

# **Computació Paral·lela amb un clúster de Raspberry Pi. Programació amb Python i MPI.**

**Joan Gerard Camarena Estruch.**

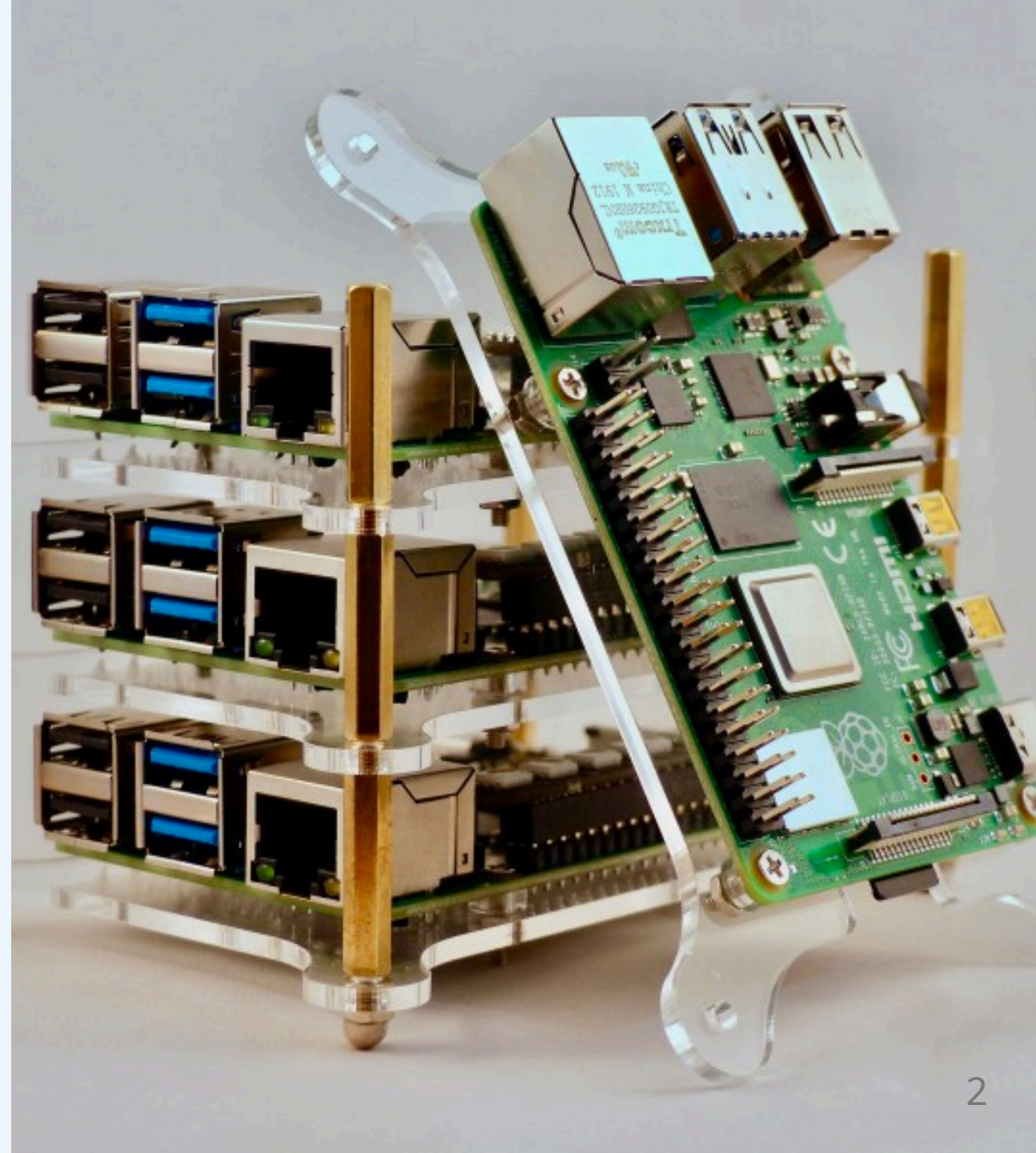
# Elements necessaris

## Hardware

- 4 raspberry Pi amb alimentador
- 4 microSD
- 1 Switch
- 1 rack per a Raspberry

## Software:

- MPI (Message Passing Interface)
- Python
- ssh



# Pla de treball

1. Acoblament del clúster
2. Instal·lar i configurar la imatge bàsica (*master*)
3. Copia i configuració de la resta d'imatges (*nodes*)
4. Configuració del clúster. Carpetes en xarxa. `ssh` i `clusterssh`
5. MPI (Message Passing Interface) Teoria i fonaments
6. Proves de programació distribuïda

# Acoblament del cluster

Teniu com a material:

- 4 raspberrys Pi
- 4 carregadors
- 4 microSD
- 5 cables RJ45
- 1 PC
- 1 rack
- 1 switch

Seguint les instruccions, atornillarem les plaquetes al rack i l'ensamblarem.  
També hauriem de cablejar les raspberrys al switch

# Imatge (1)

Descarregarem la imatge del sistema operatiu de la raspberry.

Donat que ho configurarem tot per `ssh` i la terminal, no necessitarem cap escriptori. També aconseguim un menor consum de recursos.

