CLÚSTER DE ALTO RENDIMIENTO CON OPENMOSIX

Jesús Miguel
Muñoz Rodríguez

Proyecto Integrado 2º ASI

16/12/11

Índice de contenido

Clúster de alto rendimiento con OpenMosix

Introducción

El objetivo de nuestro proyecto consiste en implementar un cluster de alto rendimiento mediante el manejo de máquinas de recursos básicos. Pudiendo llegar a crear con ellas una supercomputadora con capacidad de cálculo alta y bajo coste.

En estos tipos de clústers, la carga se reparte entre los disitntos nodos para que el tiempo de proceso sea menor. Siempre sería 1/n(nodos).

Los clúster surgieron como solución a ciertos problemas que se presentaban a la hora del cómputo de alto nivel. Había empresas que necesitaban equipos que pudieran realiza tareas pesadas en un corto espacio de tiempo; y esto collevaba a que lo costos de compra y mantenimiento de equipos con estás características eran muy altos y costosos. De ahí surgió la idea de conectar dos equipos medios y convertirlos en uno solo, con la capacidad de cómputo sumáda de los dos.

Lo cierto es que el origen de las tecnologías de clúster no se sabe con seguridad, aunque está aproximado a los años 50 y principios de los 60.

Hacia el año 1967 Gene Amdahl de IBM, publico la conocida ley de Amdahl, la cual describía matemáticamente el aceleramiento que se puede dar a una tarea cuando la paralelizamos. Esta ley se convirtió en la base de la computación de multiprocesador y clúster.

Actualmente los clúster son ampliamente utilizados para proyectos con gran capacidad de cálculo, por lo que suelen estar limitados a universidades y centros de proceso de cálculo a gran escala.

Hay una curiosidad en cuanto a los clúster interesante: aprovechando el concepto de computación distribuida, la universidad de Berkley en California (EE.UU.) ha creado el proyecto SETI para buscar vida extraterrestre. Este proyecto consiste en que cualquiera que quiera participar en él puede descargar un pequeño software e instalarlo en su equipo. Este sofware se activa cuando no se está usando el equipo y convierte a este en un nodo del clúster el cual procesa imágenes y datos obtenidos por telescopios y sondas en el espacio, mandando los resultados obtenidos a la central del proyecto. Con esto se intenta que el procesamiento de los datos, el cual sería muy pesado para un clúster convencional, crezca de manera espectacular debido a los millones de personas que participan en el proyecto a pequeña escala. (Para más información se puede consultar la página del proyecto: http://setiathome.ssl.berkeley.edu/)

Conceptos básicos

Qué es un clúster de alto rendimiento

Un clúster es un conjunto de computadoras que trabajan como una única, conectadas entre sí por una red. Se conectan de forma coordinada y centralizada para procesar una mayor carga que la que podría soportar una máquina sola.

Cada máquina por separado se denomina nodo, por lo que para que el clúster funciona ha de tener 2 o más nodos conectados entre sí.

Para llevar a cabo esto se requiere un parche en el kernel (lo que hace que prácticamente casi todos los clúster que existen el mercado sean bajo el núcleo de unix). Este parche hace que el sistema reconozca las computadoras añadidas como parte del clúster y que se pueda llevar a cabo la migración y el balanceo de los procesos. También permite que se lleve una monitorización de los diferentes procesos que realiza el clúster así como de la gestión de memoria y del procesador.

Clúster de alto rendimiento con OpenMosix

Existen dos tipos de clúster: homogéneo y heterogéneno.

Clúster homogéneo

Son clúster en los que todos los nodos tienen las mismas características de hardware y sofware. Son idénticos y por lo tanto la capacidad de procesamiento y rendimiento de cada nodo es la misma.

Clúster heterogéneo

Al contrario que los anteriores, en estos tipos de clúster los nodos son completamente distintos en cuanto a hardware y software se refiere. Esto conlleva a que las posibilidades de expansión del clúster crezcan de forma exponencial, debido a que es más facil conseguir computadoras con carecterísticas distintas que muchas con iguales carácterísticas.

Estos tipos de clúster presentan una escabilidad (capacidad de un sitema de crecer o acomodarse a las exigencias del usuario o del administrador del mismo) pasmosa. Esta puede ser de forma elevada, cuando se añaden nodos para formar parter del clúster; o de forma disminuida, cuando se quitán nodos. Con openmosix se pueden llegar a tener hasta 65536 nodos trabajando de forma simultánea en el clúster.

Alto rendimiento y balanceo de carga

Cuando se habla de un clúster de alto rendimiento, es inevitable referirse también al balanceo de carga que este lleva a cabo. Con el fin de mejorar la capacidad de procesamiento que va a llegar a tener nuestro clúster, se lleva a cabo una tarea de blanceo la cual reparte los procesos de forma completa o por partes a los distintos nodos del clúster, respetando la carga que ya tenga cada nodo. Esta tarea también se denomina paralelización.

Clúster transparentes y no transparentes

Los clúster no transparentes deben tener una configuración paralela previamente predeterminada. También se tiene que conocer previamente la topología con la cual funcionará el clúster y además la utilización de unas librerías para el paso de mensajes entre las diferentes tareas. Un ejemplo de ellos es Beowulf.

Los clúster transparentes no tienen que tener la configuración parallela previamente predeterminada, ni tampoco se tiene por qué conocer la topología con la cual se va a funcionar. Ofrecen una forma mucho más cómoda a la hora de configurar el clúster, así como del balanceo de la carga. Con este tipo, uno no se tiene que preocupar de paralelizar los procesos, puesto que se hace de forma automática. Este tipo es el que vamos a comprobar con OpenMosix.

Flops

Un flop es la medida utilizada para las operaciones de coma flotante por segundo. Mide la velocidad del procesamiento numérico del procesador. Se utiliza en unidades de millones de flops (MegaFlops), miles de millones de flops (GigaFlops), etc...

Beowulf

Un Beowulf es una clase de computador masivamente paralelo de altas prestaciones principalmente construido a base de un clúster de componentes hardware estándard. Beowulf

ejecuta un sistema operativo de libre distribución como Linux o FreeBSD, y se interconecta mediante una red privada de gran velocidad. Los nodos en el clúster de computadoras no se hayan en los puestos de trabajo de los usuarios, sino que están totalmente dedicados a las tareas asignadas al clúster.

El software puede ejecutarse más rápido en un Beowulf si se dedica algún tiempo a reestructurar los programas. En general es necesario partirlos en tareas paralelas que se comunican usando alguna librería como MPI o PVM, o sockets como SysV o IPC. Con ellas se permite el hecho de partir los procesos a realizar en partes más pequeñas las cuales son controladas por un nodo master, el cuál reparte la carga entre los demás nodos en función de la capacidad de cómputo que tengan y de la carga que tengan en ese determinado momento.

En 1994, Donald Becker y Thomas Sterling construyeron la primera Beowulf. Fue construida con 16 computadores personales con procesadores Intel DX4 de 200 MHz, que estaban conectados a través de un switch Ethernet. El rendimiento teórico era de 3,2 Gflops.

Debido a que requiere de conocimientos en cuanto a programación y paralelización de tareas su uso se limita a sistemas muy grandes, como los clúster de centros de alto rendimiento para procesamiento de datos así como ciertas universidades.

Openmosix

Openmosix es un proyecto que surgió de la separación de los dos principales desarrolladores de mosix: Ammon Barak: actual desarrollador principal de mosix; y Moshe Bar: quien empezó Openmosix bajo licencia GPL.

Openmosix es un sistema de clúster para linux que consiste en un parche en el kernel responsable de las migraciones transparentes de procesos, y unas herramientas de área de usuario, necesarias para calibrar y administrar el clúster. Esto permite que no tengamos que reprogramar nuestras aplicaciones para que aprovechen el clúster.

Los procesos no saben en qué nodo del clúster se ejecutan, y es el propio openMosix el responsable de "engañarlos", y redirigir las llamadas del sistema al nodo del clúster en el que se lanzó el proceso el cual actúa en ese momento de master.

El parche para el kernel funciona en las versiones 2.4 y 2.6, aunque en esta última solo de forma experimental. Actualmente el desarrollo del proyecto está parado.

Las ventajas de utilizar openmosix frente a otras arquitecturas de clúster es que no hay que preocuparse por agregar librerías, no es necesario programar las aplicaciones y cuenta con un demonio para descubrir nodos de forma automática denominado omdiscd.

Omdiscd se encarga de crear automáticamente una lista con las máquinas existentes en la red, y de estar escuchando en caso de que haya otros demonios de detección de nodos; así como de informar al kernel sobre los nodos operativos para la migración de procesos.

Algunas desventajas que ofrece openmosix en comparación a otros es que, como se mencionó anteriormente tiene un núcleo dependiente; y como no funciona totalmente en el núcleo 2.6 puede dar problemas para el hardware que no soporte la versión de kernel 2.4.

Encuanto a la estructura interna de openmosix utiliza un sistema de archivos propio denominado oMFS (openMosix File System) que permite que todos los nodos de un clúster tengan acceso al sistema de archivos de todos los otros nodos

Cluster-knoppix

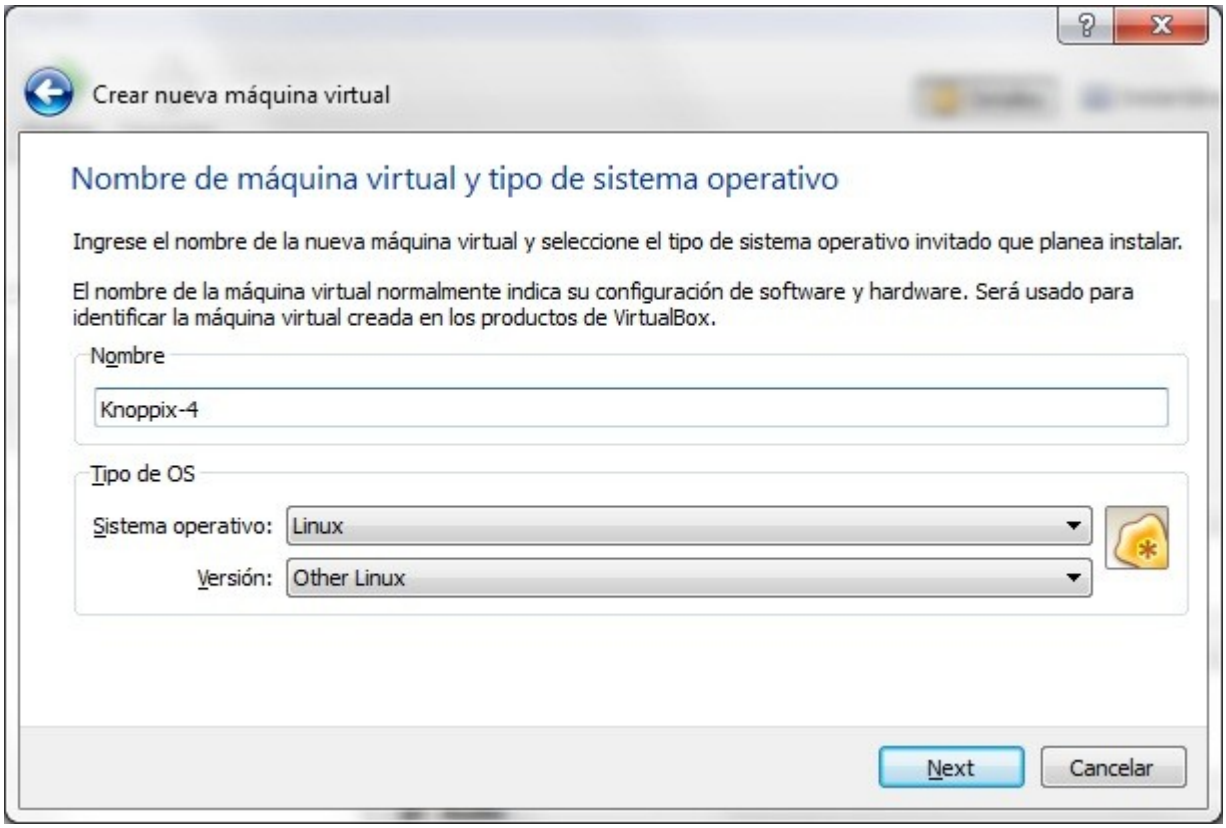
Para la realización de nuestro proyecto vamos a usar una distro de linux ya preparada para el clústering con openmosix denominada Cluster-knoppix.

ClusterKnoppix es una derivada de Knoppix. Provee a los usuarios todas las prestaciones de Knoppix (la abundancia de aplicaciones, booteo desde el CD live, auto detección del hardware, y el soporte para muchos periféricos y dispositivos) junto con las capacidades del clúster de openMosix. ParallelKnoppix, PlumpOS, Quantian, y CHAOS son algunas de las distribuciones de linux para hacer un clúster, pero ClusterKnoppix es probablemente la más popular por su sensillez de uso y configuración. Vaeamos como se instala y configura a fin de crear el clúster.

