en un archivo con nombre authorized_keys

```
linuxtest@Linux10Nodo0:~/ssh$ ls
id_rsa  id_rsa.pub
linuxtest@Linux10Nodo0:~/ssh$ cat id_rsa.pub >> authorized_keys
linuxtest@Linux10Nodo0:~/ssh$ _
```

Para comprobar que la configuración haya sido exitosa, desde el nodo maestro se debe intentar acceder al nodo esclavo, mediante ssh pero esta vez no debe solicitar contraseña de acceso.

```
linuxtest@Linux10:~/ssh$ ssh 192.168.0.11
Linux Linux10Nodo0 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue May 19 01:02:35 2020 from 192.168.0.10
linuxtest@Linux10Nodo0:~$ sudo ifconfig
sudo: unable to resolve host Linux10Nodo0: Fallo temporal en la resolución del nombre
[sudo] password for linuxtest:
linuxtest@Linux10Nodo0:~$ sudo ifconfig
sudo: unable to resolve host Linux10Nodo0: Fallo temporal en la resolución del nombre
[sudo] password for linuxtest:
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.11 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::a00:27ff:fe0c:a0b3 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:0c:a0:b3 txqueuelen 1000 (Ethernet)
    RX packets 367 bytes 42073 (41.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 380 bytes 46155 (45.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

linuxtest@Linux10Nodo0:~$ _
```

En la imagen se puede visualizar, que se empleó ifconfig para confirmar que se ingresó al nodo esclavo.

Configuración de NFS

Mediante el uso del módulo NFS server se disponibilizó la carpeta que contiene el código, para que este, posteriormente sea ejecutado por los nodos. Para lograrlo, primero se debe crear una carpeta en la ubicación /home/linuxtest con el comando mkadir en el nodo maestro.

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
linuxtest@Linux10:~$ cd /home/linuxtest/
linuxtest@Linux10:~$ mkdir share/_
```

Luego se debe editar el archivo exports, que se encuentra en la carpeta /etc, para ello se empleó el comando nano

```
linuxtest@Linux10:~$ nano /etc/exports _
```

```
GNU nano 3.2 /etc/exports Modificado
# /etc/exports: the access control list for filesystems which may be exported
# to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4 gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
#
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición M-U Deshacer
^X Salir ^R Leer fich. ^E Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea M-E Rehacer
```

En este archivo, se debe indicar el recurso será accesible por los nodos, quienes podrán acceder, el nivel de permisos que tendrán los nodos (leer/escribir), si el servidor responderá a peticiones cuando se complete la operación del disco (este caso se debe especificar sync) caso contrario se lo debe sustituir por async, y, además, se debe indicar el nivel de acceso de root de los clientes al sistema de archivos.

```
GNU nano 3.2 /etc/exports
# /etc/exports: the access control list for filesystems which may be exported
# to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4 gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
#
/home/linuxtest/share *(rw,sync,no_subtree_check,no_root_squash)
```

En este caso, se indicó que el recurso que se compartió es share que se encuentra en /home/linuxtest y que todos los nodos pueden acceder, que se puede tanto leer como escribir, que las peticiones para la operación en el disco serán sincrónicas y que el usuario de root de los nodos esclavos, tendrá el mismo nivel de permisos en esa carpeta que el root del maestro

Luego, se debe reiniciar el servicio de nfs, con el comando `nfs-kernel-server restart`. Para comprobar que la carpeta esté disponible se empleó el siguiente comando `sudo mount -e`, que indica que recursos está compartiendo el servidor.