La versió més actual és:

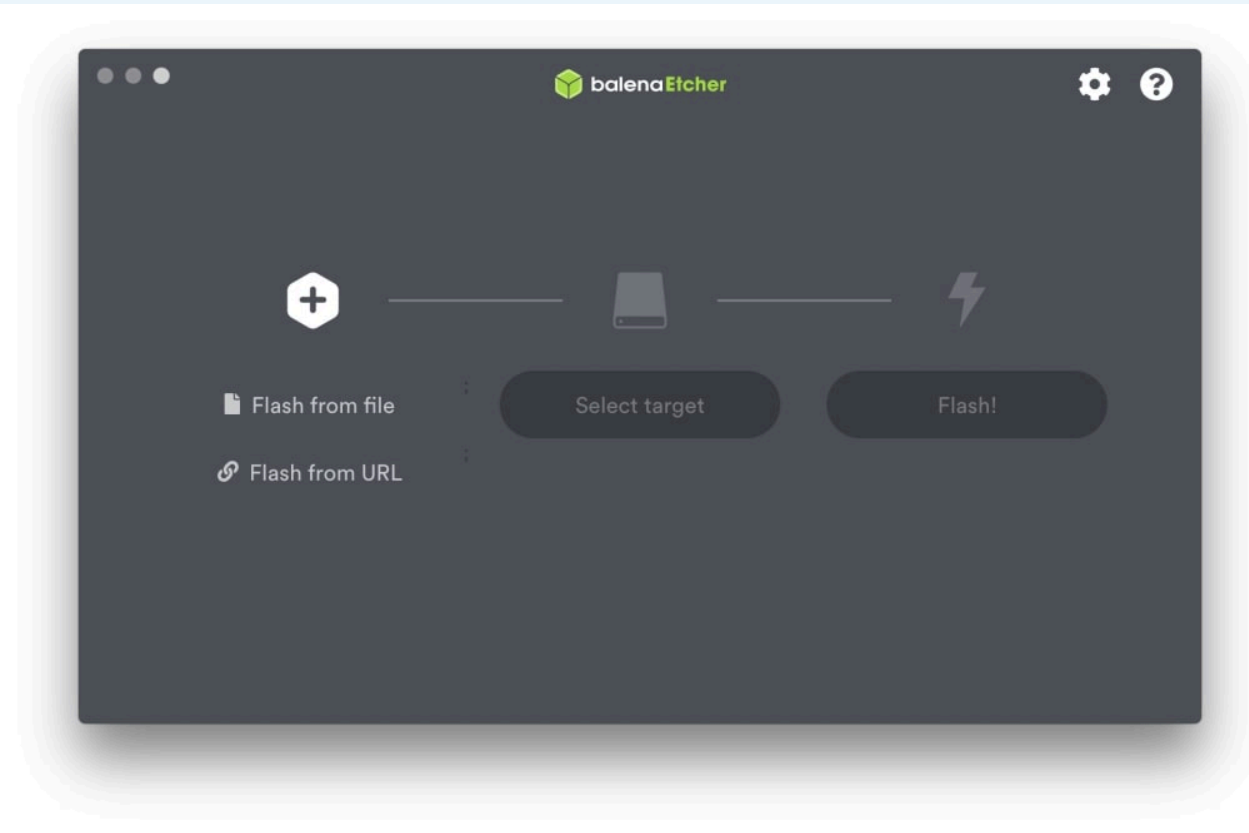
[https://downloads.raspberrypi.org/raspios\\_lite\\_arm64/images/raspios\\_lite\\_arm64-2023-02-22/2023-02-21-raspios-bullseye-arm64-lite.img.xz](https://downloads.raspberrypi.org/raspios_lite_arm64/images/raspios_lite_arm64-2023-02-22/2023-02-21-raspios-bullseye-arm64-lite.img.xz)

Nosaltres farem servir la versió de gener del 21, per millorar la compatibilitat.

# Imatge (2)

El fitxer `img` resultant el tenim que escriure a una targeta microSD. Per a fer-ho podem fer servir distints programes:

- [Windows] → `Win32ImageWriter`, disponible en <https://launchpad.net/win32-image-writer>
- [Ubuntu] → `usb-imagewriter`, dispoible als repositoris d'ubuntu
- Alternativa multiplataforma i recomanada `balenaEtcher` <https://www.balena.io/etcher/> (sols grava a sd)



# Imatge (3)

Des de la consola en sistemes **Linux/Mac**, tant per gravar com per clonar:

1. Executar `sudo fdisk -l` i fixar-se dels discs que tenim al nostre sistema. En mac `diskutil list`
2. Inserir la microSD directament o acoplador. Es recomana un port USB 3.
3. Tornem a executar `sudo fdisk -l`. Fixar-se en el tamnay (8 o 16GB). Serà algun disk que no teniem al pas 1. Suposarem que s'ha muntat en `/dev/sdx`.
4. En cas que linux l'haja muntada, haurem de desmuntar-la `sudo umount /dev/sdx`. Si dona error que el disc no està muntat; no passa res.
5. I finalment, a la carpeta on tenim el fitxer: (`if` i `of` són input/output file)

```
sudo dd bs=16M if=arxiu.img of=/dev/sdx
```

# Configuració comuna(1)

Totes les raspberrys del clúster han de tindre una configuració comuna:

1. Arrancar una raspberry (la 1) amb monitor i teclat. Tindrà xarxa per DHCP.
2. `sudo apt update` i `sudo apt upgrade`
3. Executem `sudo raspi-config` i passar per les següents parts:
  - i. Menú `1 System Options` :
    - `S4 Hostname` → `node1`
    - `S5 Boot/Autologin` → triar opció `B2 Console Autologin`
  - ii. Menú `2 Display Option` : Ací no tocarem res
  - iii. Menú `3 Interface Options` :
    - `P2 SSH` i habilitarem el login remot i el servidor SSH
  - iv. Menú `4 Performance Options` :
    - `P2 GPU Memory` i en deixem sols 16MB. Pensa que sols entrarem en la consola



# Configuració comuna (2)

Dins encara del `raspi-config` :

- Menú `5 Localization` :
  - i. `L1 Locale` : llevarem `en_GB.UTF8` i posarem el `es_ES.UTF-8`
  - ii. `L2 Timezone` : posarem Europe -> Madrid
  - iii. `L3 Keyboard` : seleccionarem Spanish
- Menú `6 Advanced Options` :
  - i. `A1 Expand Filesystem` -> per ocupar tota la sd
  - ii. `A4 Network Interface Names` -> seleccionem `no` . Per tenir `eth0` en compte de `enxb827eb9f90a4` (una combinació del literal `enx` seguit de la MAC de la interfícies)
- Menú `8 Update` no cal res, ja que hem fet el `update-upgrade` abans
- Menú `9 About` no cal res