Instalación de cluster-knoppix

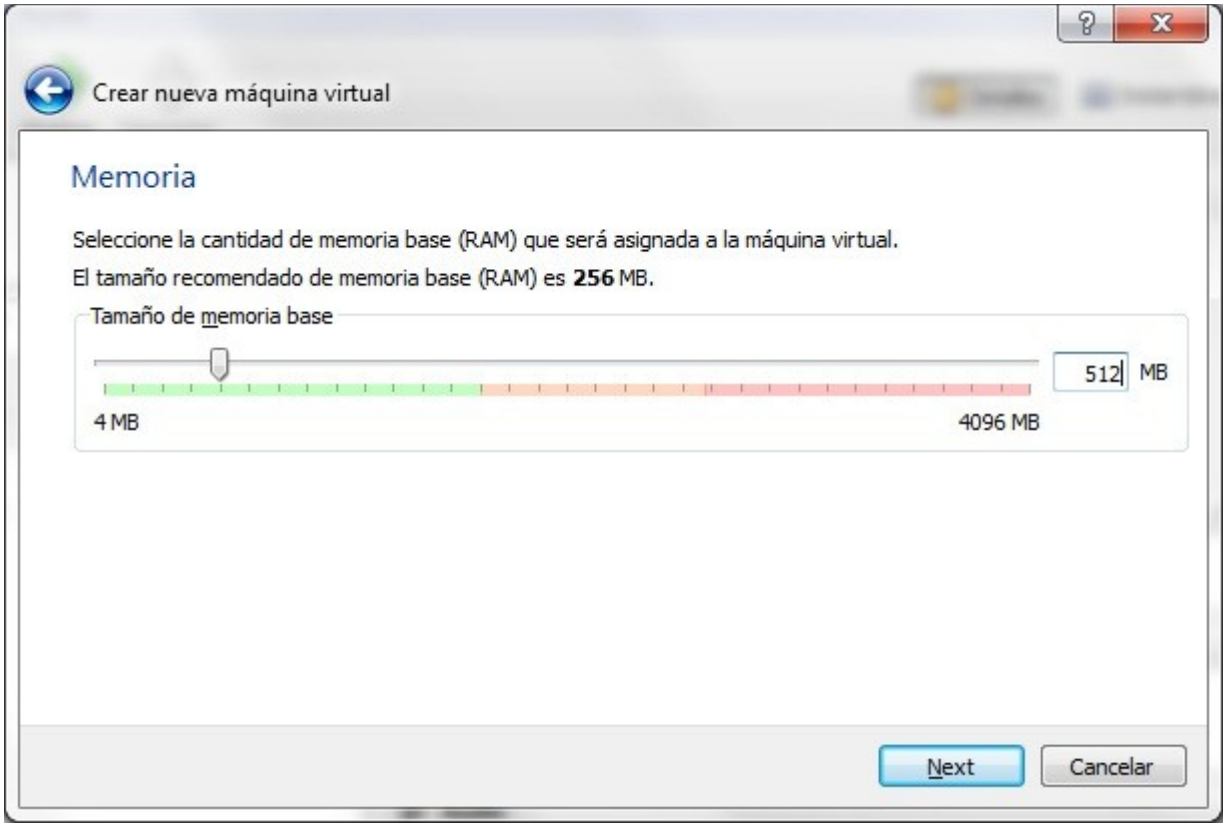
Pasamos a lo que nos ocupa: ¿Cómo crear el clúster? Lo primero que hemos de hacer es instalar las máquinas que van a ser nodos del clúster. Para ello utilizamos el live cd de cluster-knoppix el cual podemos descargarlo de este link: http://linux.softpedia.com/dyn-postdownload.php?p=2063&t=0&i=1

1. Como en nuestro caso lo vamos a hacer mediante máquinas virtuales con VirtualBox pasaremos a configurar una máquina llamada knoppix-4:

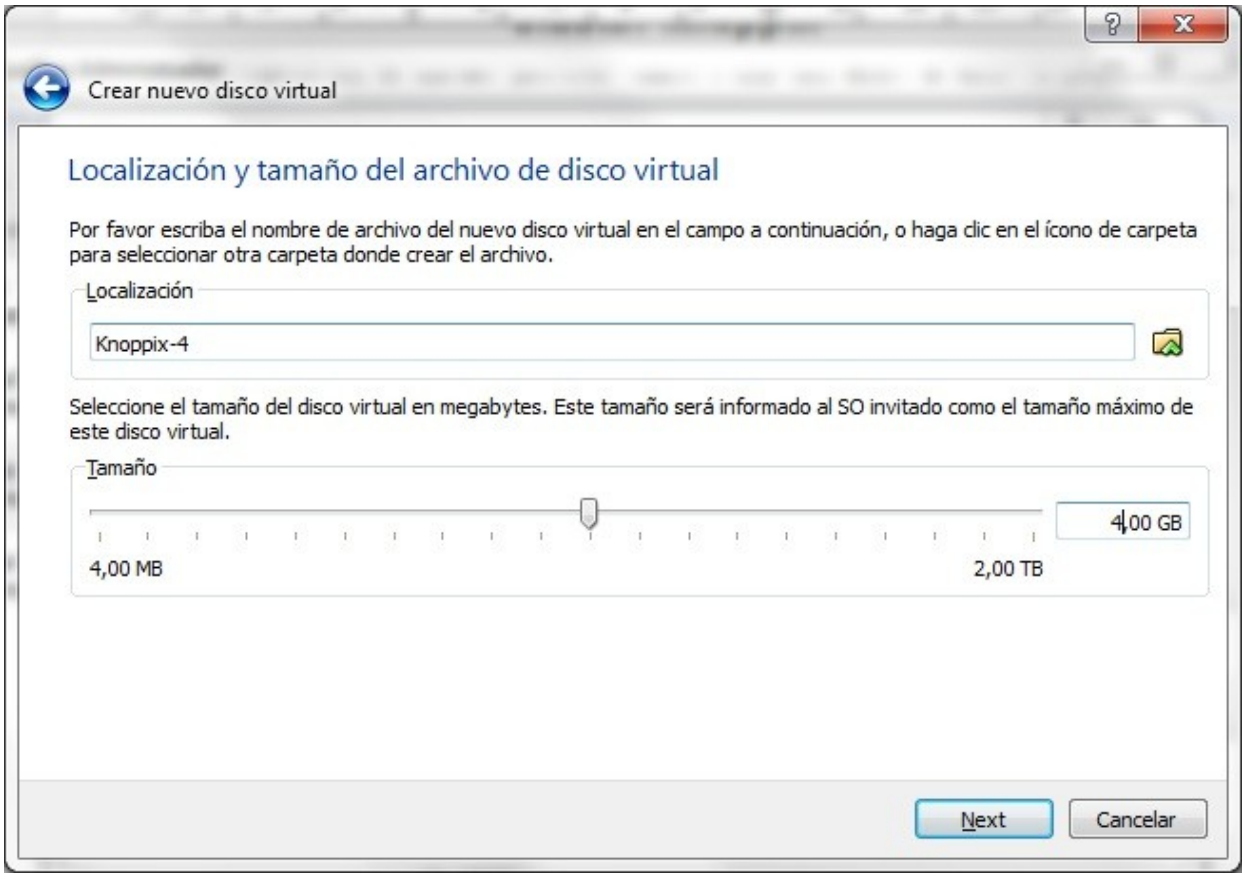


La configuraremos como SO Linux y la distribución otras, debido a que cluster-knoppix pude funcionar con el kernel 2.4 o 2.6 respectivamente. Así no tendíamos problemas de compatibilidad al usar uno u otro.

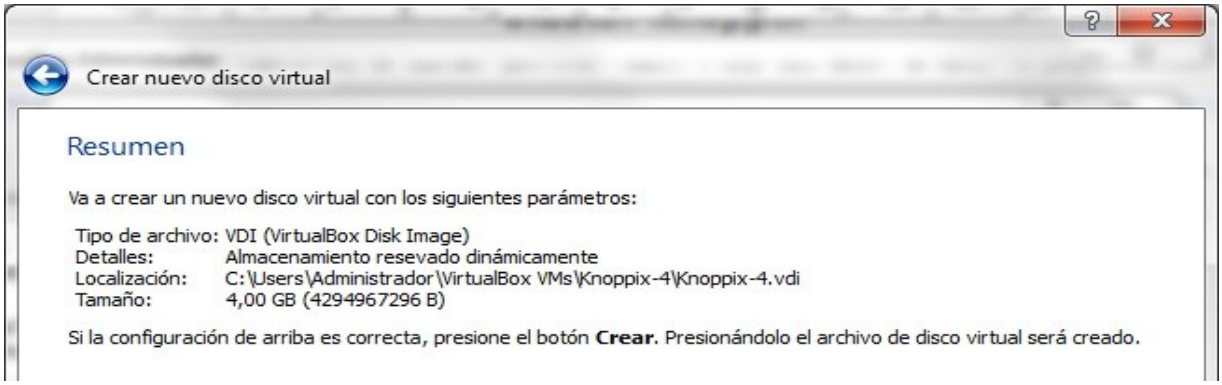
2. Ahora pasamos a asignar la memoria RAM para nuestra máquina, en este caso será de 512M:



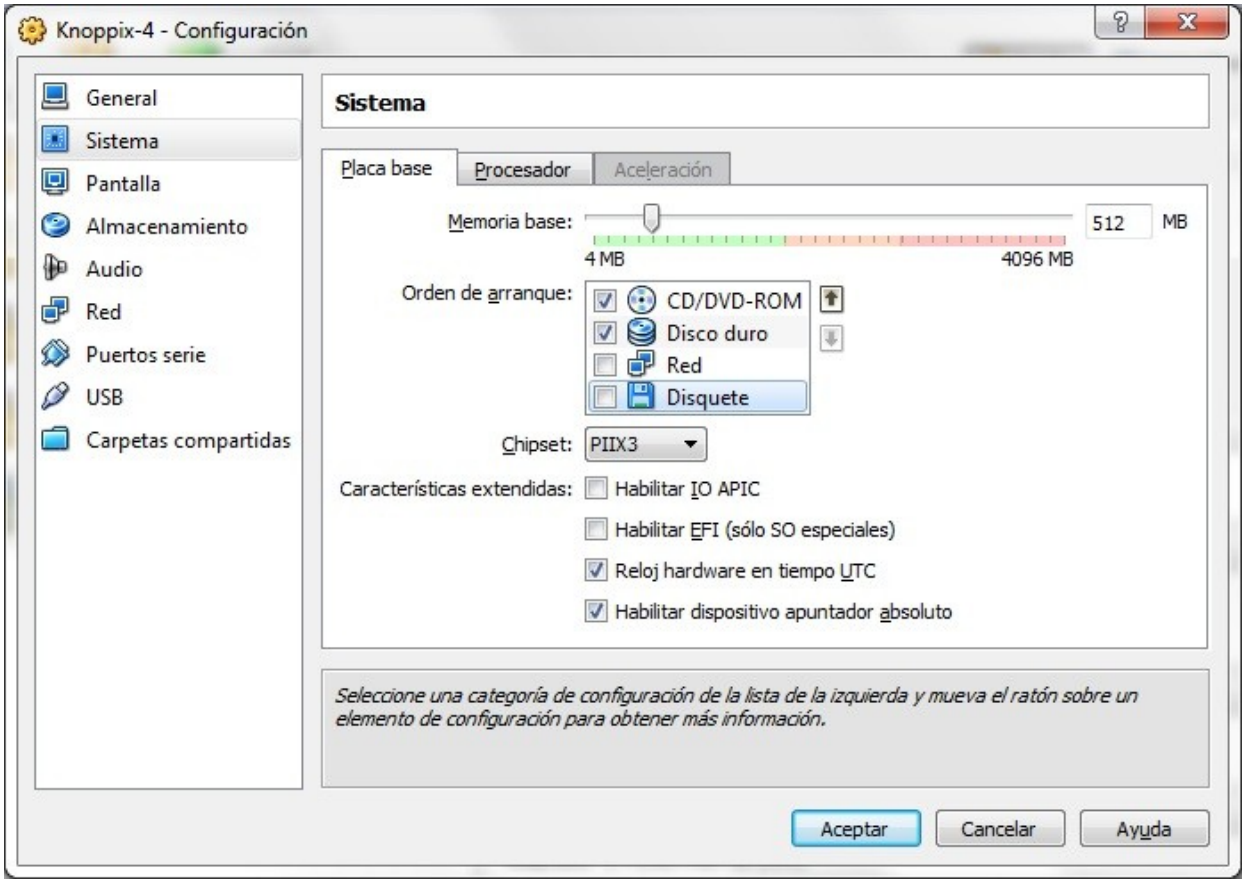
3. Luego nos pregunta sobre el nombre que le queremos dar al disco virtual y el tamaño del disco duro:



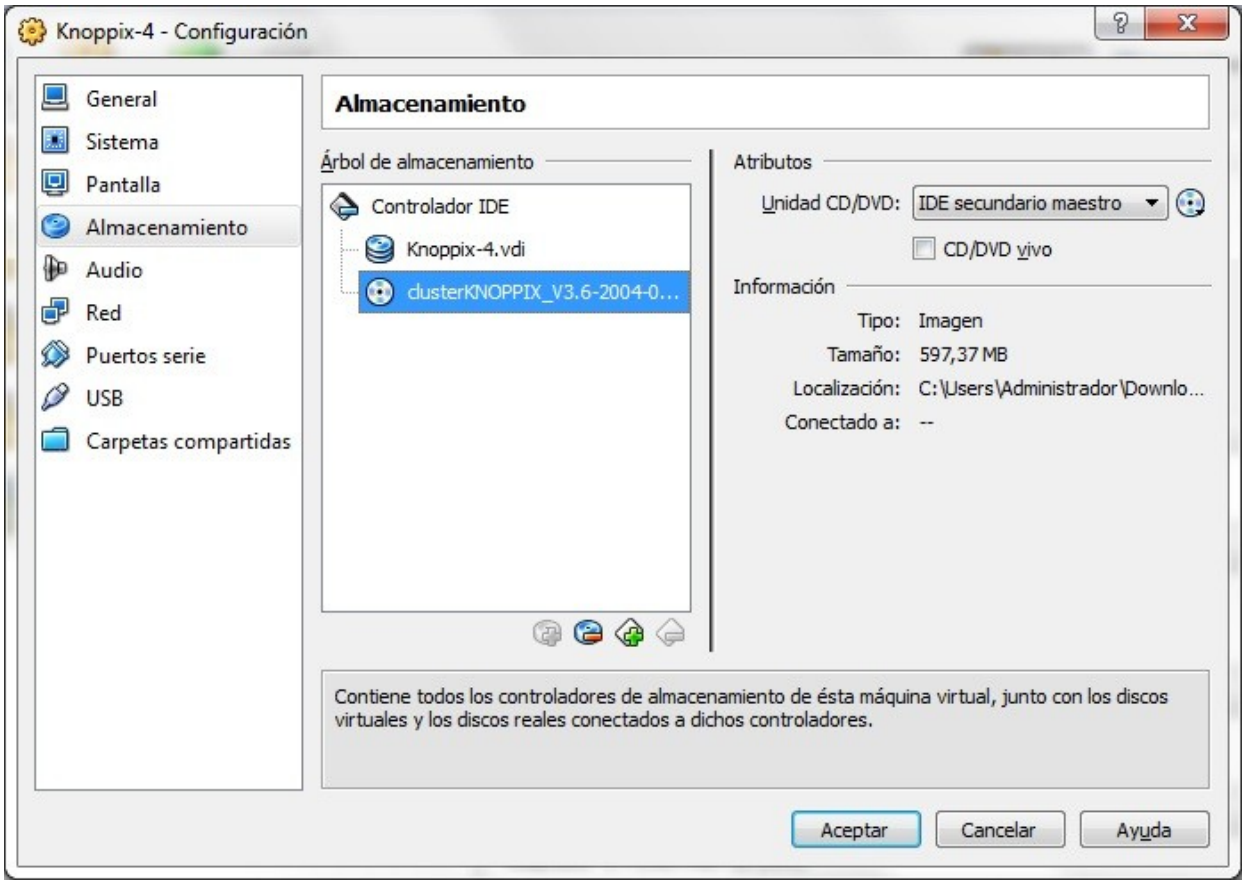
4. Finalmente le damos a crear la máquina con las características anteriormente indicadas:



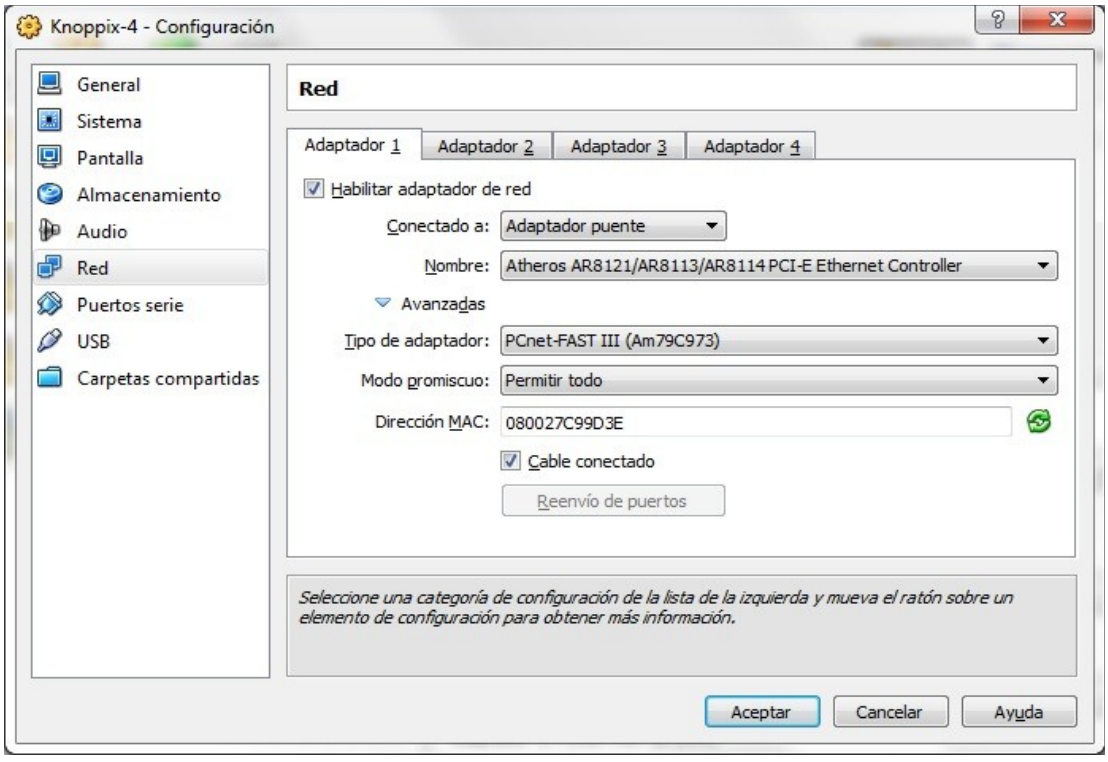
5. Una vez creada la máquina virtual hay que configurar el tipo de arranque, la red y la iso que utilizará. Para ello, en el panel de administración de VirtualBox, señalamos la máquina knoppix-4 y pulsamos sobre configuración. Ahora nos vamos a sistema y elegimos como primer arranque al CD y como segundo al disco duro:



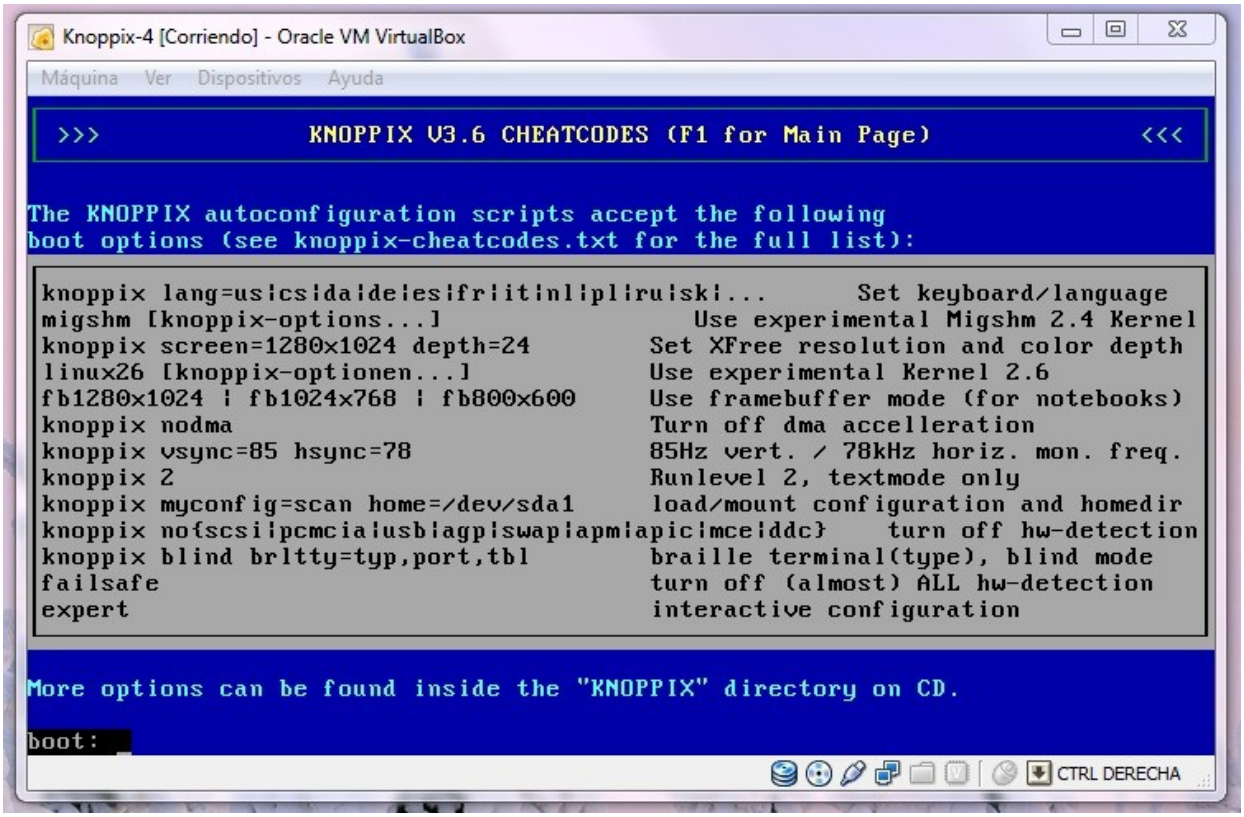
6. Posteriormente nos colocamos en almacenamiento y añadimos a la unidad de cd la iso de cluster-knoppix:



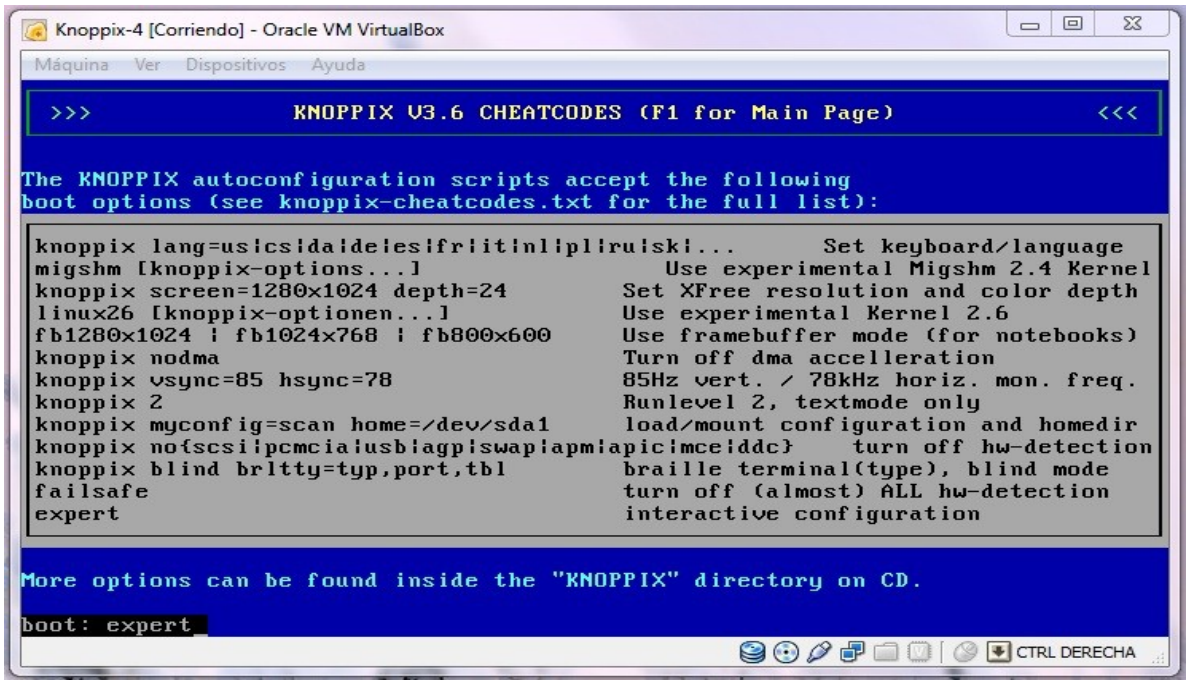
7. Por último, hay que configurar la interfaz de red para que utilice el modo puente. Este modo hará que la tarjeta de la máquina virtual funcione como un nodo de red conectado a nuestra red local. Con esto no tendremos problemas de red y permitirá que todas las máquinas de nuestro clúster formen parte de la misma red LAN:



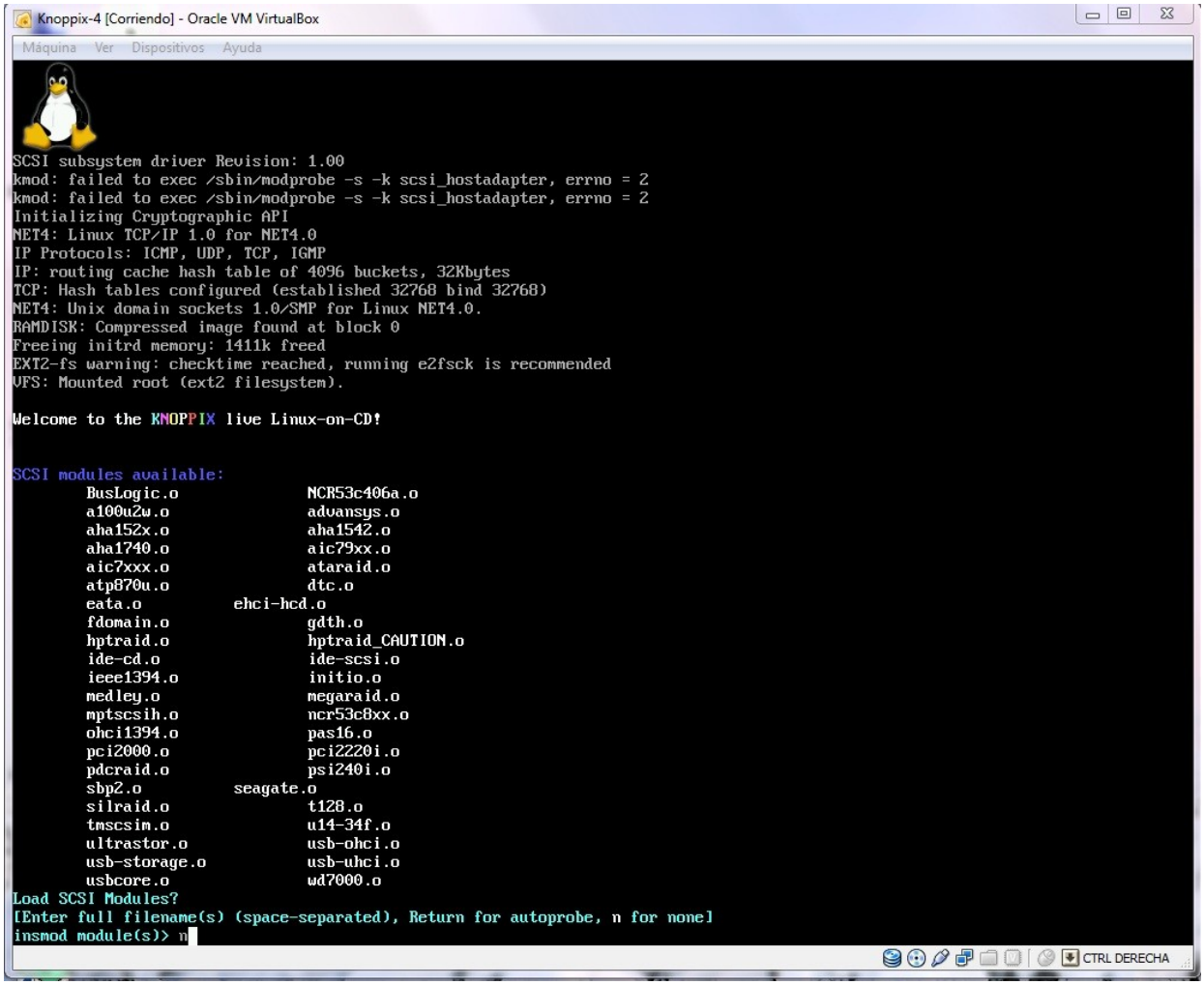
8. Ya tenemos la máquina virtual configurada de forma correcta, ahora pasamos a iniciar el cd e instalarlo en el disco duro. Cuando iniciemos, pulsamos F3 y nos saldrán los métodos de inicio distintos para ejecutar clúster-knoppix:



9. En nuestro caso lo haremos en modo "expert". Este modo permite la configuración completa de la máquina como el lenguaje del teclado, la configuración de la tarjeta de sonido, el ratón así como los distintos módulos que ha de ejecutar la máquina:



10. Una vez iniciado el modo "expert" nos pregunta si queremos cargar los módulos para los dispositivos SCSI. Como nosotros no tenemos ninguno de estos dispositivos no nos hará falta. En caso de la instalación se vaya ha hacer en un servidor dedicado sí sería necesario cargar estos módulos para dichos dispositivos:

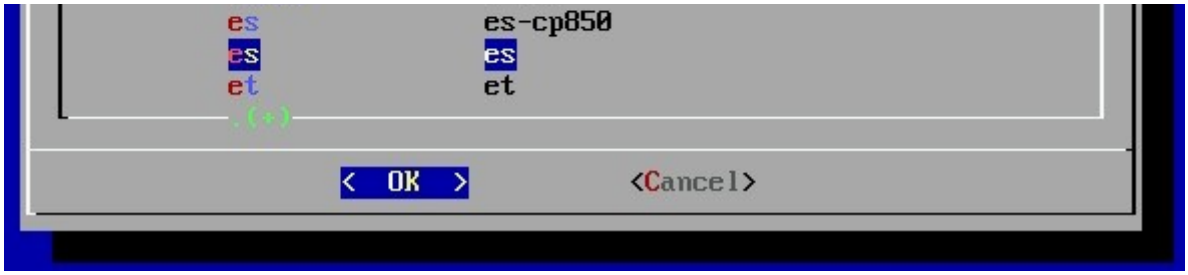


11. Luego pasa a escanear los dispositivos USB y Firewire conectados, así como las unidades de disquetes. Como tampoco tenemos le diremos que no cargue los módulos:

12. Nos preguntará si queremos reconfigurar el dioma del teclado que por defecto es "us" (teclado americano). En este caso lo vamos ha hacer:

```
Enabling DMA acceleration for: hda      [VBOX HARDDISK]
Enabling DMA acceleration for: hdc      [VBOX CD-ROM]
 Accessing KNOPPIX CDROM at /dev/scd0...
Total memory found: 513740 kB
Creating /ramdisk (dynamic size=406172k) on shared memory...Done.
Creating directories and symlinks on ramdisk...Done.
Starting init process.
INIT: version 2.78-knoppix booting
 Running Linux Kernel 2.4.27-om-20040808.
 Processor 0 is  Intel(R) Core(TM)2 Quad  CPU   Q8200  @ 2.33GHz 2293MHz, 64 KB Cache
 ACPI Bios found, activating modules: ac battery button fan processor thermal
 USB found, managed by hotplug: (Re-)scanning USB devices... sync:[001  002  ] Done.
Autoconfiguring devices...  Done.
 Mouse is Generic PS/2 Wheel Mouse at /dev/psaux
 Soundcard: Intel Corporation|82801AA AC97 Audio driver=i810_audio
Entering interactive configuration second stage.
 Your console keyboard defaults to: us
Do you want to (re)configure your console keyboard? [Y/n] Y
```

13. Elegimos la configuración de teclado española y aceptamos:



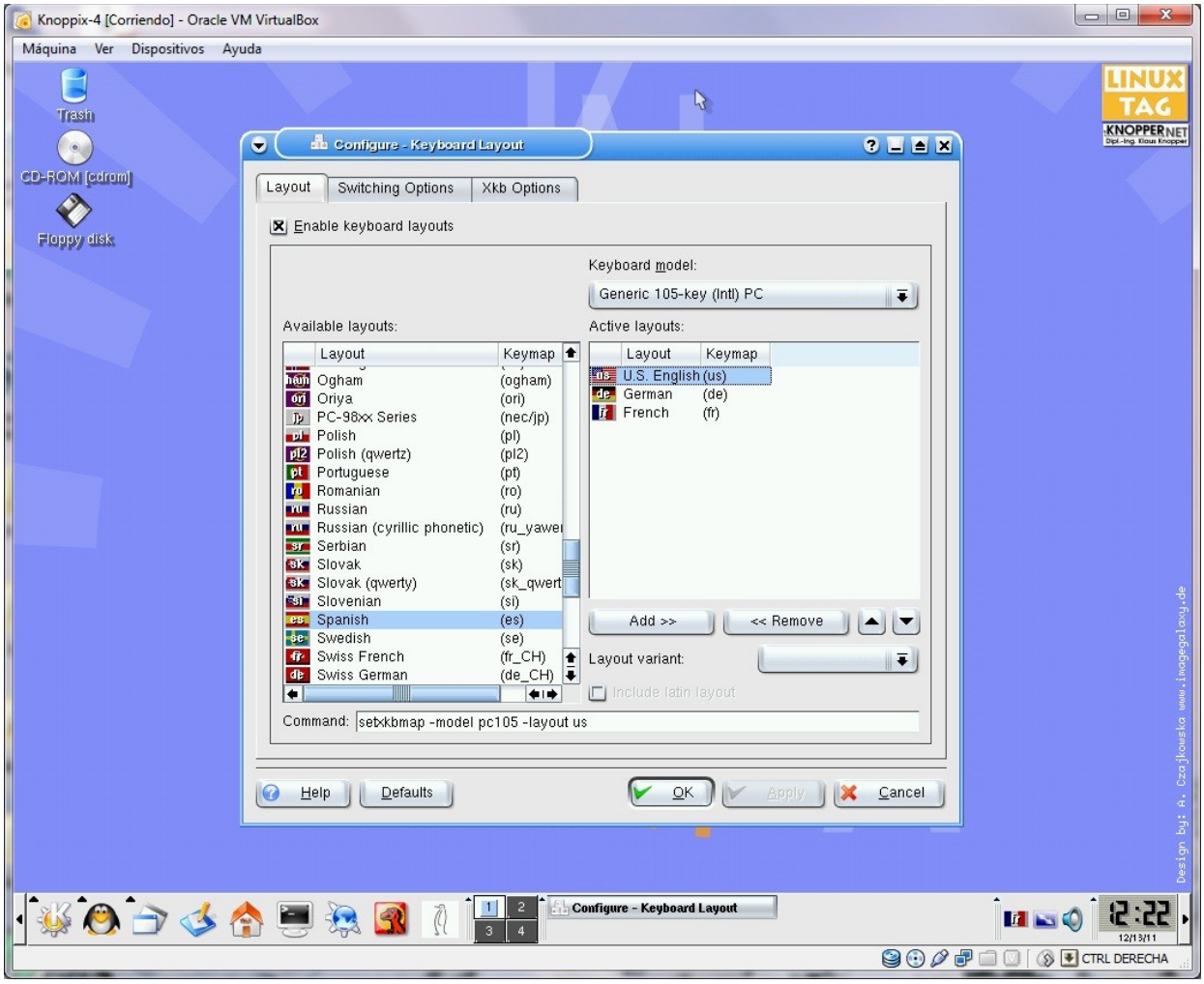
14. Ahora configura la tarjeta de sonido, si no hay problemas de compatibilidad con el hardware debería de configurarla por defecto. De no ser así la podríamos configurar a mano, aunque en nuestro caso ya está preconfigurada por lo que decimos que no y seguimos:

```
Loading /usr/share/keymaps/i386/qwerty/es.kmap.gz
Do you want to (re)configure your soundcard? [Y/n] n
```

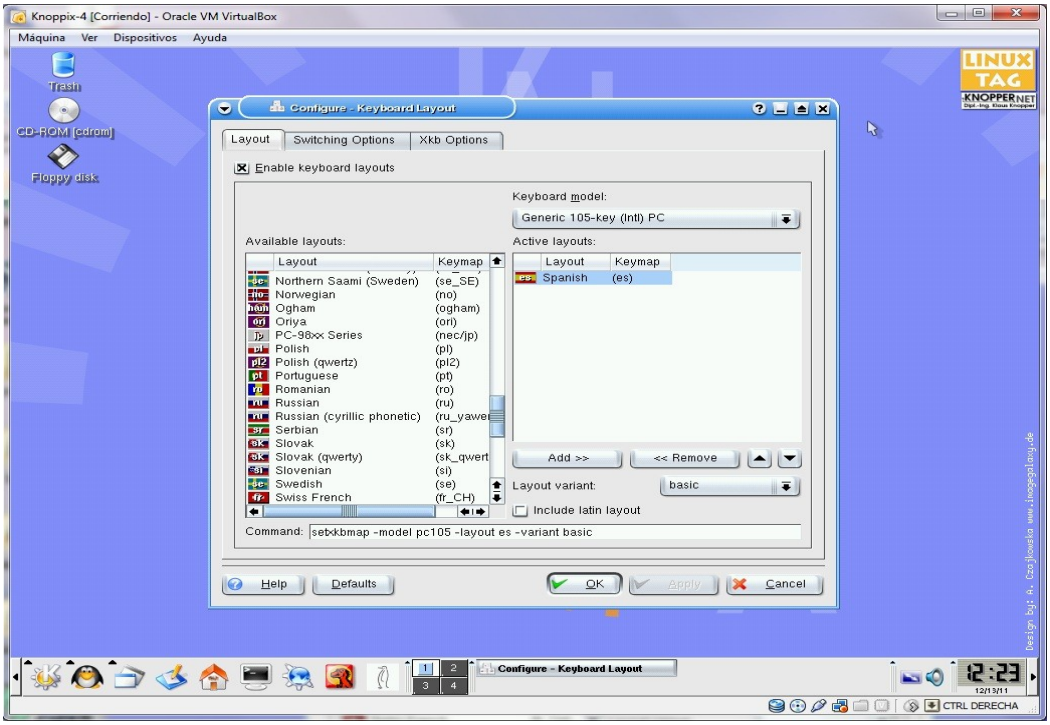
15. La configuración del ratón también la obtiene por defecto por lo que tampoco habría que configurarlo. En cuanto a esto, virtualbox tiene un sistema de integración del ratón automática, por ciertos problemas esta ha de estar deshabilitada para que podamos usar el ratón de forma adecuada, de no ser así por mucho que lo intentemos cluster-knoppix no detectará el ratón aun a pesar de que lo configuremos. Decimos que no configuraremos el ratón y avanzamos:

```
 Your mouse has been autodetected as: /dev/mouse -> /dev/psaux
Do you want to (re)configure your mouse? [Y/n] n
```

