

# Computación en la Nube

---

- **Nombre:** Pierre Simon Callist Yannick Tondreau
- **Repositorio Git:** [https://github.com/PierreSimT/pr\\_cn/tree/master/p2](https://github.com/PierreSimT/pr_cn/tree/master/p2)
- **Máster Ingeniería Informática - Universidad de La Laguna**

## Ejercicio 1

**Despliega en el laas de la ULL un cluster de 8 nodos con un core cada uno con la posibilidad de establecer comunicaciones entre ellos (port tcpip).**

Se ha realizado el despliegue de ocho máquinas virtuales con las siguientes características:

- Sistema Operativo: CentOS 7
- CPU: 1 Core | 1 Hilo | 1 Socket
- RAM: 4GB
- Almacenamiento: 20GB

Las máquinas desplegadas tienen los siguientes nombres y direcciones IP:

- master: 192.168.210.100
- slave-1: 192.168.210.101
- slave-2: 192.168.210.102
- slave-3: 192.168.210.103
- slave-4: 192.168.210.104
- slave-5: 192.168.210.105
- slave-6: 192.168.210.106
- slave-7: 192.168.210.107

Los programas se encuentran compartidos mediante NFS desde el **master** al resto de máquinas en el directorio **/mnt/nfs**.

## Ejercicio 2

**Repite el ejercicio 5 de la práctica anterior en este cluster para analizar el rendimiento de las comunicaciones mediante el programa ping-pong.**

Programa **prod.c**

```
> mpirun -np 1 --hostfile hosts prod.run
Process 0 of 1 on master
wall clock time = 3.621597, Prod time: 0.0000000036215965, x =
1000000000.000000
```

El programa se ejecuta desde la máquina **master** y se ejecuta en su mismo procesador, podemos esperar el mismo resultado desde el resto de máquinas dado que tienen el mismo procesador.

## Programa `ptop.c`

```
> mpirun -np 2 --hostfile hosts ptop.run
```

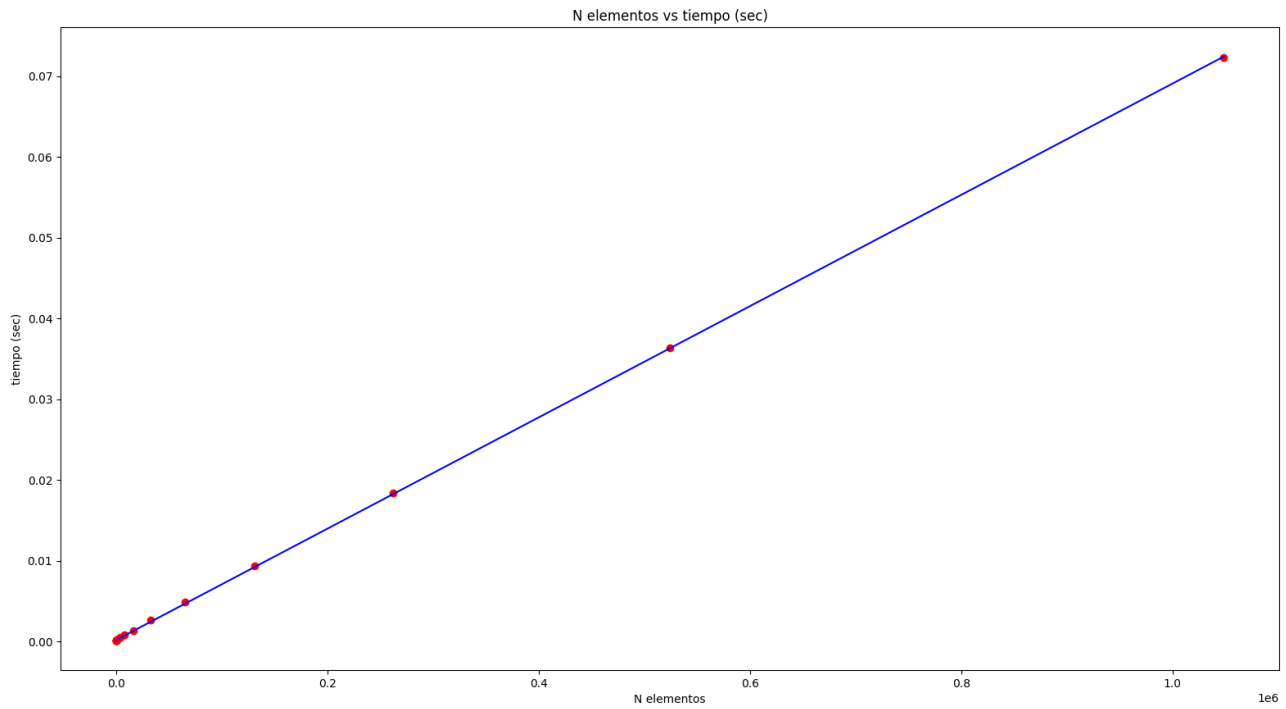
Procesador: master 0

Procesador: slave-1 1

| Kind      | n       | time (sec) | MB / sec   |
|-----------|---------|------------|------------|
| Send/Recv | 1       | 0.000077   | 0.103492   |
| Send/Recv | 2       | 0.000083   | 0.192134   |
| Send/Recv | 4       | 0.000083   | 0.384610   |
| Send/Recv | 8       | 0.000071   | 0.901309   |
| Send/Recv | 16      | 0.000075   | 1.697554   |
| Send/Recv | 32      | 0.000079   | 3.239210   |
| Send/Recv | 64      | 0.000085   | 6.006419   |
| Send/Recv | 128     | 0.000098   | 10.496266  |
| Send/Recv | 256     | 0.000146   | 14.004680  |
| Send/Recv | 512     | 0.000147   | 27.958186  |
| Send/Recv | 1024    | 0.000189   | 43.285972  |
| Send/Recv | 2048    | 0.000268   | 61.207640  |
| Send/Recv | 4096    | 0.000543   | 60.327560  |
| Send/Recv | 8192    | 0.000860   | 76.212051  |
| Send/Recv | 16384   | 0.001317   | 99.524783  |
| Send/Recv | 32768   | 0.002639   | 99.317510  |
| Send/Recv | 65536   | 0.004870   | 107.653015 |
| Send/Recv | 131072  | 0.009342   | 112.248796 |
| Send/Recv | 262144  | 0.018384   | 114.072775 |
| Send/Recv | 524288  | 0.036383   | 115.282007 |
| Send/Recv | 1048576 | 0.072293   | 116.036137 |

Comparando estos resultados con la ejecución en la máquina local, se puede destacar que la velocidad de transferencia de datos es más lenta dado que viene capada por el ancho de banda de la red.

Aún así, obteniendo la gráfica de  $N/\text{Tiempo (sec)}$  se obtiene una regresión lineal en la cual a medida que sube la cantidad de datos, el tiempo de ejecución de la operación aumenta.



Una vez calculada la regresión lineal, obtenemos los datos del regresor que se ha creado en Python:

```
Coeficientes:  
[6.88993213e-08]  
Mean squared error: 1.1683436424267604e-08  
Coeficiente de determinacion: 0.9999588707962632
```

Y la recta vendrá dada por la siguiente función:

$$y = 0.00000006889932129445x + 0.00016855806895378818$$