Per últim reiniciar mitjançant un `sudo reboot`

# Configuració comuna. Instal·lar MPI (1)

1. `mkdir mpich3` La carpeta que contindrà l'instal·lador
2. `cd mpich3` Entrem i descarreguem la versió
3. `wget http://www.mpich.org/static/downloads/3.3.2/mpich-3.3.2.tar.gz`
4. Descomprimim: `tar xzf mpich-3.3.2.tar.gz`
5. `sudo mkdir /home/rpimpi`
6. `sudo mkdir /home/rpimpi/mpi-install`
7. `sudo mkdir /home/pi/mpi-build`
8. `cd /home/pi/mpi-build`
9. `sudo apt-get install gfortran`

# Configuració comuna. Instal·lar MPI (2)

Compilem MPI per a la plataforma. Aquest procés pot durar fins 1 hora

1. `sudo /home/pi/mpich3/mpich-3.2.2/configure -prefix=/home/rpimpi/mpi-install`
2. `sudo make` → compilem
3. `sudo make install` → instalem
4. afegim mpi al path: `nano .bashrc` en l'ultima linia  
`PATH=$PATH:/home/rpimpi/mpi-install/bin`
5. reiniciar → `sudo reboot`
6. Comprovar que mpi funciona: `mpiexec -n 1 hostname`

# Configuració comuna. Instal·lar MPI (3)

Instalem les llibreries per a programar en Python

1. Instal·lar les llibreries extra de python `sudo apt install python-dev`
2. Instal·lar el gestor de llibreries → `sudo apt install python-pip`
3. Instal·lar mpi4py → `pip install mpi4py`
4. Provem que funciona → `mpiexec -n 4 python helloworld.py`

```
# -*- coding: utf-8 -*-
from mpi4py import MPI
import sys

size=MPI.COMM_WORLD.Get_size()
rank=MPI.COMM_WORLD.Get_rank()
name=MPI.Get_processor_name()

print("Hello world, Soc el procés %2d de %2d al node %10s" %(rank,size,name))
```

# Clonació de la targeta SD.

En aquests apartat el que haurem de fer és :

1. Apagar la Raspberry
2. Fer un fitxer **img** a partir de la microSD.
3. A partir del nou img gravar les altres 3 targetes.

Es subministra un fitxer img creat amb tota la instal·lació ja creada. Per tant la primera part de configuració **no caldrà fer-la**.

En aquests moments és com si tinguérem 4 màquines idèntiques. Haurem de fer alguna cosa per fer-les distintes

# Configuración de cada node per separat

Hem de configurar en cada node:

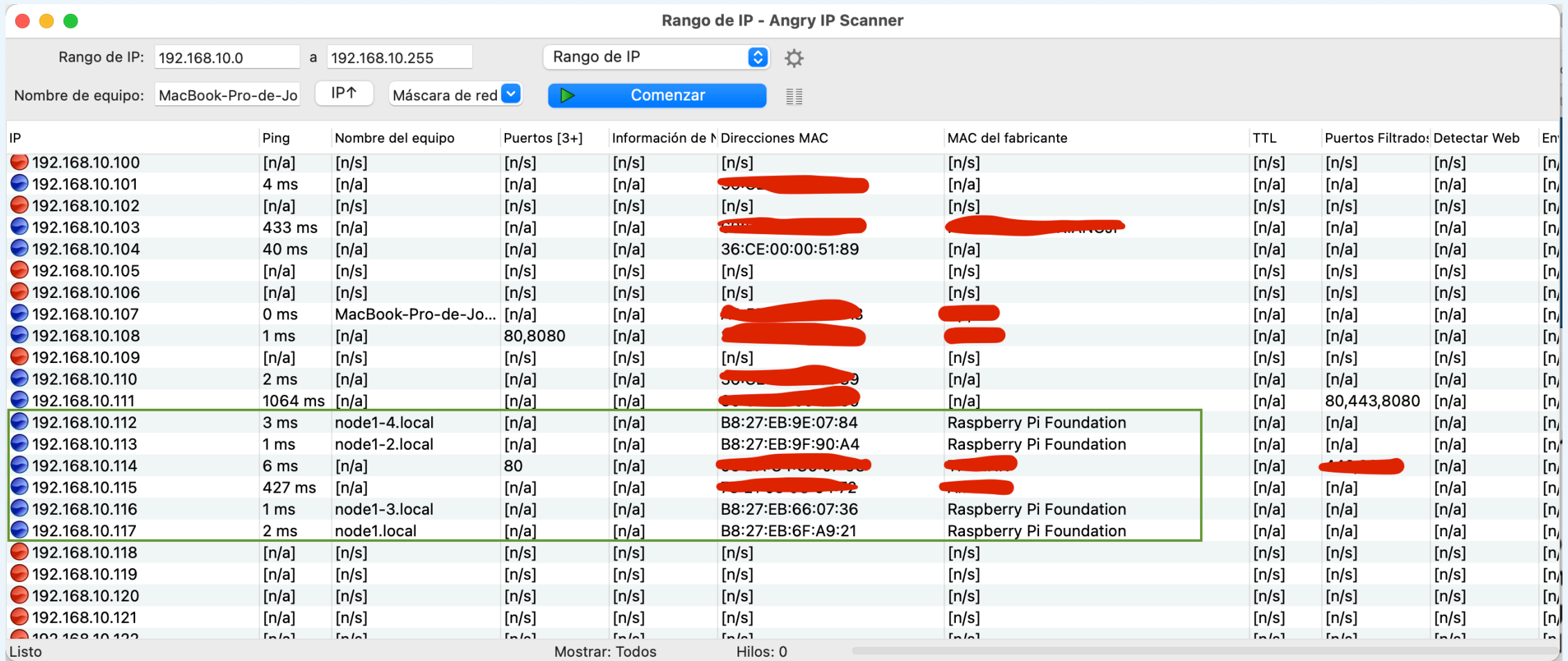
1. Adreça estàtica
2. Hostname
3. Crear clau *ssh*
4. Transferir la clau entre tots els nodes
5. Muntar una carpeta compartida amb *nfs*

Notes:

- Necessitarem un *sniffer* de la xarxa per descobrir quines IP's tenen. Es recomana <https://angryip.org>.
- Per treballar amb totes les màquines alhora, podem fer servir alguna implementació de *cluster-ssh*

# Configuración de cada node (1)

Descubrir quines adrees IP tenen cadascun dels nodes.