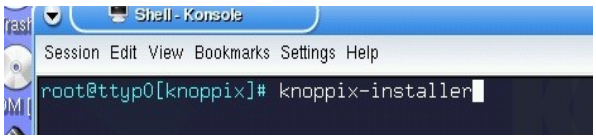
16. Una vez realizadas todas las opciones de configuración que nos pide el cd, accedemos al SO live. Lo primero que deberíamos de hacer es configurar el sistema para que siempre utilie el español como lenguaje para el teclado, puesto que cada vez que iniciemos sesión se podrá el idioma por defecto "us" (teclado americano). Para hacer esto hemos de clikear en la bandera que tenemos en la barra, abajo a la derecha; y elegimos que el idioma por defecto solo sea el español:



En la lista de la derecha aparcen los idiomas que tenemos cargados, nosotros vamos a borrarlos y cargar solo el paquete español:



17. Vamos a instalar ahora nuestro sistema, para ello podemos usar el comando knoppix-instaler que nos ejecutará el instalador del sistema. Hay que ejecutarlo con privilegios de root:

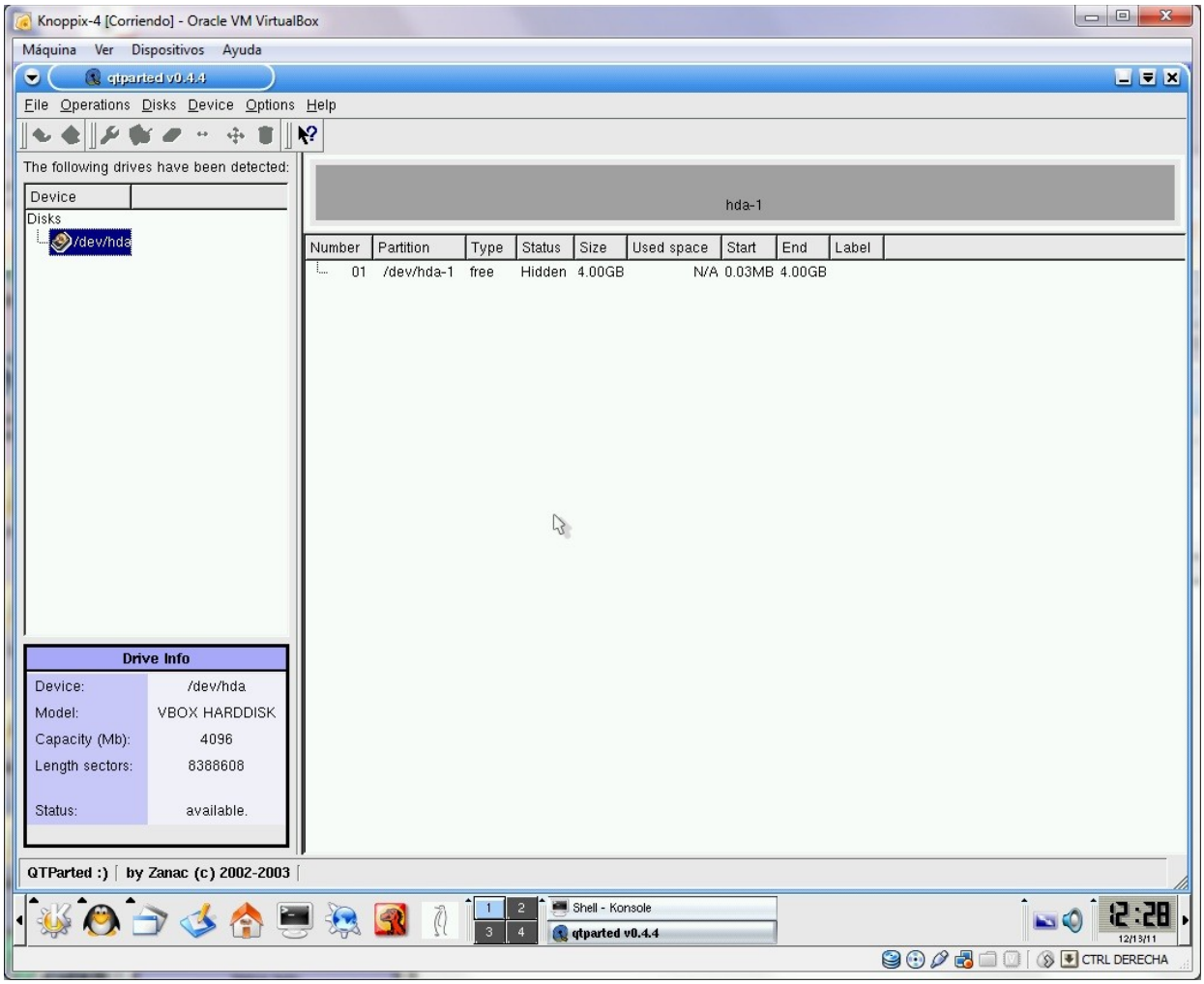


18. Una vez iniciado el instalador nos aparece un recuadro que nos indica que el disco aun no está particionado y nos sugiere cual es el espacio recomendado para ello. Pulsamos ok:

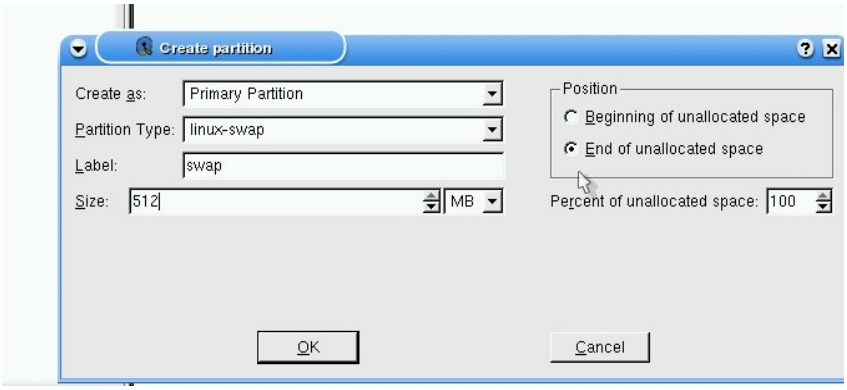
19. Nos pregunta si queremos particionar el disco, pulsamos ok para hacerlo:

20. Al darle a particionar nos aparce el programa qtparted, el cual es una herramienta para la gestión de las particiones de disco que incorpora el liveCD. Nos vamos al disco, damos click derecho sobre él y pulsamos sobre "crear nueva tabla de particiones". Nos pregunta si estamos seguro, pulsamos que sí:

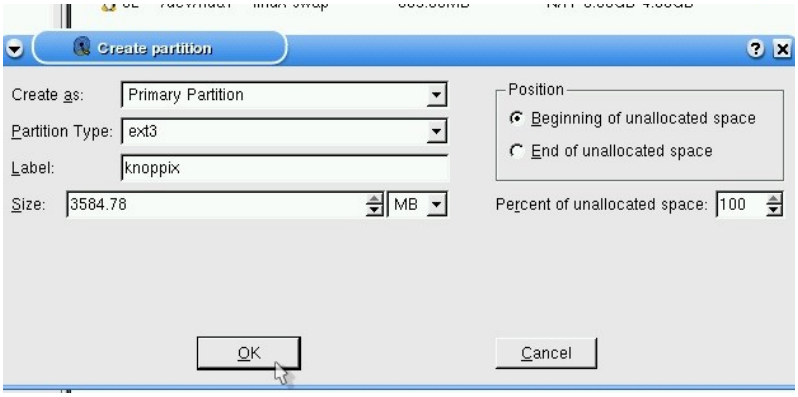
Ya tenemos la tabla de particiones creada:



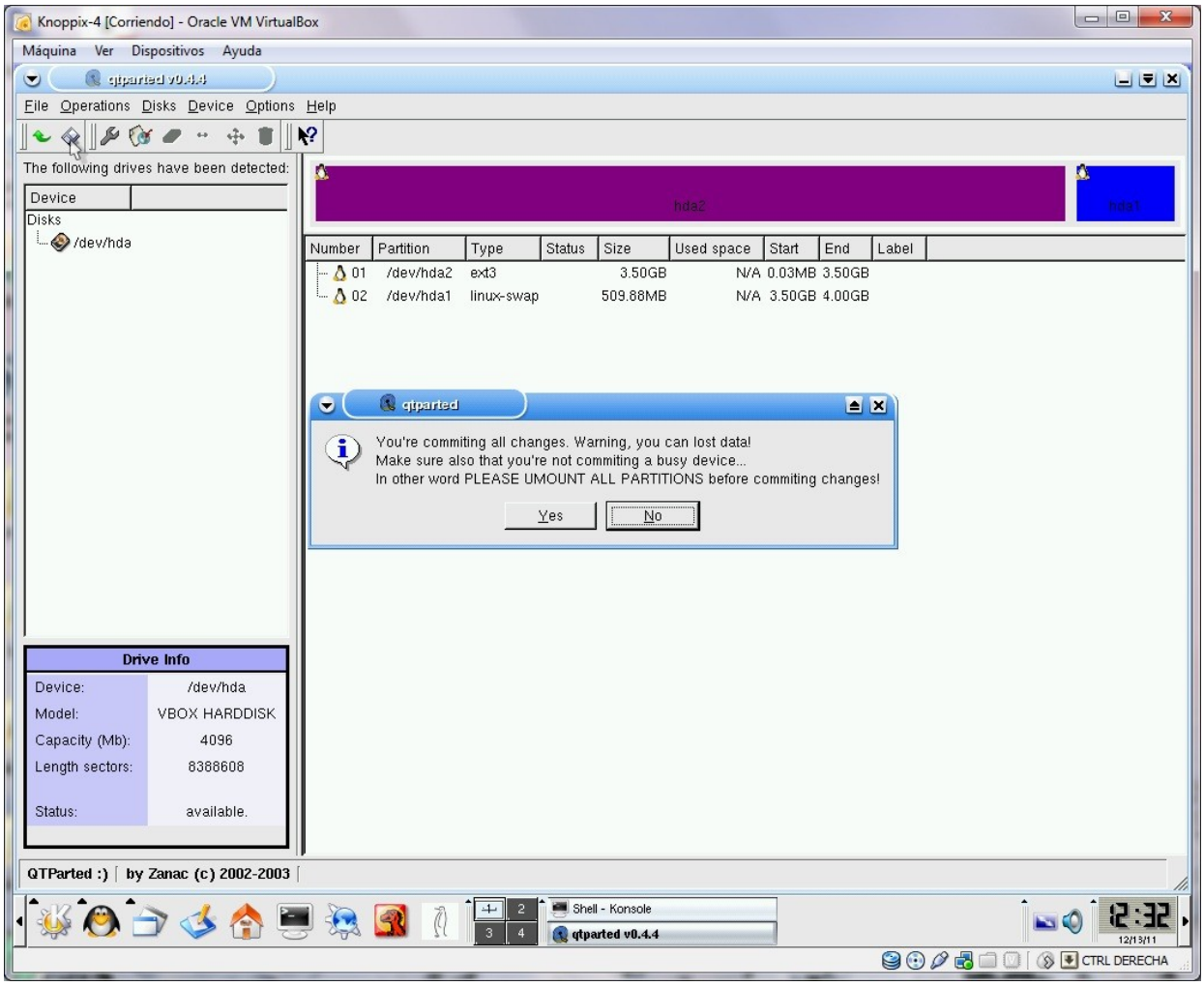
21. Una vez creada la tabla de particiones pulsamos sobre crear particiones. Hay que crear la partición que tendrá el sistema y la de swap. Como ya sabemos la partición de swap ha de ser el doble de la memoria RAM aunque se recomienda que este valor no supere los 512mb pues sería espacio desperdiciado. Creamos la swap primero, le ponemos una etiqueta y el tamaño:

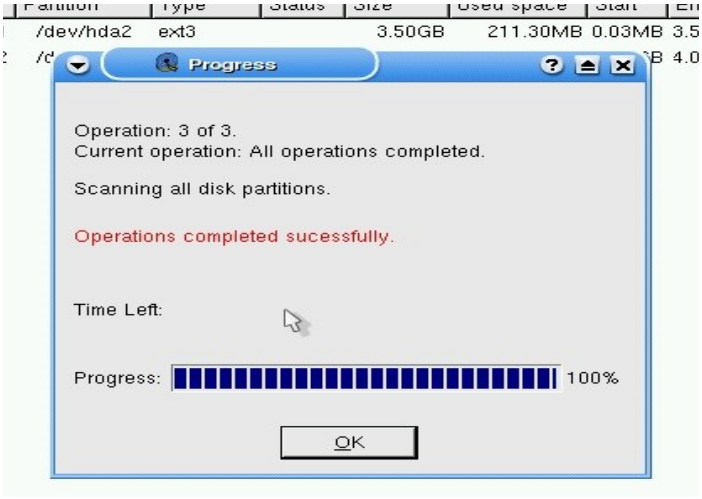


22. Ahora creamos la partición que soportará todo el sistema. La configuramos como tipo ext3, de nombre knoppix y con el resto del espacio libre en el disco duro:



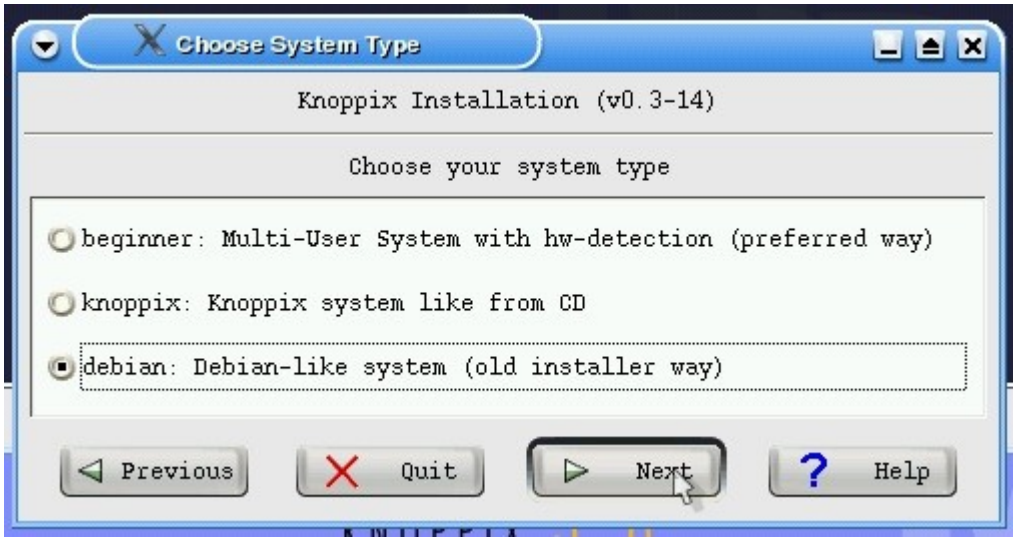
23. En la ventana que aparece en la parte superior izquierda hay un disquete, pulsamos sobre él y le decimos que sí para que inicie el proceso de formateo de las particiones creadas. Una vez finalizado cerramos la ventana de qtparted:



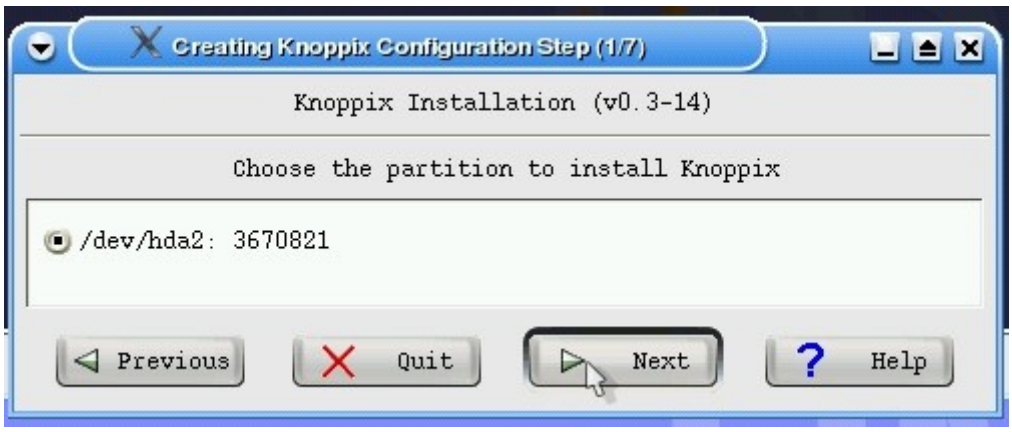


24. Cuando hallamos cerrado la ventana, nos aparece un nuevo recuadro con disitintas opciones, la primera es la configuración de la instalación, cliekeamos sobre ella:

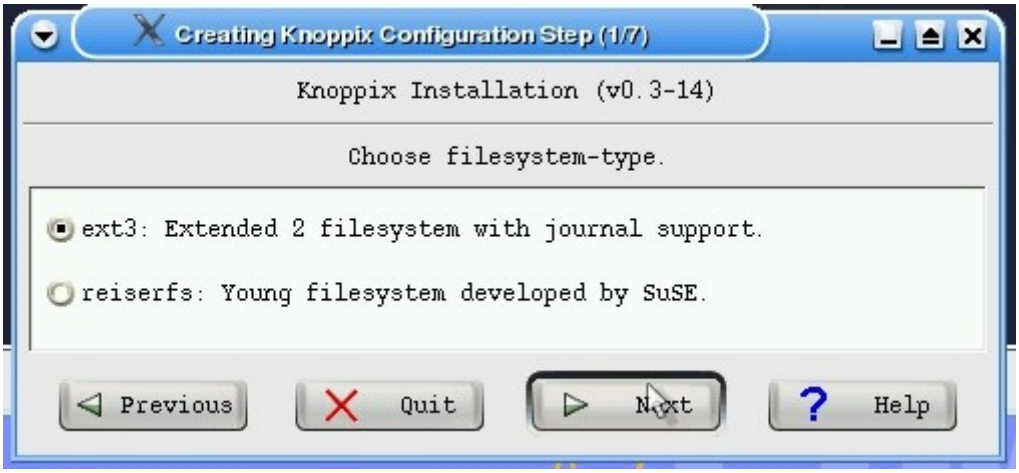
25. Ahora seleccionamos la última opción: debian: sistemas como debian (método del antiguo instalador):



26. Seleccionamos el disco y continuamos:



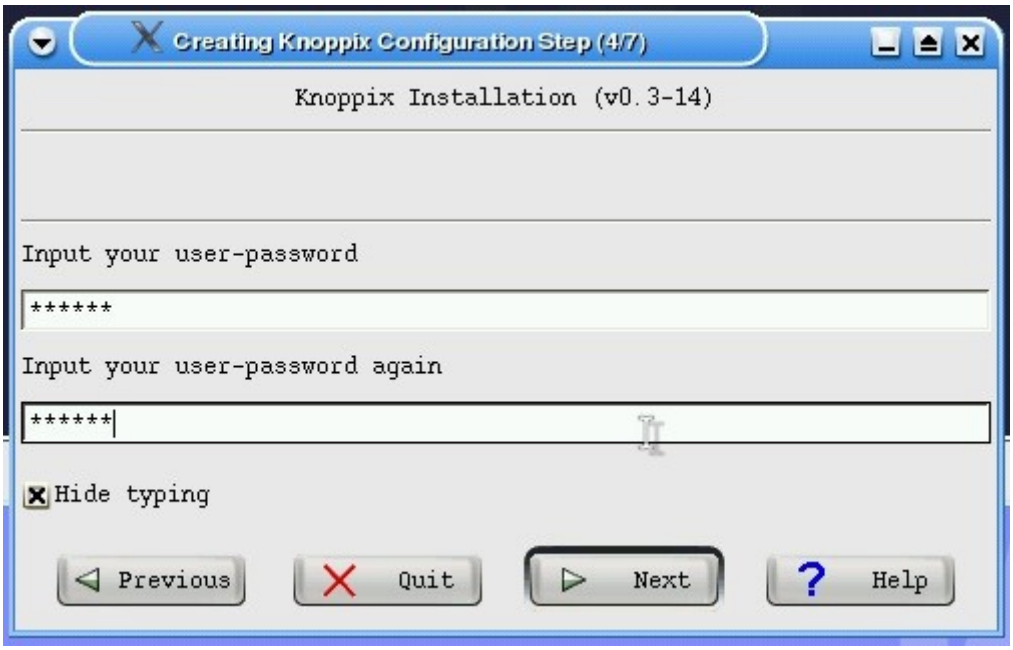
27. Nos pregunta el sistema de ficheros a usar, en nuestro caso ext3:



28. Ingresamos el nombre completo del usuario que va a utilizar la máquina (en nuestro caso será knoppix):

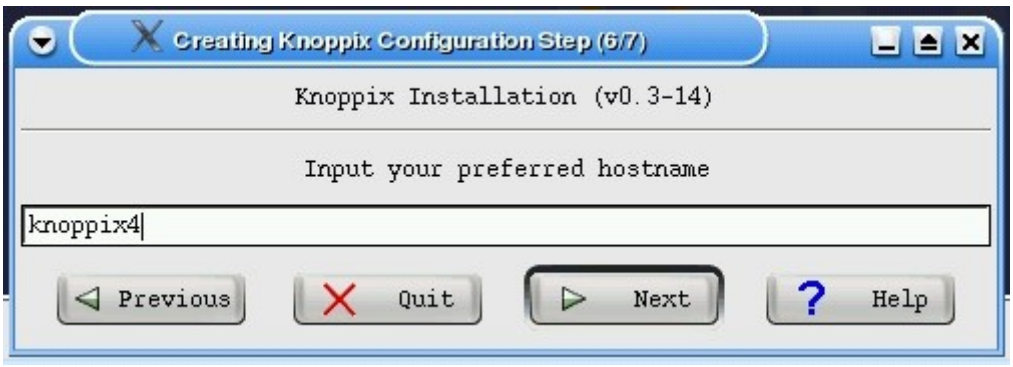
29. También ingresamos el nombre de usuario de la máquina (también es knoppix):

30. La clave del usuario:

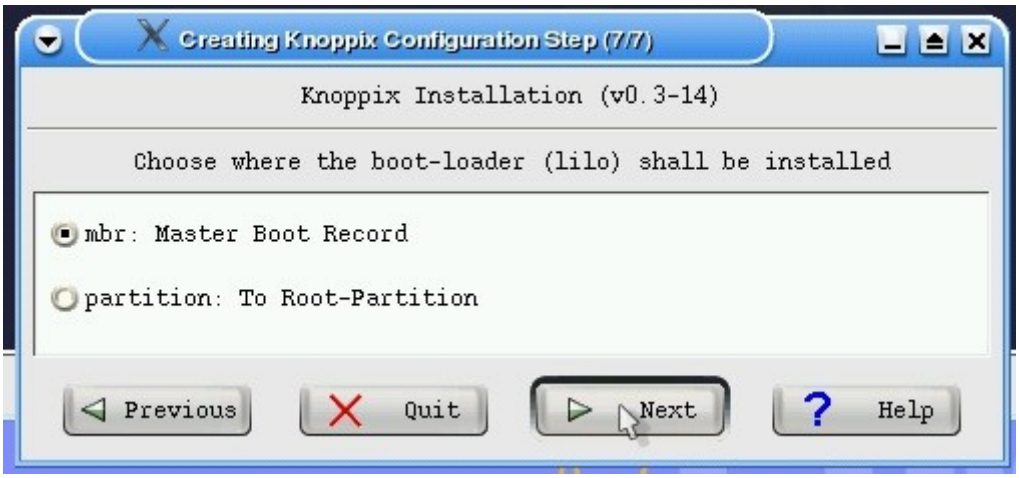


31. La clave de root:

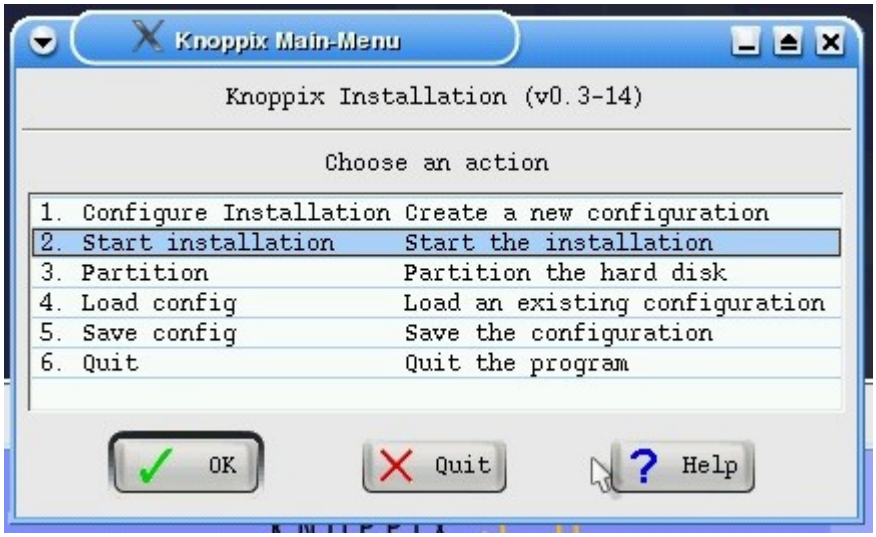
32. Ahora nos pide el nombre del host, le vamos a decir que se llame knoppix4:



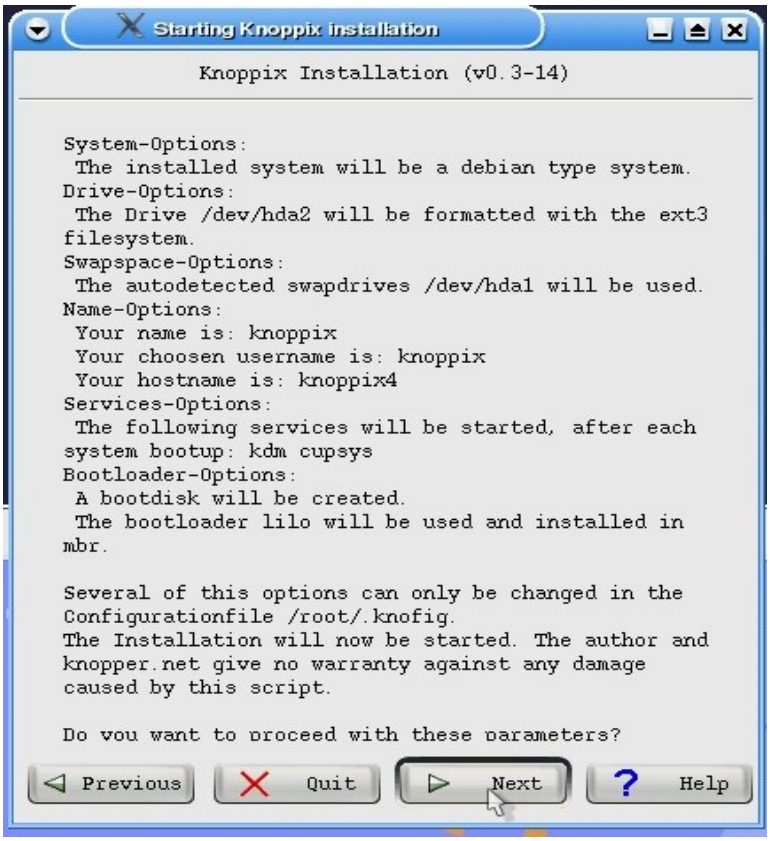
33. Pasamos a indicar el gestor de arranque que será "mbr: sector de arranque primario":



