

SISTEMAS COMPUTADORES DE ALTAS PRESTACIONES

PRACTICA 1



Universidad
de Huelva

ALEJANDRO GORDILLO PEDRAZA

ÍNDICE

SISTEMAS COMPUTADORES DE ALTAS PRESTACIONES	1
PRACTICA 1	1
ÍNDICE	2
RESUMEN	3
Arrancar VirtualBox y configurar nuestra primera máquina.	4
Maquina Padre	4
Configuración de las maquinas	4
Configuración de Red	5
Configurar el clúster	6
OpenMosix View	7
Opciones OpenMosix	8
Pruebas de Rendimiento	10
Conclusión	14

RESUMEN

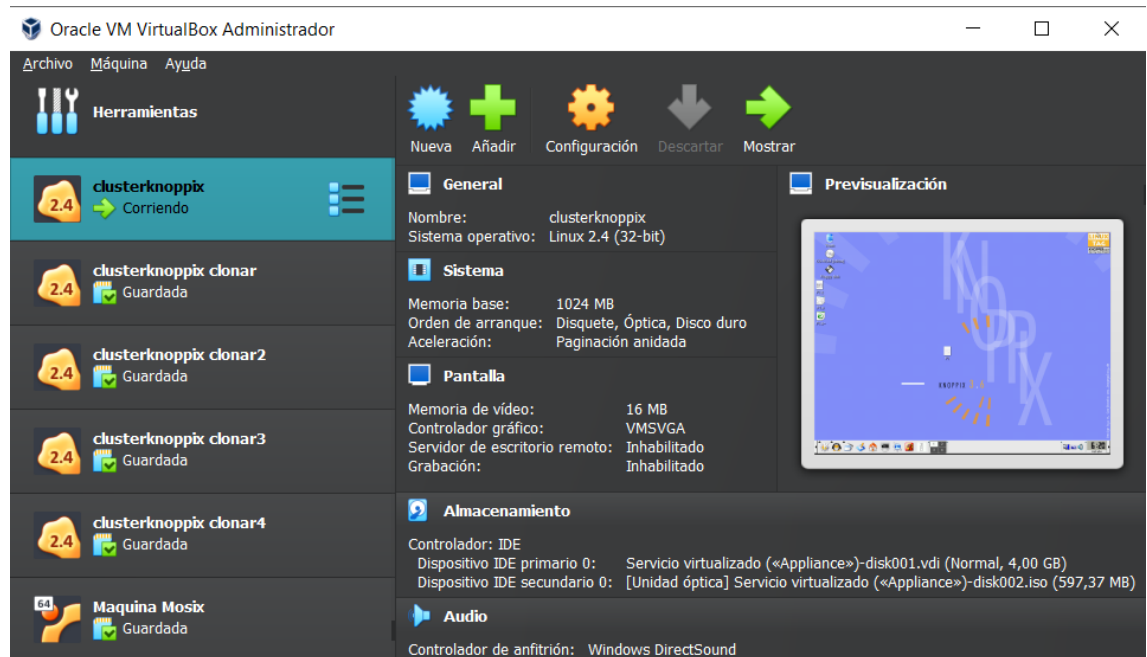
Practica 1, en esta practica deberemos tratar con el sistema Cluster Knoppix, el cual es un sistema de Linux basado en Debian. Con este sistema deberemos crear 5 maquinas virtuales iguales en virtualbox.

Estas mismas maquinas trabajaran entre si permitiéndonos diversificar la carga de trabajo entre ellas, haciendo que nuestro trabajo se realice de una forma mas eficiente que si trabajáramos con un único computador.

Deberemos crear las maquinas y configurarlas para tras hacer las pruebas comprobar que esto se cumpla.

Arrancar VirtualBox y configurar nuestra primera máquina.

Para ello iniciaremos virtualbox y instalaremos nuestro sistema knoppix, una vez instalado haremos 4 clonaciones de este. Para así tener el clúster de 5 equipos que deseamos en esta práctica.



Maquina Padre

Seleccionamos la primera maquina que hemos creado como maquina padre, para esto no tendremos que hacer nada especial, solo tener en cuenta cual es la maquina seleccionada.