IP	Ping	Nombre del equipo	Puertos [3+]	Información de N	Direcciones MAC	MAC del fabricante	TTL	Puertos Filtrado:	Detectar Web	En
192.168.10.100	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.101	4 ms	[n/a]	[n/a]	[n/a]	[redacted]	[n/a]	[n/a]	[n/a]	[n/a]	[n/
192.168.10.102	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.103	433 ms	[n/a]	[n/a]	[n/a]	[redacted]	[redacted]	[n/a]	[n/a]	[n/a]	[n/
192.168.10.104	40 ms	[n/a]	[n/a]	[n/a]	36:CE:00:00:51:89	[n/a]	[n/a]	[n/a]	[n/a]	[n/
192.168.10.105	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.106	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.107	0 ms	MacBook-Pro-de-Jo...	[n/a]	[n/a]	[redacted]	[redacted]	[n/a]	[n/a]	[n/a]	[n/
192.168.10.108	1 ms	[n/a]	80,8080	[n/a]	[redacted]	[redacted]	[n/a]	[n/a]	[n/a]	[n/
192.168.10.109	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.110	2 ms	[n/a]	[n/a]	[n/a]	[redacted]	[n/a]	[n/a]	[n/a]	[n/a]	[n/
192.168.10.111	1064 ms	[n/a]	[n/a]	[n/a]	[redacted]	[n/a]	[n/a]	80,443,8080	[n/a]	[n/
192.168.10.112	3 ms	node1-4.local	[n/a]	[n/a]	B8:27:EB:9E:07:84	Raspberry Pi Foundation	[n/a]	[n/a]	[n/a]	[n/
192.168.10.113	1 ms	node1-2.local	[n/a]	[n/a]	B8:27:EB:9F:90:A4	Raspberry Pi Foundation	[n/a]	[n/a]	[n/a]	[n/
192.168.10.114	6 ms	[n/a]	80	[n/a]	[redacted]	[redacted]	[n/a]	[redacted]	[n/a]	[n/
192.168.10.115	427 ms	[n/a]	[n/a]	[n/a]	[redacted]	[redacted]	[n/a]	[n/a]	[n/a]	[n/
192.168.10.116	1 ms	node1-3.local	[n/a]	[n/a]	B8:27:EB:66:07:36	Raspberry Pi Foundation	[n/a]	[n/a]	[n/a]	[n/
192.168.10.117	2 ms	node1.local	[n/a]	[n/a]	B8:27:EB:6F:A9:21	Raspberry Pi Foundation	[n/a]	[n/a]	[n/a]	[n/
192.168.10.118	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.119	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.120	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.121	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/
192.168.10.122	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/s]	[n/

Mostrar: Todos      Hilos: 0

# Configuración de cada node (2)

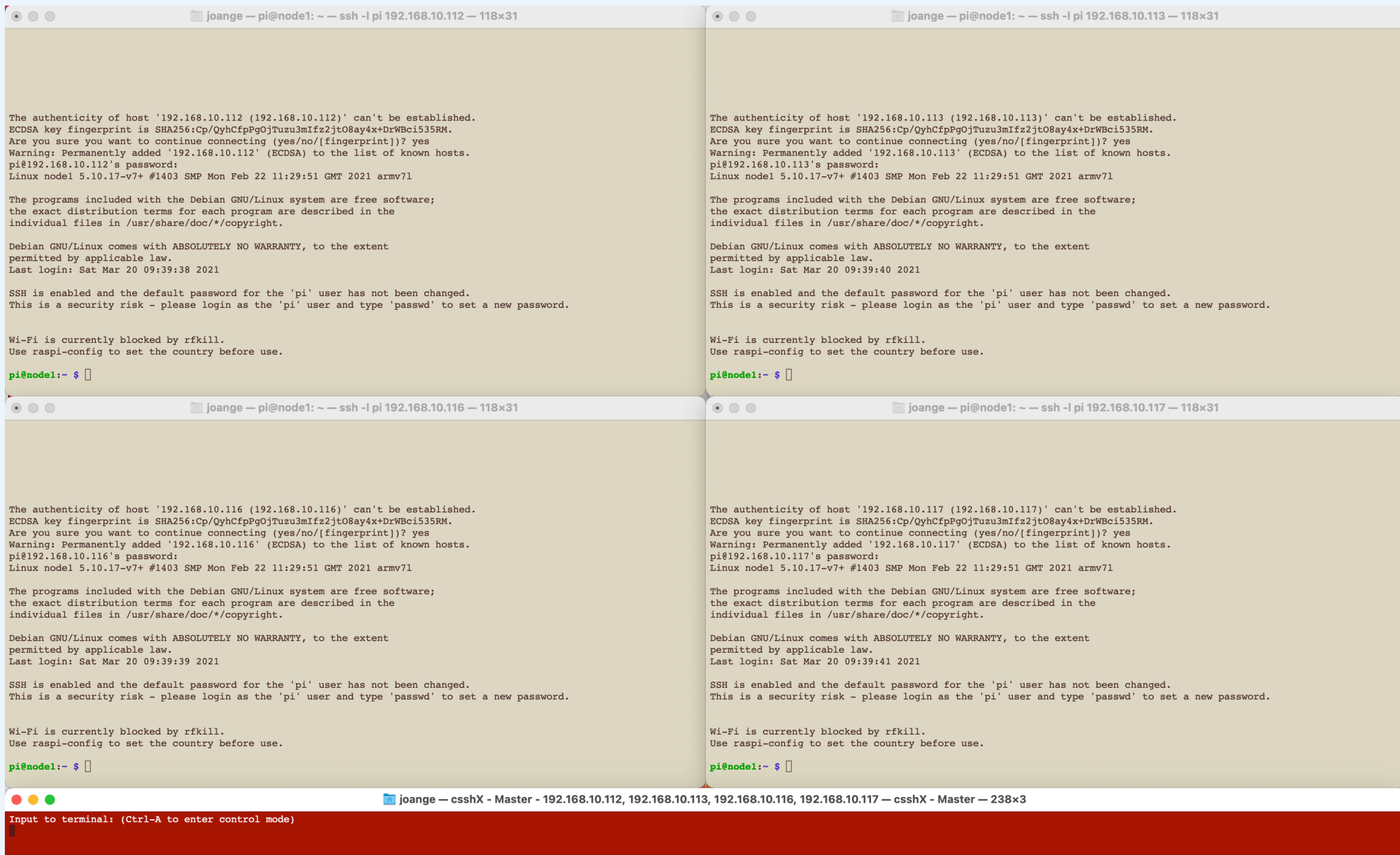
Amb alguna ferramenta com cluster-ssh ens connectem a elles

```
MacBook-Pro-de-Joan:~ joange$ cat clusterPI
192.168.10.112
192.168.10.113
192.168.10.116
192.168.10.117
MacBook-Pro-de-Joan:~ joange$ csshX --login pi --host clusterPI
```