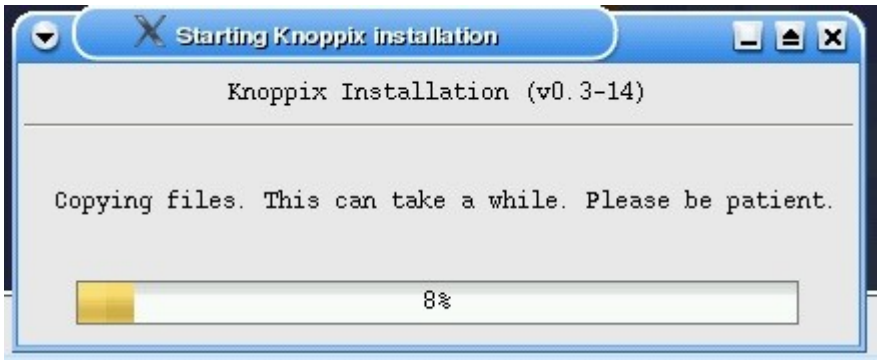
34. Ahora nos devuelve de nuevo al menú de instalación, como ya está configurada la instalación pasamos a comenzarla:



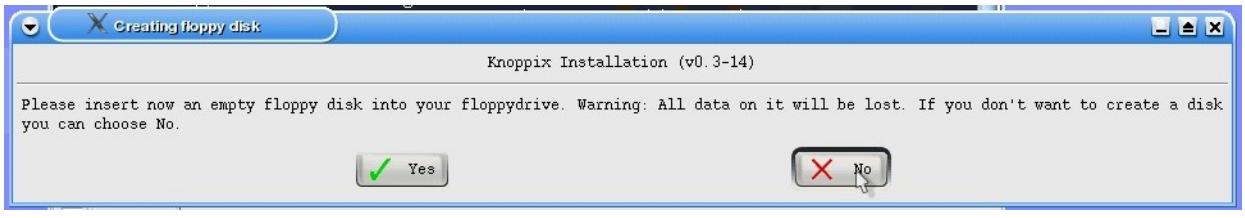
35. Nos muestra una ventana con la configuración que le dimos previamente, si está todo correcto continuamos, de no ser así siempre podemos volver a configurar la instalación:



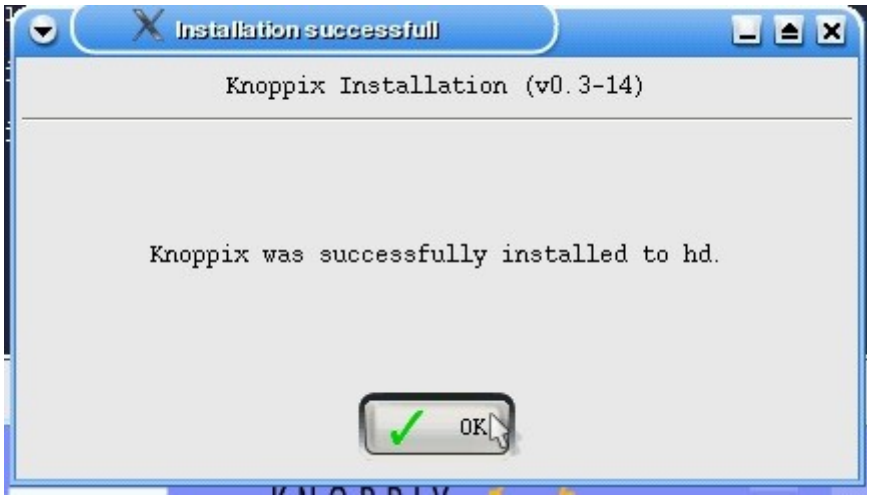
36. Ya está inciado el proceso de instalación:



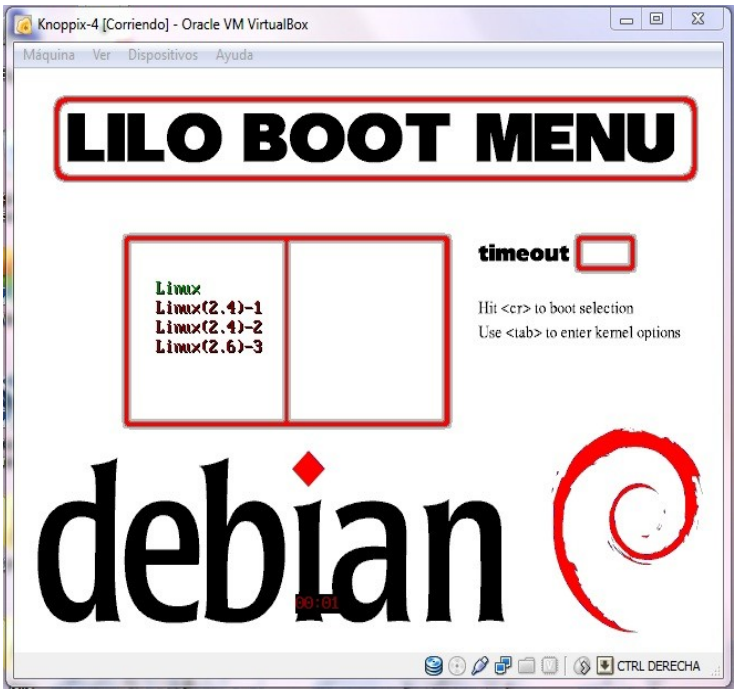
37. Por último nos pide que introduzcamos un disquete, pero le decimos que no y seguimos:



38. La instación se llevó a cabo de forma satisfactoria:



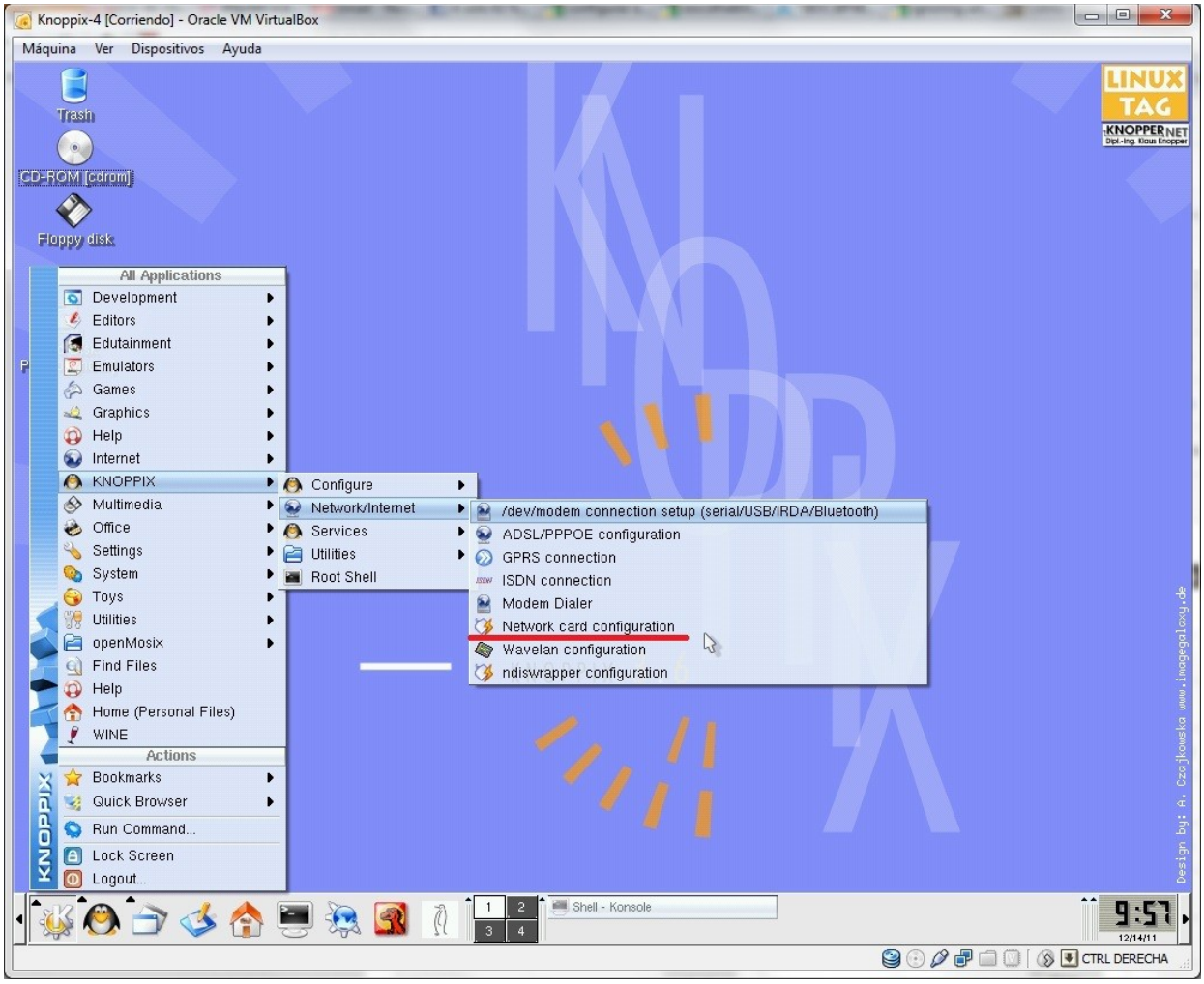
39. Reiniciamos el equipo y desmontamos la iso para que inicie desde el propio disco duro. De ser así nos saldrá esta pantalla de inicio:



Configuración de red

Una vez iniciado el sistema de clúster-knoppix, lo primero que deberiamos hacer es la configuración de la interfaz de red. La podríamos hacer de forma manual pero cluster-knoppix ya trae herramientas para realizarlo de forma automática.

1. Para ello nos serviremos de una aplicación dentro del sistema para la configuración de la red (utiliza un comando llamando **netcardcofig**):

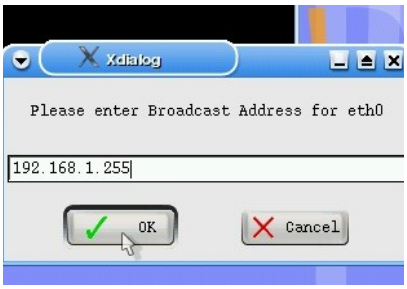


2. Lo primero que nos pide es si queremos configurar la tarjeta mediante dhcp, como nos interesa que sea de forma manual para poner las ip estáticas, le decimos que no:

3. Le indicamos que ip va a tener, en nuestro caso será 192.168.1.54:

4. La mácara será /24 como el resto de equipos de la red:

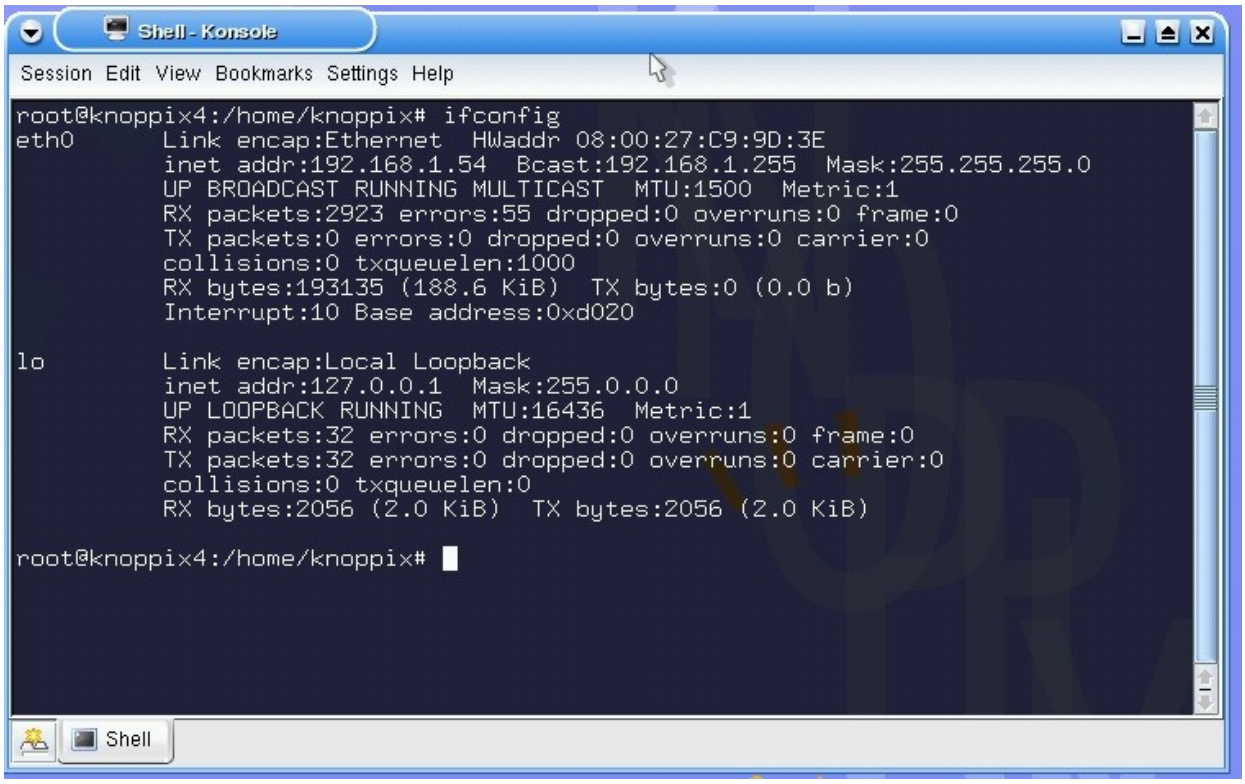
5. También ponemos la dirección de broadcast:



6. La puerta de enlace será nuestro router de la red que da salida a Internet, este tiene ip 192.168.1.1:

7. El servidor DNS también es el router:

8. Con el comando **ifconfig** podemos comprobar que la configuración se ha hecho de forma correcta:



Con esto tendremos configurada la red de la máquina para que funcione el clúster de openmosix.