Configuración de las maquinas

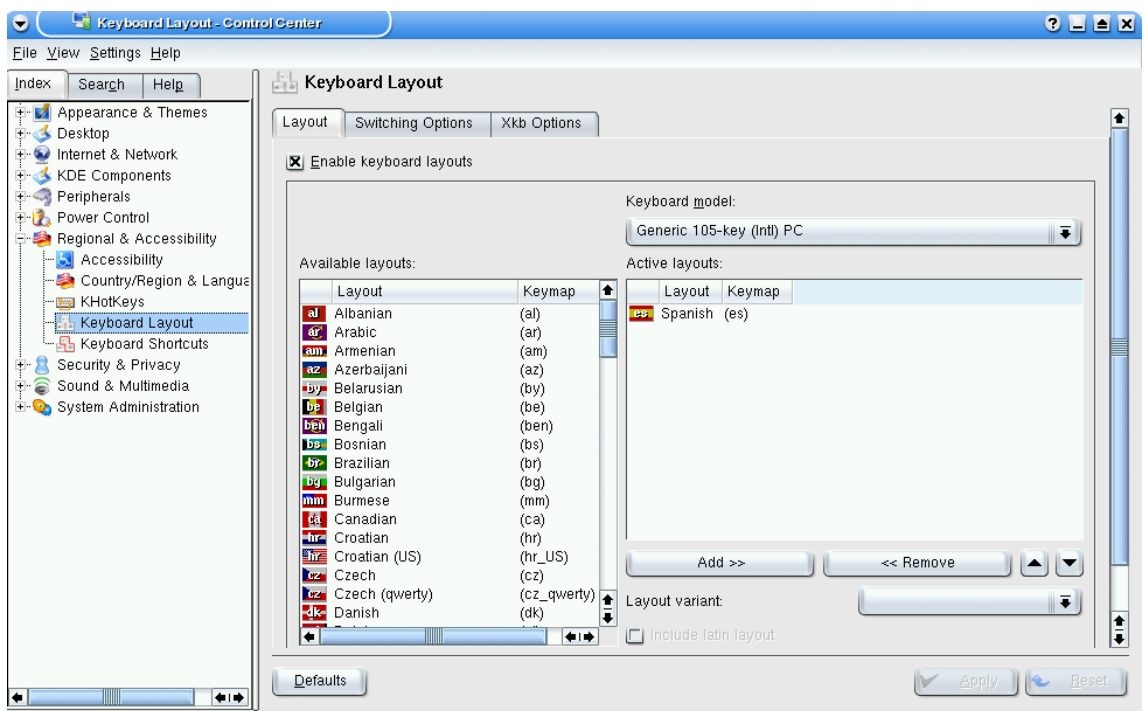
La primera configuración que deberemos darle a todas las maquinas es el idioma que vamos a manejar, para ello al arrancar la maquina meteremos el siguiente comando de arranque:

knoppix noscsi

Una vez arrancada nuestra maquina procederemos con el cambio de idioma. Por defecto nos traerá 3 idiomas, pero nosotros los borraremos y añadiremos el español. Esto nos permitirá tener el teclado de la maquina en nuestro idioma y distribución nativa.

Para esto deberemos seguir los siguientes pasos:

- **Menú KDE ◇ Settings ◇ Control Center**
- **Regional Accesibility**
- **Keyboard Layout**



Configuración de Red

El siguiente paso es llevar a cabo la configuración de la red, para poder tener acceso a Internet y lo que es más importante, agregar todos los nodos del clúster en la misma red. Para acceder a la tarjeta de red debemos seguir los siguientes pasos:

- **KNOPPIX – NETWORK/INTERNET- NETWORK CARD CONFIGURATION**
- **DHCP: No**
- **IP address: 192.168.1.X**
- **Network Mask: 255.255.255.0**

- Broadcast Address: 192.168.1.255
- Gateway: 192.168.1.254
- DNS: 192.168.1.254

Con esto ya tenemos configurada la red de la máquina para que funcione el clúster de OpenMosix.

Para comprobar que todo haya surtido efecto comprobaremos con el comando `ifconfig` en la terminal los datos de la red.

```

knoppix@ttyp0[knoppix]$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:1B:F8:E1
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:80167 errors:0 dropped:0 overruns:0 frame:0
          TX packets:84235 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7444857 (7.0 MiB)  TX bytes:6758844 (6.4 MiB)
          Base address:0xd020  Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:6724 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6724 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:647654 (632.4 KiB)  TX bytes:647654 (632.4 KiB)

knoppix@ttyp0[knoppix]$

```

Repetiremos estos pasos en cada maquina virtual, cambiando las respectivas direcciones IP de cada una.