- Hem de crear un fitxer de text amb les IP's que hem trobat anteriorment ( `nano clusterPI` a l'exemple)
- Haurem de escriure un `yes` per acceptar el *fingerprint*
- Escriure el password `raspberry`
- Des d'ubuntu `cssh -f clusterPI user@` per a connectar-nos.



# Configuración de cada node (3)



```
joange — pi@node1: ~ — ssh -l pi 192.168.10.112 — 118x31
The authenticity of host '192.168.10.112 (192.168.10.112)' can't be established.
ECDSA key fingerprint is SHA256:Cp/QyhCfpPgOjTuzu3mIfz2jt08ay4x+DrWBci535RM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.10.112' (ECDSA) to the list of known hosts.
pi@192.168.10.112's password:
Linux node1 5.10.17-v7+ #1403 SMP Mon Feb 22 11:29:51 GMT 2021 armv7l

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: Sat Mar 20 09:39:38 2021

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@node1:~$

joange — pi@node1: ~ — ssh -l pi 192.168.10.113 — 118x31
The authenticity of host '192.168.10.113 (192.168.10.113)' can't be established.
ECDSA key fingerprint is SHA256:Cp/QyhCfpPgOjTuzu3mIfz2jt08ay4x+DrWBci535RM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.10.113' (ECDSA) to the list of known hosts.
pi@192.168.10.113's password:
Linux node1 5.10.17-v7+ #1403 SMP Mon Feb 22 11:29:51 GMT 2021 armv7l

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: Sat Mar 20 09:39:40 2021

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@node1:~$

joange — pi@node1: ~ — ssh -l pi 192.168.10.116 — 118x31
The authenticity of host '192.168.10.116 (192.168.10.116)' can't be established.
ECDSA key fingerprint is SHA256:Cp/QyhCfpPgOjTuzu3mIfz2jt08ay4x+DrWBci535RM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.10.116' (ECDSA) to the list of known hosts.
pi@192.168.10.116's password:
Linux node1 5.10.17-v7+ #1403 SMP Mon Feb 22 11:29:51 GMT 2021 armv7l

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: Sat Mar 20 09:39:39 2021

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@node1:~$

joange — pi@node1: ~ — ssh -l pi 192.168.10.117 — 118x31
The authenticity of host '192.168.10.117 (192.168.10.117)' can't be established.
ECDSA key fingerprint is SHA256:Cp/QyhCfpPgOjTuzu3mIfz2jt08ay4x+DrWBci535RM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.10.117' (ECDSA) to the list of known hosts.
pi@192.168.10.117's password:
Linux node1 5.10.17-v7+ #1403 SMP Mon Feb 22 11:29:51 GMT 2021 armv7l

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: Sat Mar 20 09:39:41 2021

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@node1:~$

joange — csshX - Master - 192.168.10.112, 192.168.10.113, 192.168.10.116, 192.168.10.117 — csshX - Master — 238x3
Input to terminal: (Ctrl-A to enter control mode)
```

# Configuración de cada node (4)

Adreça estàtica - `sudo nano /etc/network/interfaces`

```
# source-directory /etc/network/interfaces.d

auto eth0
#iface eth0 inet dhcp
iface eth0 inet static
address 192.168.10.5x
netmask 255.255.255.0
gateway 192.168.10.yyy
```

## Notes:

- La primera línia la comentarem, així com afegirem una comentada per dhcp
  - la `x` serà del 1..4. Després segons el valor de x li donarem nom al hostame.
  - canviar `192.168.10` per la xarxa de l'aula adequada.
  - canviar `yyy` per la ip de la porta d'enllaç adequada.
- **Atenció:** com muntarem dos clústers, un que comencé en la 100 i altre en la 200.

# Configuración de cada node (5)

## Nom del host - `sudo nano /etc/hostname`

Per a identificar cada màquina dins de cada clúster i en tota l'aula, li posarem com a nom `node-X-Y` on:

- `X` serà el número de clúster
- `Y` serà el número de node dins del clúster.