```
linuxtest@Linux10:~$ sudo showmount -e
[sudo] password for linuxtest:
Export list for Linux10:
/home/linuxtest/share *
linuxtest@Linux10:~$ _
```

Luego se debe ir a cada nodo y crear una carpeta (es donde se mapeará el sistema de archivos). Luego, cuando se esté posicionado sobre la carpeta (en el esclavo) se debe ejecutar el siguiente comando.

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
linuxtest@Linux10Nodo0:~$ mount -y nfs 192.168.0.10:/home/linuxtest/share/_
```

Para comprobar que la carpeta se haya montado correctamente se debe ejecutar el siguiente comando y se debe ver la carpeta del nodo maestro

```
linuxtest@Linux10Nodo0:~$ df -h
S.ficheros          Tamaño Usados  Disp Uso% Montado en
udev                480M      0   480M   0% /dev
tmpfs               99M      3,0M   96M   3% /run
/dev/sda1           6,9G    1,7G   4,9G  26% /
tmpfs               494M      0   494M   0% /dev/shm
tmpfs               5,0M      0   5,0M   0% /run/lock
tmpfs               494M      0   494M   0% /sys/fs/cgroup
192.168.0.10:/home/linuxtest/share 6,9G    1,7G   4,9G  26% /home/linuxtest/share
tmpfs               99M      0    99M   0% /run/user/1000
linuxtest@Linux10Nodo0:~$ _
```

Luego accediendo a dicha carpeta, y listando los archivos se podrá ver todos los archivos creados.

Nodo Esclavo

```
linuxtest@Linux10Nodo0:~/share$ ls
4      archivo2 calculo calculPi hello host pruebamensaje.c
archivo archivo3 calculo.c calculPi.c hello.c pruebamensaje
linuxtest@Linux10Nodo0:~/share$ _
```

Nodo Maestro

```
linuxtest@Linux10:~/share$ ls
4      archivo2 calculo calculPi hello host pruebamensaje.c
archivo archivo3 calculo.c calculPi.c hello.c pruebamensaje
linuxtest@Linux10:~/share$
```

Para que el recurso compartido, sea montado de forma automática (cada vez que se inicia el nodo) se debe editar el archivo `fstab` que se encuentra en la carpeta `/etc` y se debe especificar cuál es carpeta del nodo maestro que se desea montar y hacia que carpeta del nodo esclavo y mediante que protocolo.

```
GNU nano 3.2 /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda1 during installation
UUID=c39d52af-966e-4c0c-99c6-55443763ff5c / ext4 errors=remount-ro 0 1
# swap was on /dev/sda5 during installation
UUID=569c66d8-3de1-42d8-9412-6b9b0f4a36a4 none swap sw 0 0
/dev/sr0 /media/cdrom0 udf,iso9660 user,noauto 0 0
192.168.0.10:/home/linuxtest/share /home/linuxtest/share nfs defaults 0 0
```

Luego para comprobar que la configuración este correcta se debe reiniciar el equipo.

Posteriormente, se debe cambiar el nombre del equipo, para ello se debe editar el archivo hostname, en cada maquina y asignarle un nombre. En este Caso, el maestro se llama Linux10 y los esclavos Linux10_Nodo_0 Linux10_Nodo1.

Nodo Maestro

```
GNU nano 3.2 /etc/hostname
#debian
Linux10
```

Nodo esclavo

```
GNU nano 3.2 /etc/hostname
#debian
Linux10_Nodo_0
```

```
GNU nano 3.2 /etc/hostname Modificado
#debian
Linux10_Nodo_1
```

Luego se debe editar el archivo de host, es en este archivo se debe ingresar el nombre que identificara cada dirección IP. Luego se debe reiniciar cada máquina,

```
#127.0.0.1    localhost
#127.0.1.1    debian

# The following lines are desirable for IPv6 capable hosts
#::1          localhost ip6-localhost ip6-loopback
#ff02::1      ip6-allnodes
#ff02::2      ip6-allrouters

192.168.0.10  Linux10
192.168.0.11  Linux10_Nodo_0
192.168.0.12  Linux10_Nodo_1
```