Configurar el clúster

El siguiente paso consiste en añadir los diferentes equipos virtualizados al clúster, para ello deberemos usar una función que trae de serie nuestro sistema a la que llaman el demonio.

Esta función se llama `omdiscd`.

Para utilizarla deberemos abrir el terminal y escribir el comando:

Su omdiscd

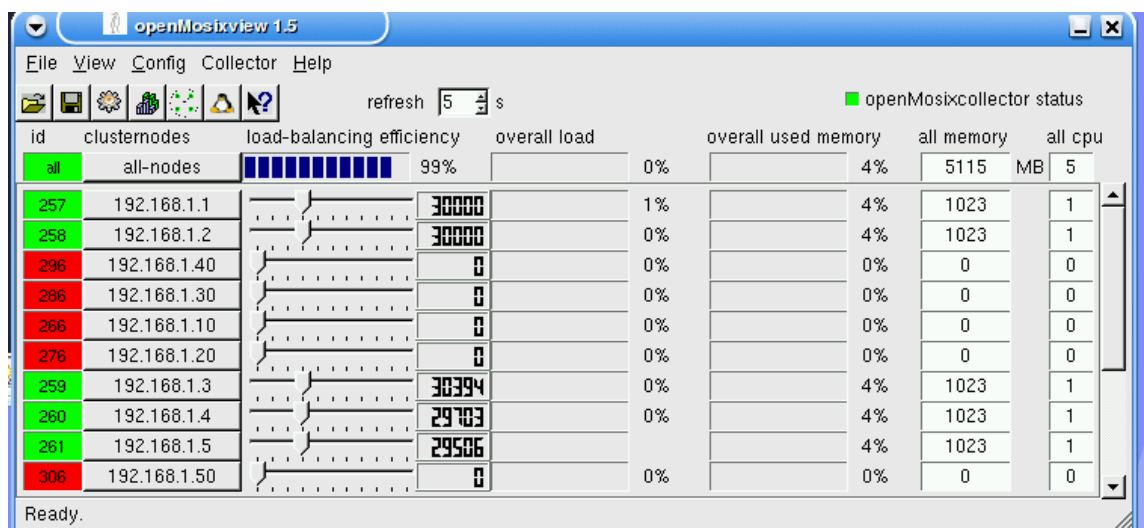
Este comando habrá que ponerlo en cada maquina que queramos añadir al clúster. En el proceso de añadir equipos al clúster no tendremos ninguna respuesta visual por parte del programa, pero estará añadiéndolos.

OpenMosix View

Una vez tenemos todos los equipos deseados añadidos al clúster, podremos hacer uso de una herramienta que trae el sistema por defecto. Esta herramienta es openmosix view, la herramienta nos permitirá ver los equipos que forman el clúster y configurarlos a nuestro gusto. Para así repartir la carga de reparto como queramos y hacer diferentes pruebas con los mismos.

Para utilizar la herramienta, simplemente abrimos una ventana de la terminal y ponemos el siguiente comando:

Su openmosixview



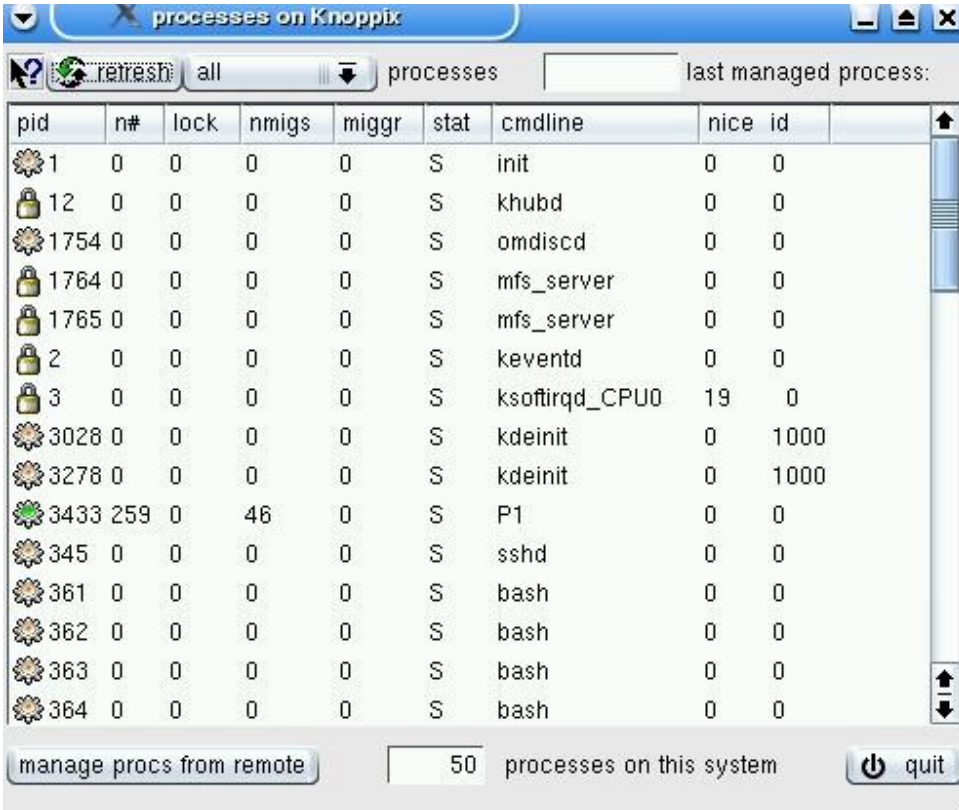
Como podéis apreciar aparecen mas nodos de los que queríamos incluir en nuestro clúster, esto es debido a que más compañeros estaban conectados a la misma red de internet a la hora de crear el clúster y los reconoció el sistema. Pero solo nos saldrán en verde en nuestro caso los 5 que pertenecen a nuestro sistema nativo. Siempre y cuando no compartamos red de internet con estos compañeros.

Opciones OpenMosix

OpenMosixProcs:

Esta opción es el 3º botón que hay en la fila, al lado de guardar, y nos permitirá recibir información acerca de los procesos que se ejecutan en cada nodo.

En naranja aparecerán los procesos que ejecutamos en el nodo padre y en verde los que hemos migrado a otro nodo. También aparecerán algunos con una llave, esto significa que esos procesos están bloqueados.

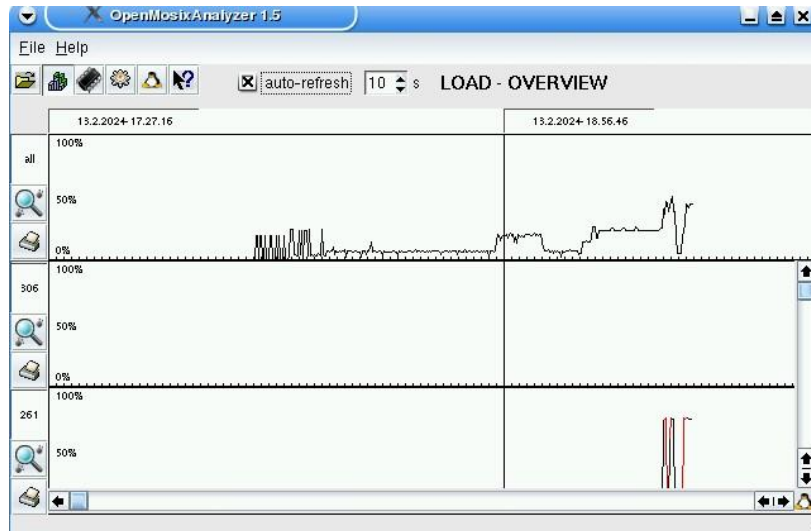


The screenshot shows a window titled "processes on Knoppix". At the top, there is a toolbar with a "refresh" button, a dropdown menu set to "all", and a "processes" button. To the right of these is a text field labeled "last managed process:". Below the toolbar is a table with the following columns: pid, n#, lock, nmigs, miggr, stat, cmdline, nice, and id. The table lists various system processes, some with orange flower icons and others with green flower icons. At the bottom of the window, there is a "manage procs from remote" button, a text field containing "50", and a "processes on this system" button. To the right of these is a "quit" button with a power icon.