Dins de cada fitxer `/etc/hostname` contindrà el seu nom de host.

```
192.168.10.51 ↔ node-1-1
192.168.10.52 ↔ node-1-2
192.168.10.53 ↔ node-1-3
192.168.10.54 ↔ node-1-4
```

# Configuración de cada node (6)

## Equips del clúster - `sudo nano /etc/hosts`

Aquest fitxer conté el llistat de tots els nodes del clúster, amb el seu nom i la seua IP. Això ens servirà per fer servir noms en compte d'adreces IP.

```
127.0.0.1      localhost
               [ . . . ]
192.168.10.51  node-1-1
192.168.10.52  node-1-2
192.168.10.53  node-1-3
192.168.10.54  node-1-4
```

Reiniciar tot el cluster i fer proves de `ping`. Hauras de netejar el teu fitxer de hosts conegut del ordinador des d'on ens connectem, ja que hem canviat la IP:

```
rm ~/.ssh/known_hosts
```

# Primera prova d'execució (1)

Anem a treballar sols amb el `node1`

1. Crec carpeta `mpi_test`: `mkdir mpi_test`
2. Moure o crear amb l'editor nano el programa d'abans: `mv | nano  
helloworldmpi.py mpi_test/`
3. Entrem i crear el següent `machinefile`. Aquest fitxer conte una linia per cada node del cluster, amb la sintaxi `node[:cores]`

```
node-1-1:4  
node-1-2:4  
node-1-3:4  
node-1-4:4
```

El numeret després dels dos punts indica els nuclis d'execució que supporta cada node.

# Primera prova d'execució (2)

```
pi@node1:~/mpi_test $ mpiexec -f machinefile -n 4 python helloworldmpi.py
Hello world, I'm process 0 of 4 in processor node-1-1
Hello world, I'm process 1 of 4 in processor node-1-1
Hello world, I'm process 2 of 4 in processor node-1-1
Hello world, I'm process 3 of 4 in processor node-1-1
```

Tot be, ja que cada node té 4 cores, i s'executa tot en el `node1` . Però que passa si fem:

```
pi@node1:~/mpi_test $ mpiexec -f machinefile -n 16 python helloworldmpi.py
Host key verification failed.
Host key verification failed.
Host key verification failed.
```

**Falla.** No tenim accés a la resta de nodes a mitjançant la xarxa. MPI es comunica per `ssh` . Anem a solucionar-ho.

# Configuración de cada node (7)

1. Hem d'aconseguir poder fer login automàticament desde qualssevol node a qualssevol altre node.
2. Per aixó farem en `node-1-x` ( $x=1..4$ )

```
# generem les claus SSH
ssh-keygen                # acceptar tot per defecte
cd ~/.ssh
cp id_rsa.pub node-1-x_key # x=1..4
```

Exportem en tots els nodes les claus dels altres tres. Exemple de importació de la clau de `node-1-1` als altres tres:

```
scp node-1-1:/home/pi/.ssh/node-1-1_key .      # vigila el últim .
cat node-1-1_key>>authorized_keys
```

El comandament transfereix la clau pública del node 1 a la màquina des d'on ho hem executat.