Posteriormente, para comprobar que las configuraciones son correctas se debe acceder por ssh pero esta vez empleando el nombre anteriormente configurado

```
linuxtest@Linux10:~/share$ ssh Linux10_Nodo_0
Linux Linux10Nodo0 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed May 20 00:24:11 2020 from 192.168.0.10
linuxtest@Linux10Nodo0:~$ sudo ifconfig
sudo: unable to resolve host Linux10Nodo0: Fallo temporal en la resolución del nombre
[sudo] password for linuxtest:
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.0.11  netmask 255.255.255.0  broadcast 192.168.0.255
    inet6 fe80::a00:27ff:fe0c:a0b3  prefixlen 64  scopeid 0x20<link>
    ether 08:00:27:0c:a0:b3  txqueuelen 1000  (Ethernet)
    RX packets 839  bytes 85344 (83.3 KiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 915  bytes 92636 (90.4 KiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 0  bytes 0 (0.0 B)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 0  bytes 0 (0.0 B)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

linuxtest@Linux10Nodo0:~$
```

Para la paralización se empleó, el caso de estudio del cálculo de número pi. Tanto el código como su compilado, deben estar guardado en el recurso compartido a la red, para que todos los nodos puedan acceder.

Se debe crear un archivo en la carpeta disponible a la red donde se indique la dirección ip o el nombre de las máquinas que van a ejecutar el código, la configuración debe ser la siguiente

```
GNU nano 3.2      host      Modificado
Linux10
Linux10_Nodo_0
192.168.0.12
```

Luego con el comando nano o touch se puede crear el el programa que debe tener extensión .c, ya que es un programa codificado en c.

```
GNU nano 3.2      calculo.c

#include<stdio.h>
#include"mpi.h"
#include<math.h>
double valorreal=3.141592653589793238462643;
double f(double);
double f(double a){
    return 4/(1.0+a*a);
}

int main(int argc, char *argv[])
{
    int n, myid, numprocs, i;
    double mypi, pi, h, sum=0.0, x;
    double end, begin=0.0;
    int namelen;
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &myid);
    MPI_Get_processor_name(processor_name, &namelen);
    //fprintf(stdout,"proceso %d de %d en %s\n", &myid,numprocs, processor_name);
    printf("el proeso %d de %d en %s\n", &myid, numprocs, processor_name);
    //fflush(stdout);
    n=1000000;
    if(myid==0){
        begin=MPI_Wtime();
    }
    //MPI_Bcast(&n, 1,MPI_INT, 0, MPI_COMM_WORLD);
    h=1.0 /((double)n);
    //printf("el valor de h es %lg\n ", &h);
    //sum=0.0;

    [ 48 líneas leídas ]
^G Ver ayuda  ^O Guardar  ^W Buscar  ^K Cortar txt  ^J Justificar  ^C Posición  M-U Deshacer
^X Salir      ^R Leer fich. ^_ Reemplazar ^U Pegar txt  ^T Ortografía ^_ Ir a línea M-E Rehacer
```

```
for(i=myid+1; i<=n;i+=numprocs){
    x=h*((double)i-0.5);
    sum+= f(x);
}
mypi=h*sum;
//printf("el valor de mypi %lg", &mypi);
MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
if(myid==0){
    end=MPI_Wtime();
    printf("el valor de pi es %.16f con un tiempo de %f\n ",pi, end-begin);
    printf("el margen de error es %.16f\n ",pi-valorreal );
    //fflush(stdout);
}
MPI_Finalize();
return 0;
```

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición M-U Deshacer
^X Salir ^R Leer fich. ^_ Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea M-E Rehacer

Para compila el programa, y verificar que el mismo no posee errores se debe utilizar el siguiente comando mpicc calculo.c -o calculo, luego para poder ejecutarlo se emplea `mpiexec -f host -n 4 ./calculo`, en este comando se indica

la cantidad de procesos que se ejecutan, en caso que se quiere hacer un cálculo más exacto se debe aumentar el número de procesos a ejecutar.