pid	n#	lock	nmigs	miggr	stat	cmdline	nice	id
1	0	0	0	0	S	init	0	0
12	0	0	0	0	S	khubd	0	0
1754	0	0	0	0	S	omdiscd	0	0
1764	0	0	0	0	S	mfs_server	0	0
1765	0	0	0	0	S	mfs_server	0	0
2	0	0	0	0	S	keventd	0	0
3	0	0	0	0	S	ksoftirqd_CPU0	19	0
3028	0	0	0	0	S	kdeinit	0	1000
3278	0	0	0	0	S	kdeinit	0	1000
3433	259	0	46	0	S	P1	0	0
345	0	0	0	0	S	sshd	0	0
361	0	0	0	0	S	bash	0	0
362	0	0	0	0	S	bash	0	0
363	0	0	0	0	S	bash	0	0
364	0	0	0	0	S	bash	0	0

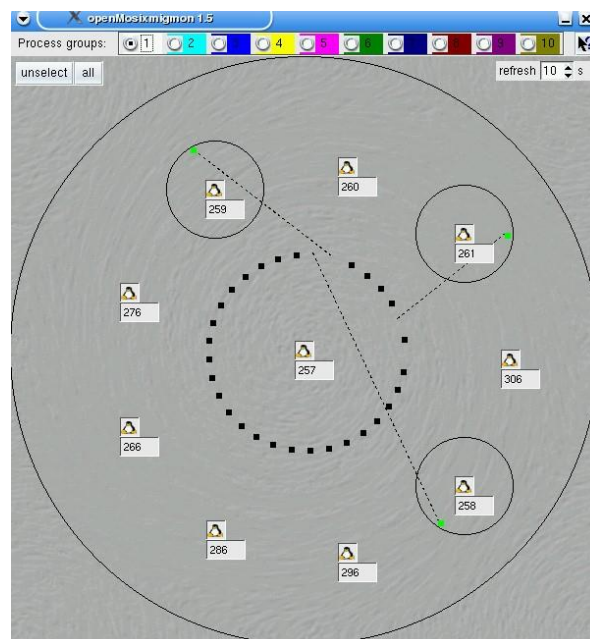
OpenMosixAnalyzer:

Esta opción es el 4 boton de la fila, y nos permite ver un historial de la carga de trabajo del cluster, destacando que si en algún momento pasamos del 75% la línea pasara a ser roja en vez de negra.



OpenMosixMigmom:

Esta opción será el 5º botón y nos mostrará un círculo de pingüinos, con uno en el centro que representa nuestro nodo padre. Del Pingüino central saldrá una línea apuntando a diferentes pingüinos con un círculo, esto representa como va migrando trabajo entre nodos.



Pruebas de Rendimiento

Para las pruebas que vamos a realizar, hemos usado un pequeño código en C que nos permitirá crear un bucle en segundo plano que le provoque estrés a nuestra maquina y así necesite hacer uso del cluster.

```
#include  
  
int main ()  
{  
    int i;  
    while (1)  
    {  
        i=1;  
    }  
    return 0;  
}
```

Este código lo meteremos en un txt y lo compilaremos con la terminal del SO.

```
Gcc -c p1.c
```

```
Gcc -o P1 p1.o
```

Una vez lo tengamos ya compilado como P1 podremos ejecutarlo con el comando

```
./P1 &
```

Para ello deberemos estar en el mismo directorio que el archivo, y en nuestro caso no lo hemos ejecutado 1 vez, si no que lo hemos ejecutado 3 veces simultáneamente para provocar una carga un poco más elevada de trabajo.