# Configuración de cada node (8)

```
joange — pi@node1: ~/.ssh — ssh — 118x31
pi@node1:~/.ssh $ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDOuXvReWpOLdcZcgC9qMwsmQilZLm1cU2UphQYHsHBoAz77FQH0x1jvD1s3rTb4A2Px1Dq46wei2AWal
OqmlYjvmLFldcSM7SjxkQilT4ky9meJHC7znVsB1NyaxKuRu51UrFcuSnmA+mz2MjrHwOIxsbNOWh0ivurOagO3d0iMPf1CQ4ob2W6JqLWSfOfBiapxIDz
RzfxyKPzjEqmq0fpihEdQ1AQPuIOqbf4S2LUr9CN1UCqodW2i4gkRR3sWgdngY3lVZ50aavLaJkIooIFNJF5gxXHMcUanYICfaChazSybUvMUz3VH9Xvdu
9bZ2Y8FOuyYUohxcHL3cAar49H pi@node2
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDJMfjL281Vh2rHteaScmUYtlQ9P2WttBiQ9KMjaLcXtKhb8Wit3mR7m1FBqPLw9dit2cGyUahgOm50rg
nW21Vlan7XefBZ9BUXn0BiIOYlp/8S8RNwkJnpZTFR0ZEOPohlmuglcWPQrSCWLlrt9nwCmLmDLFjt6xVDE9bJce20kNIBJfO6PxTzvH0H2tgGCRax6pNg
aop65fe29gPLL8yliwl2FGkDAz4L90FiBENLsdOidQmY9y5p7NwSRjQpPFA4EZfAY0ul8TnGANYG92I0ZSmyj9eNQiUv8HoUMNvEvgeEPkuXVK/1R8BHpP
Bxljwfl4JEG24RB7x4NfQFylLD pi@node3
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDI53Jn7002NncTrLmNob9qyLwIW8afNwxhl1TrVewdN2pg2pVrP+FBNWXzNx6tVMkcsM0hD+4UA1szXto
f8OMsDvc0KJgZS9L9N3Rkk1yEweoEW2f1RFYDcZ25UpzJbv5yqYhTspTN6tbSDOd+pyfHULrcJ/hCPdkmf78eDBGulJm0SGN5tLW4KjJY17yi0qD1cf5IR
2rDTmU/XSJ9XjG445kDQ5ZwiYEFK71ekJnb8MbD9bv9v006rApBB8gK/FXHv0FA68xgOwblIbrx1GpkARK12aTsamu2oIM5HvOcBxsa7Y75U/BBNjqGmVO
SMi6LX8EQqZbjVgI/3EefvzsUJ pi@node4
pi@node1:~/.ssh $

joange — pi@node2: ~/.ssh — ssh -l pi 192.168.10.52 — 118x31
pi@node2:~/.ssh $ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDYiO0BJdMd3ALiWQxB/nSprk4c/FEIiP55j67OWtQEbSurhf76yTbYwrR8WRbiFgS1IKjVhyNhRYD72j
uTTemODSruZgMUvpadMr0u8c9cEMld+vlgIiFM8eLohgE/3vrlh91mYGxtR0qplw94Xdgmuiv34CaAXA4hxt1cDb+V0YY4yUVA7D6kFNM8eJPghWQHbBn
rXjJpujw5lASPTj/hjkPmdEUHk4ztiPkPagXVEuVvXWjAW0eOnMq3jqx39GnolCbmRVNqaxpmJZUZfIW+m5d6TuAo9ngtCYLotGonOtyR5Rg81jewM1HTj
PWgAWNL2h0y74RUyysbXy0UDgf pi@node1
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDJMfjL281Vh2rHteaScmUYtlQ9P2WttBiQ9KMjaLcXtKhb8Wit3mR7m1FBqPLw9dit2cGyUahgOm50rg
nW21Vlan7XefBZ9BUXn0BiIOYlp/8S8RNwkJnpZTFR0ZEOPohlmuglcWPQrSCWLlrt9nwCmLmDLFjt6xVDE9bJce20kNIBJfO6PxTzvH0H2tgGCRax6pNg
aop65fe29gPLL8yliwl2FGkDAz4L90FiBENLsdOidQmY9y5p7NwSRjQpPFA4EZfAY0ul8TnGANYG92I0ZSmyj9eNQiUv8HoUMNvEvgeEPkuXVK/1R8BHpP
Bxljwfl4JEG24RB7x4NfQFylLD pi@node3
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDI53Jn7002NncTrLmNob9qyLwIW8afNwxhl1TrVewdN2pg2pVrP+FBNWXzNx6tVMkcsM0hD+4UA1szXto
f8OMsDvc0KJgZS9L9N3Rkk1yEweoEW2f1RFYDcZ25UpzJbv5yqYhTspTN6tbSDOd+pyfHULrcJ/hCPdkmf78eDBGulJm0SGN5tLW4KjJY17yi0qD1cf5IR
2rDTmU/XSJ9XjG445kDQ5ZwiYEFK71ekJnb8MbD9bv9v006rApBB8gK/FXHv0FA68xgOwblIbrx1GpkARK12aTsamu2oIM5HvOcBxsa7Y75U/BBNjqGmVO
SMi6LX8EQqZbjVgI/3EefvzsUJ pi@node4
pi@node2:~/.ssh $

joange — pi@node3: ~/.ssh — ssh — 118x31
pi@node3:~/.ssh $ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDYiO0BJdMd3ALiWQxB/nSprk4c/FEIiP55j67OWtQEbSurhf76yTbYwrR8WRbiFgS1IKjVhyNhRYD72j
uTTemODSruZgMUvpadMr0u8c9cEMld+vlgIiFM8eLohgE/3vrlh91mYGxtR0qplw94Xdgmuiv34CaAXA4hxt1cDb+V0YY4yUVA7D6kFNM8eJPghWQHbBn
rXjJpujw5lASPTj/hjkPmdEUHk4ztiPkPagXVEuVvXWjAW0eOnMq3jqx39GnolCbmRVNqaxpmJZUZfIW+m5d6TuAo9ngtCYLotGonOtyR5Rg81jewM1HTj
PWgAWNL2h0y74RUyysbXy0UDgf pi@node1
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDOuXvReWpOLdcZcgC9qMwsmQilZLm1cU2UphQYHsHBoAz77FQH0x1jvD1s3rTb4A2Px1Dq46wei2AWal
OqmlYjvmLFldcSM7SjxkQilT4ky9meJHC7znVsB1NyaxKuRu51UrFcuSnmA+mz2MjrHwOIxsbNOWh0ivurOagO3d0iMPf1CQ4ob2W6JqLWSfOfBiapxIDz
RzfxyKPzjEqmq0fpihEdQ1AQPuIOqbf4S2LUr9CN1UCqodW2i4gkRR3sWgdngY3lVZ50aavLaJkIooIFNJF5gxXHMcUanYICfaChazSybUvMUz3VH9Xvdu
9bZ2Y8FOuyYUohxcHL3cAar49H pi@node2
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDJMfjL281Vh2rHteaScmUYtlQ9P2WttBiQ9KMjaLcXtKhb8Wit3mR7m1FBqPLw9dit2cGyUahgOm50rg
nW21Vlan7XefBZ9BUXn0BiIOYlp/8S8RNwkJnpZTFR0ZEOPohlmuglcWPQrSCWLlrt9nwCmLmDLFjt6xVDE9bJce20kNIBJfO6PxTzvH0H2tgGCRax6pNg
aop65fe29gPLL8yliwl2FGkDAz4L90FiBENLsdOidQmY9y5p7NwSRjQpPFA4EZfAY0ul8TnGANYG92I0ZSmyj9eNQiUv8HoUMNvEvgeEPkuXVK/1R8BHpP
Bxljwfl4JEG24RB7x4NfQFylLD pi@node3
pi@node3:~/.ssh $

joange — pi@node4: ~/.ssh — ssh -l pi 192.168.10.54 — 118x31
pi@node4:~/.ssh $ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDYiO0BJdMd3ALiWQxB/nSprk4c/FEIiP55j67OWtQEbSurhf76yTbYwrR8WRbiFgS1IKjVhyNhRYD72j
uTTemODSruZgMUvpadMr0u8c9cEMld+vlgIiFM8eLohgE/3vrlh91mYGxtR0qplw94Xdgmuiv34CaAXA4hxt1cDb+V0YY4yUVA7D6kFNM8eJPghWQHbBn
rXjJpujw5lASPTj/hjkPmdEUHk4ztiPkPagXVEuVvXWjAW0eOnMq3jqx39GnolCbmRVNqaxpmJZUZfIW+m5d6TuAo9ngtCYLotGonOtyR5Rg81jewM1HTj
PWgAWNL2h0y74RUyysbXy0UDgf pi@node1
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDOuXvReWpOLdcZcgC9qMwsmQilZLm1cU2UphQYHsHBoAz77FQH0x1jvD1s3rTb4A2Px1Dq46wei2AWal
OqmlYjvmLFldcSM7SjxkQilT4ky9meJHC7znVsB1NyaxKuRu51UrFcuSnmA+mz2MjrHwOIxsbNOWh0ivurOagO3d0iMPf1CQ4ob2W6JqLWSfOfBiapxIDz
RzfxyKPzjEqmq0fpihEdQ1AQPuIOqbf4S2LUr9CN1UCqodW2i4gkRR3sWgdngY3lVZ50aavLaJkIooIFNJF5gxXHMcUanYICfaChazSybUvMUz3VH9Xvdu
9bZ2Y8FOuyYUohxcHL3cAar49H pi@node2
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDJMfjL281Vh2rHteaScmUYtlQ9P2WttBiQ9KMjaLcXtKhb8Wit3mR7m1FBqPLw9dit2cGyUahgOm50rg
nW21Vlan7XefBZ9BUXn0BiIOYlp/8S8RNwkJnpZTFR0ZEOPohlmuglcWPQrSCWLlrt9nwCmLmDLFjt6xVDE9bJce20kNIBJfO6PxTzvH0H2tgGCRax6pNg
aop65fe29gPLL8yliwl2FGkDAz4L90FiBENLsdOidQmY9y5p7NwSRjQpPFA4EZfAY0ul8TnGANYG92I0ZSmyj9eNQiUv8HoUMNvEvgeEPkuXVK/1R8BHpP
Bxljwfl4JEG24RB7x4NfQFylLD pi@node3
pi@node4:~/.ssh $
```