Con 4 procesos de ejecutándose por nodo el margen de error es 0.00000000000001101

```
linuxtest@Linux10:~/share$ mpiexec -f host -n 4 ./calcula
el proceso 67513140 de 4 en Linux10Nodo1
el proceso 833449812 de 4 en Linux10Nodo0
el proceso -1287188892 de 4 en Linux10
el proceso 1777154404 de 4 en Linux10
el valor de pi es 3.1415926535899033 con un tiempo de 0.028668
el margen de error es 0.00000000000001101
linuxtest@Linux10:~/share$
```

Si se aumenta a 8 el margen de error disminuye

```
linuxtest@Linux10:~/share$ mpiexec -f host -n 8 ./calcula
el proceso 105013636 de 8 en Linux10Nodo1
el proceso -2129253116 de 8 en Linux10Nodo1
el proceso -345691164 de 8 en Linux10Nodo0
el proceso 602142244 de 8 en Linux10
el proceso 89817140 de 8 en Linux10Nodo0
el proceso -60181804 de 8 en Linux10Nodo0
el proceso 2099787524 de 8 en Linux10
el proceso 1920486580 de 8 en Linux10
el valor de pi es 3.1415926535898899 con un tiempo de 0.011873
el margen de error es 0.0000000000000968
linuxtest@Linux10:~/share$
```

Anexo Código fuente

```
#include<stdio.h>
#include"mpi.h"
#include<math.h>
double valorreal=3.141592653589793238462643;
double f(double);
double f(double a){
    return 4/(1.0+a*a);
}

int main(int argc, char *argv[])
{
    int n, myid, numprocs, i; // n numero de interacion, myid es el identificador de
    proceso, numprocs es el numero del proceso
    double mypi, pi, h, sum=0.0, x; // myp se la usara como variable auxiliar para el
    calculo de pi, pi contendra el valor real de pi, h aproximacion del area para el calculo de
    pi, sum acumulador para la suma.
    double end, begin=0.0; // indica el tiempo de inicio y el fin
    int namelen; // guardara el nombre del proceso que ejecuta
    char processor_name[MPI_MAX_PROCESSOR_NAME]; // se empleara para
    obtener el nombre del proceso
    MPI_Init(&argc, &argv); // inicializa el proceso
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs); // se obtiene el numero
    total de procesos
    MPI_Comm_rank(MPI_COMM_WORLD, &myid); // se obtiene el identificador del
    proceso-
    MPI_Get_processor_name(processor_name, &namelen); // se obtiene el nombre
    del proceso que ejecuta
    //fprintf(stdout,"proceso %d de %d en %s\n", &myid,numprocs,
    processor_name);
    printf("el proceso %d de %d en %s\n", &myid, numprocs, processor_name); // se
    imprime el identificador de proceso, el numero y el nombre de quien ejecuta el proceso.
    //fflush(stdout);
    n=1000000; // se establece el numero de iteraciones.
    if(myid==0){ //si ejecuta el proceso padre ingresa al if
        begin=MPI_Wtime(); // almacena el tiempo de inicio
    }
    //MPI_Bcast(&n, 1,MPI_INT, 0, MPI_COMM_WORLD);
    h=1.0/(double)n; // se obtiene la aproximacion del area
    //printf("el valor de h es %lg\n ", &h);
    //sum=0.0;
    for(i=myid+1; i<=n;i+=numprocs){ // el proceso se ejecuta siempre y cuando el
    numero de procesos mas la cantidad de procesos a ejecutar no supere el valor de n
        x=h*((double)i-0.5);
        sum+= f(x);
    }
    mypi=h*sum; // almacena el valor de calculo del acumulador mas el area
    //printf("el valor de mypi %lg", &mypi);
    MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0,
    MPI_COMM_WORLD); // mypi es el valor local del pi, pi es donde se va a guardar, 1
    numero de datos que vamos a guardar por cada ejecucion, MPI_DOUBLE tipo de dato
```

que se va a guardar, MPI_SUM operacion que se va a realizar sobre pi, 0 el proceso que recibe los datos, que es el maestro

```
    if(myid==0){ si es el nodo maestro ingreso
        end=MPI_Wtime(); //gaurdo el tiempo en que finalizo
        printf("el valor de pi es %.16f con un tiempo de %f\n ",pi, end-begin);
//imprimo el valor calculado de pi y el tiempp que se tardo
        printf("el margen de error es %.16f\n ",pi-valorreal );//imprimo el margen
de error, diferencia entre lo calculado y la aproximacion definida de pi.
        fflush(stdout);
    }
    MPI_Finalize();
    return 0;
}
```