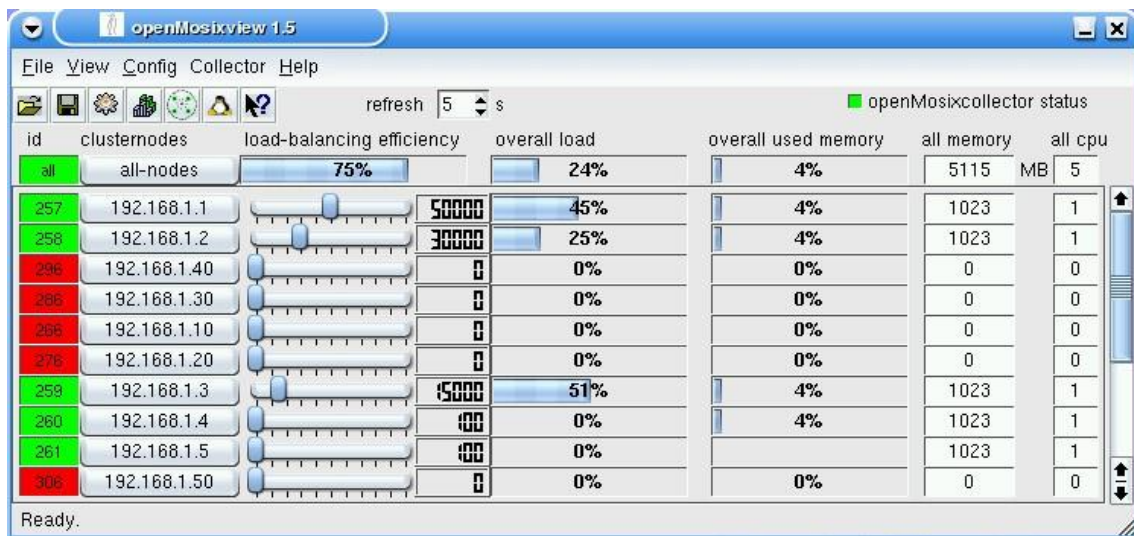
1. Cambiar la eficiencia de balanceo de carga de un nodo.
 - a. Para realizar esta prueba he cambiado la velocidad de procesamiento en los diferentes nodos haciendo algunas comprobaciones, para ver como cambiaba la carga de trabajo entre los nodos según su poder de procesamiento.

Trata de 2 pruebas, la primera se realizo dejando el nodo principal en 50k y dándole al Noda 2 30K y al Nodo 3 15k, dejando los otros dos nodos con 100 de speed.

Sudo mosctl setspeed 30000

Esto haría que los dos nodos restantes sean tan insignificantes que nuestro cluster no migraría procesos a ellos.

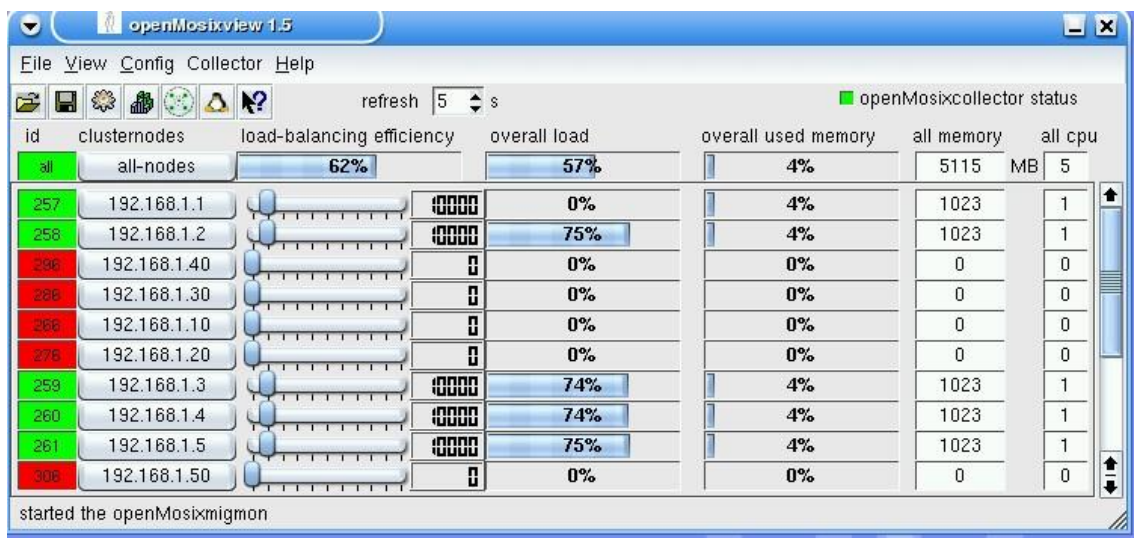
Con esta prueba quería comprobar si hiciera uso de los nodos con mas potencia o si los intentara hacer a partes iguales. Pero para mi sorpresa era todo un poco aleatorio, había veces que trabajaba mas el Nodo 1 y veces que trabajaba más el Nodo 2.



- b. Viendo el resultado de la prueba anterior dándole diferentes capacidades a los nodos, quise hacer la prueba dándole la misma capacidad a todos los nodos. Incluido el nodo padre, para ello les día a todos 10k de capacidad.