En `authorized_keys` tenim les claus ssh dels altres tres nodes. Ja podem entrar (login) sense problema



# Primera prova d'execució (3)

Tornem al prova, des del `node-1-1`

```
pi@node1:~/mpi_test $ mpiexec -f machinefile -n 16 python helloworldmpi.py  
[proxy:0:3@node-1-4] launch_procs ... /home/pi/mpi_test (No such file or directory)
```

***Altre Error !!!*** El que indica el següent missatge és que quan estavem en `node-1-4` (o altre node del clúster), ha intentat entrar en la carpeta `/home/pi/mpi_test` i no la troba.

**Problema** → El codi font ha d'estar en la mateixa ruta en tots els nodes

## Possibles solucions:

- Cada cop que editem el codi, l'haurem de copiar en totes els nodes. Implica copiar-ho per a cada modificació
- Creem una carpeta en xarxa, compartida per tots els nodes per NFS. **Aquesta és la bona** 👍

# Configuración de cada node (9)

La manera de treball serà:

## **node-1-1:**

- Executarà un servidor de `nfs`, i exportarem (compartirem) una carpeta per xarxa
- Com a programadors treballarem exclusivament en aquest node.

## **resta de nodes:**

- Monterem durant l'arranc la carpeta en xarxa que existeix en `node-1-1`
- Estaran en funcionament, però no treballarema amb ells. Seràn invocats per `node-1-1` al moment d'executar algun programa.

# Configuración de cada node (10)

## Instalar servidor nfs i compartir carpeta: (en node-1-1)

- `sudo apt install nfs-kernel-server`
- `mkdir /home/pi/cloud`
- `sudo nano /etc/exports` i afegir la següent línia

```
/home/pi/cloud *(rw,sync,no_root_squash,no_subtree_check)
```

- `sudo chmod -R 777 /home/pi/cloud`
- `sudo update-rc.d rpcbind enable`
- `sudo service rpcbind restart`
- `sudo exportfs -a`
- `sudo service nfs-kernel-server restart`

# Configuración de cada node (11)

## Instalar client `nfs` i montar carpeta - (en `node-1-x`, $x=2,3,4$ )


- `sudo mkdir /home/pi/cloud`
- `sudo nano /etc/fstab` i afegir la linia:

```
node-1-1:/home/pi/cloud /home/pi/cloud nfs auto,noatime,nolock,bg,nfsvers=4,intr,actimeo=1800 0 0
```

- Reiniciem en tots els `node-1-x` per comprovar que ho automonta

```
pi@node4:~/cloud $ df -h
```

S.ficheros	Tamaño	Usados	Disp	Uso%	Montado en
/dev/root	14G	1,8G	12G	13%	/
devtmpfs	455M	0	455M	0%	/dev
tmpfs	487M	0	487M	0%	/dev/shm
tmpfs	487M	13M	475M	3%	/run
tmpfs	5,0M	4,0K	5,0M	1%	/run/lock
tmpfs	487M	0	487M	0%	/sys/fs/cgroup
/dev/mmcblk0p1	253M	49M	204M	20%	/boot
node1:/home/pi/cloud	14G	1,8G	12G	13%	/home/pi/cloud
tmpfs	98M	0	98M	0%	/run/user/1000



# Primera prova d'execució (4)

- En `node-1-1` movem el programa `helloworldmpi.py` i el `machinefile` de `mpi_test` a `cloud`, que serà ja la carpeta de treball.

```
pi@node1:~/cloud $ mpiexec -f machinefile -n 16 python helloworldmpi.py
Hello world, Soc el procés 0 de 16 a node-1-1
Hello world, Soc el procés 8 de 16 a node-1-3
Hello world, Soc el procés 12 de 16 a node-1-4
Hello world, Soc el procés 4 de 16 a node-1-2
Hello world, Soc el procés 1 de 16 a node-1-1
Hello world, Soc el procés 9 de 16 a node-1-3
Hello world, Soc el procés 13 de 16 a node-1-4
Hello world, Soc el procés 5 de 16 a node-1-2
Hello world, Soc el procés 2 de 16 a node-1-1
Hello world, Soc el procés 10 de 16 a node-1-3
Hello world, Soc el procés 14 de 16 a node-1-4
Hello world, Soc el procés 6 de 16 a node-1-2
Hello world, Soc el procés 3 de 16 a node-1-1
Hello world, Soc el procés 11 de 16 a node-1-3
Hello world, Soc el procés 15 de 16 a node-1-4
Hello world, Soc el procés 7 de 16 a node-1-2
```

# Primera prova d'execució (i 5)

Observem que:

- S'han executat 4 tasques en cada node, gràcies a la configuració del `machinefile` (registres `node-1-1:4` ?)
- El funcionament és anàrquic: molts processos fent coses alhora, i no hi ha control del que ix alhora
- Fes distintes proves en múltiples de 4

## Fi de la primera part

**Tornem i a programar...**