Sudo mosctl setspeed 10000

La conclusión tras hacer esto fue, que seguía derivando según necesitara el trabajo a diferentes nodos y rara vez trabajaban todos a la vez, pero había una diferencia esta vez. Y es que el nodo padre a veces no trabajaba y solo lo hacían los nodos hijos. Pero cuando esto pasaba los nodos hijos tenían una carga de trabajo muy parecida entre ellos.



2. Apagamos 1 Nodo.

Hacemos la prueba apagando 1 de los nodos, en nuestro caso hemos apagado el Nodo 5. Y el resultado nos aporta 2 casos de resultado:

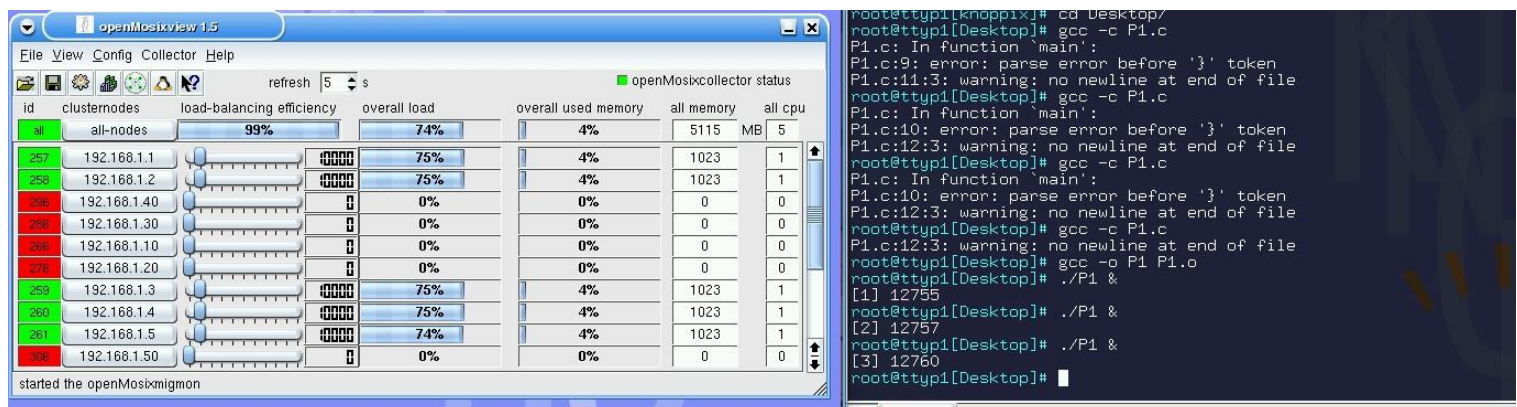
- Apagamos el nodo durante la ejecución del bucle, esto provoca que redistribuya la carga entre 3 nodos y no entre los 4 dejando 1 nodo libre por si hiciera falta en algún momento, o esa es la conclusión que yo saque de este aspecto.

- Apagamos 1 Nodo sin estar activo el bucle y entonces al arrancar el programa y haría uso de todos los nodos, no dejando ninguno libre en este caso.

3. Ejecutamos el programa en 1 Nodo

Hasta ahora las pruebas de rendimiento del cluster y su funcionamiento lo habíamos hecho ejecutando el programa en el nodo Padre, entonces pensé que debería probar si el cluster funciona correctamente, si en vez de ejecutarlo el padre lo hace uno de los hijos, hablando de la carga de trabajo.

Cogi el mismo programa del bucle infinito y lo ejecute en 1 de los nodos hijos y el cluster funcionaba perfectamente, seguía repartiendo la carga de trabajo sin ningún problema.



Conclusión

En esta practica hemos podido aprender un poco más acerca de distribuciones de Linux y ver mas de cerca como funciona un cluster, como este ayuda a reducir la carga de trabajo y como de interesante puede ser trabajar con uno de estos. Por supuesto nosotros lo hemos hecho a una escala muy pequeña, pero nos permite hacernos una idea de como funcionaria con una carga más grande y a mayor escala.

Gracias a esta practica he atado un poco mejor algunos conceptos de la asignatura y también de mi vida cotidiana que no llegaba a entender, y creo que ahora entiendo un poco mejor ese funcionamiento.