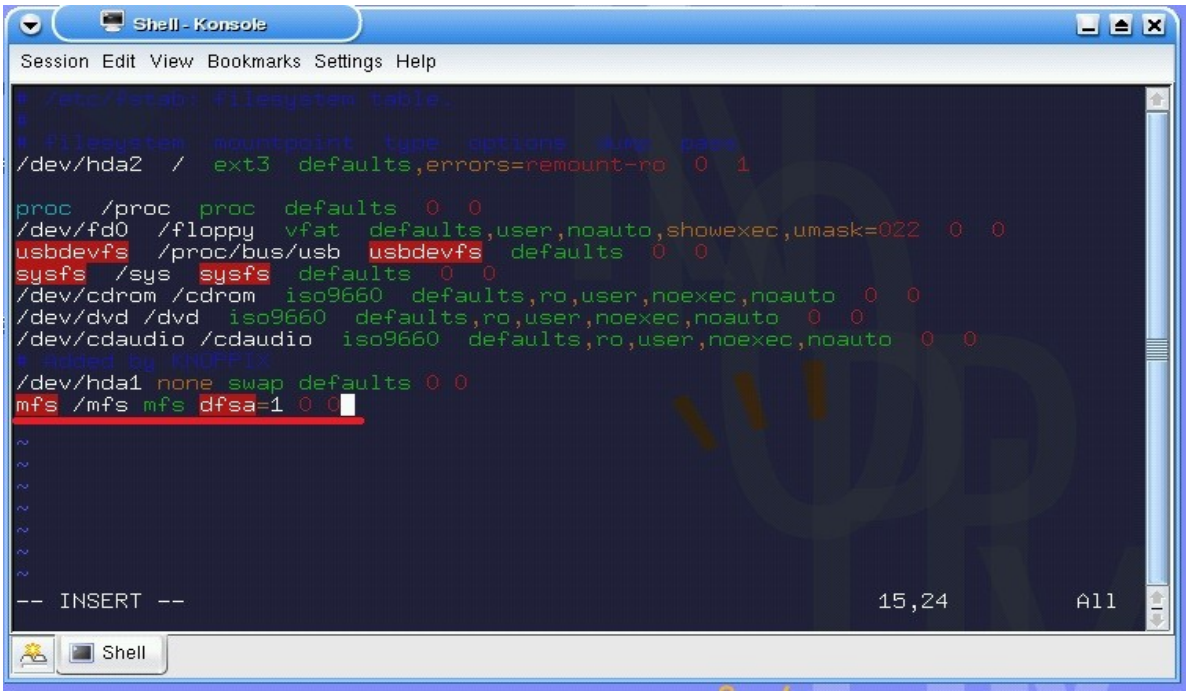
Montar el sistema de ficheros MFS

Como dijimos antes, openmosix funciona con un sistema de fichero llamado MFS, este permite que se puedan repartir los distintos procesos entre los nodos del clúster para aumentar el rendimiento. Sin este sistema de ficheros nuestro clúster no funcionaría por lo que es algo imprescindible. Pero por defecto no está montado, por lo que hay que realizar dos tareas: 1º editar el fichero /etc/fstab para que monte el mfs de forma automática y 2º montarlo en un directorio que nosotros crearemos. Veamos como hacerlo.

1. Configuramos el fichero **/etc/fstab** con el editor de texto vim que ya viene instalado con el siguiente comando:

2. Para montar el sitema de fichero hay que agregar la siguiente línea:

   mfs /mfs mfs dfsa=1 0 0:



Para los que no estamos acostumbrados al uso del editor de texto **vi**, podríamos entrar en caos al intentar editar algún fichero con vim. Para ello podemos usar estos pasos sencillos:

1º Pulsamos la letra "i" para insertar cambios.

2º Una vez hayamos hecho los cambios pertinentes pulsamos "Esc" para salir del modo de edición.

3º Para guardar los cambios pulsamos ":x" se guardan y salimos.

3. Volviendo al montaje de mfs, una vez que hayamos editado el fichero fstab hay que crear el directorio donde va a ser montado el sistema de ficheros, el cual será **/mfs** con el siguiente comando:

```
root@knoppix4:/# mkdir /mfs
root@knoppix4:/#
```

4. Ahora solo queda montarlo con el comando **mount**:

```
root@knoppix4:/# mount mfs
root@knoppix4:/#
```

5. Para asegurarnos de que esté funcionando podemos probar con reiniciar el demonio de openmosix y veremos si se está ejecutando el sistema mfs:

```
root@knoppix4:/# /etc/init.d/openmosix restart
openMosix: Using map file /etc/openmosix.map
openMosix: WARN: Invalid configuration in map-file /etc/openmosix.map
openMosix: Falling back to autodiscovery mode using /usr/sbin/omdiscd
Stopping openMosix...
openMosix: All remote processes were expelled and no further remote processes a
ccepted.
Initializing openMosix...
Local processes already allowed to leave automatically.
automatic load-balancing already enabled.
Remote processes now allowed in.
MFS access already enabled.
root@knoppix4:/#
```

Con esto tendremos el sistema de ficheros prepardo para usarlo en el clúster y configurado para que se ejecute en cada inicio de sesión de la máquina.

Configuración de SSH

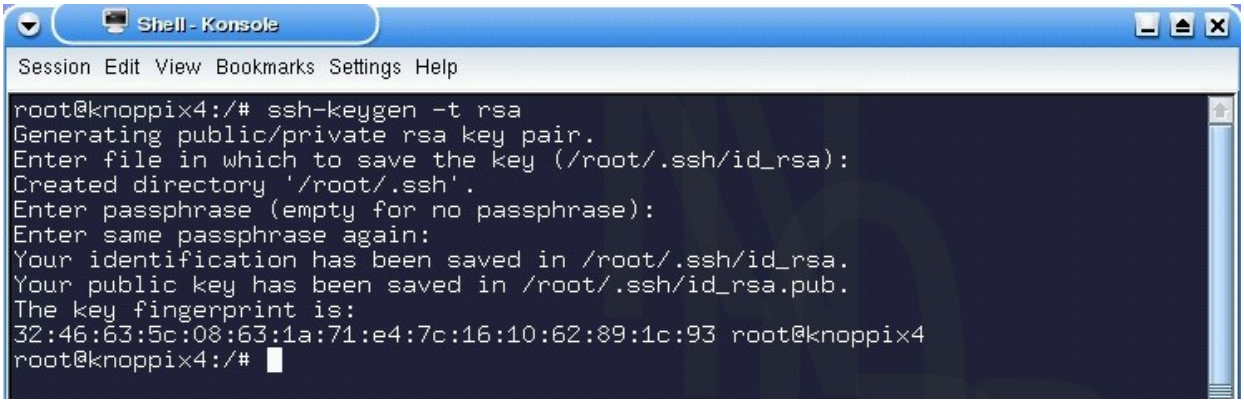
Openmosix comunica todos y cada uno de sus nodos mediante conexiones de SSH. A través de este canal pasa los mensajes de los procesos a los ditintos nodos y estos a su vez lo devuelven al nodo que actúe en ese momento de master (el nodo que ejecutó el proceso). Esto mediante el método convencional de conexiones ssh no sería del todo factible. La razón es que cada vez que se iniciara un proceso, saldría peticiones para introducir la clave del usuario del nodo al que se va a conectar por ssh y un nuevo mensaje cuando dicho nodo quiera conectarse al master. Si se quiere automatizar este proceso la única forma es utilizar claves públicas y privadas de ssh para que la comunicación sea transparente para el usuario o administrador del clúster.

Eso es precisamente lo que vamos a configurar. Crearemos la claves y le diremos al clúster que toda comunicación se haga a través de las claves públicas que estarán añadidas en el fichero **authorized_keys**. Veamos cómo hacerlo.

1. Para crear las claves usamos el siguiente comando de ssh:

```
root@knoppix4:/# ssh-keygen -t rsa
```

2. Nos pedirá dónde guardar las claves (le diremos que en el fichero /root/.ssh/id_rsa), también una frase de paso que no le diremos para que no nos pida ningún tipo de clave. Con esto ya tendremos creadas la clave pública y privada de la máquina y nos indica donde se han guardado:



3. Verificamos que ambos archivos se han creado de forma satisfactoria. La clave privada en

/root/.ssh/id_rsa y la pública en **/root/.ssh/id_rsa.pub:**

```
root@knoppix4:/# ls /root/.ssh/
id_rsa  id_rsa.pub
root@knoppix4:/#
```

4. Para que la conexión transparente tenga éxito vamos a crear el fichero **authorized_keys** para autorizar la conexión ssh de ese nodo y así igualmente con las claves de los nodos a los cuales se quiere que se conecten. Para ello usamos la siguiente salida del comando cat para redirigir el contenido al fichero antes indicado:

```
root@knoppix4:/# cat /root/.ssh/id_rsa.pub > /root/.ssh/authorized_keys
root@knoppix4:/#
```

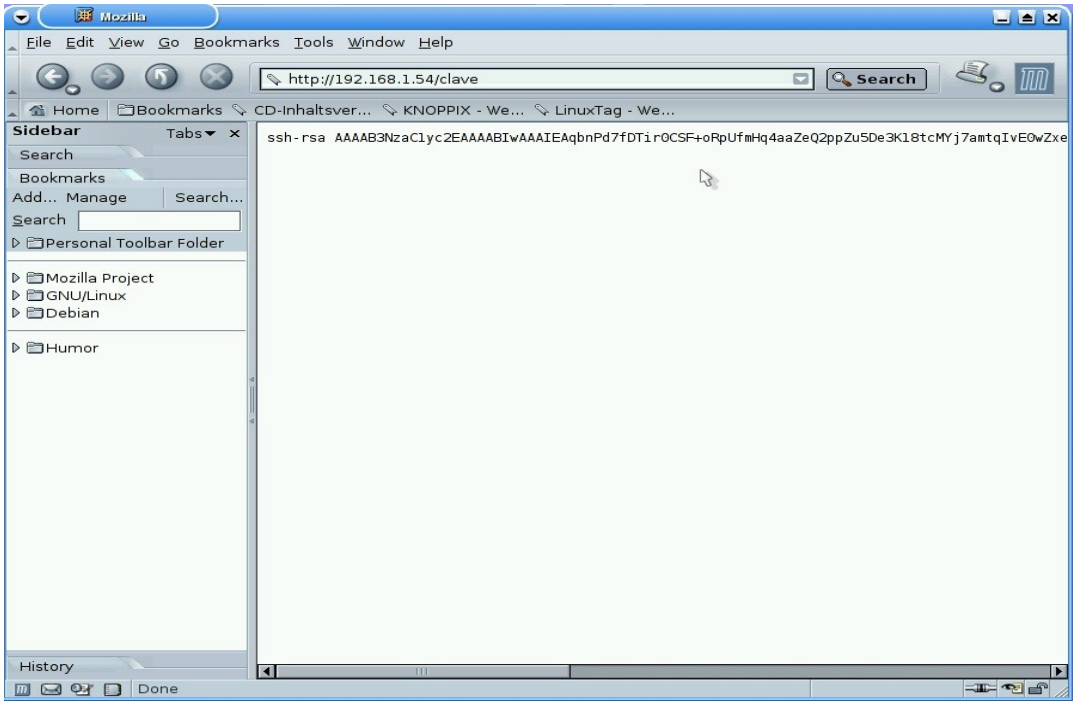
5. Una buena práctica es publicar en el servidor web que incorpora nuestro clúster-knoppix, la clave pública para que desde los otros nodos podamos añadirla al fichero de autorización de los demás con el fin de que todos se puedan conectar con todos de forma automática. Para ello copiamos el fichero **id_rsa.pub** al directorio del servidor web:

```
root@knoppix4:/# cp /root/.ssh/id_rsa.pub /var/www/clave
root@knoppix4:/#
```

6. Para que los cambios surtan efecto reiniciamos el servidor web apache:

```
root@knoppix4:/# /etc/init.d/apache restart
Restarting apache[Wed Dec 14 10:27:40 2011] [warn] module ssl_module is already
 loaded, skipping
.
root@knoppix4:/#
```

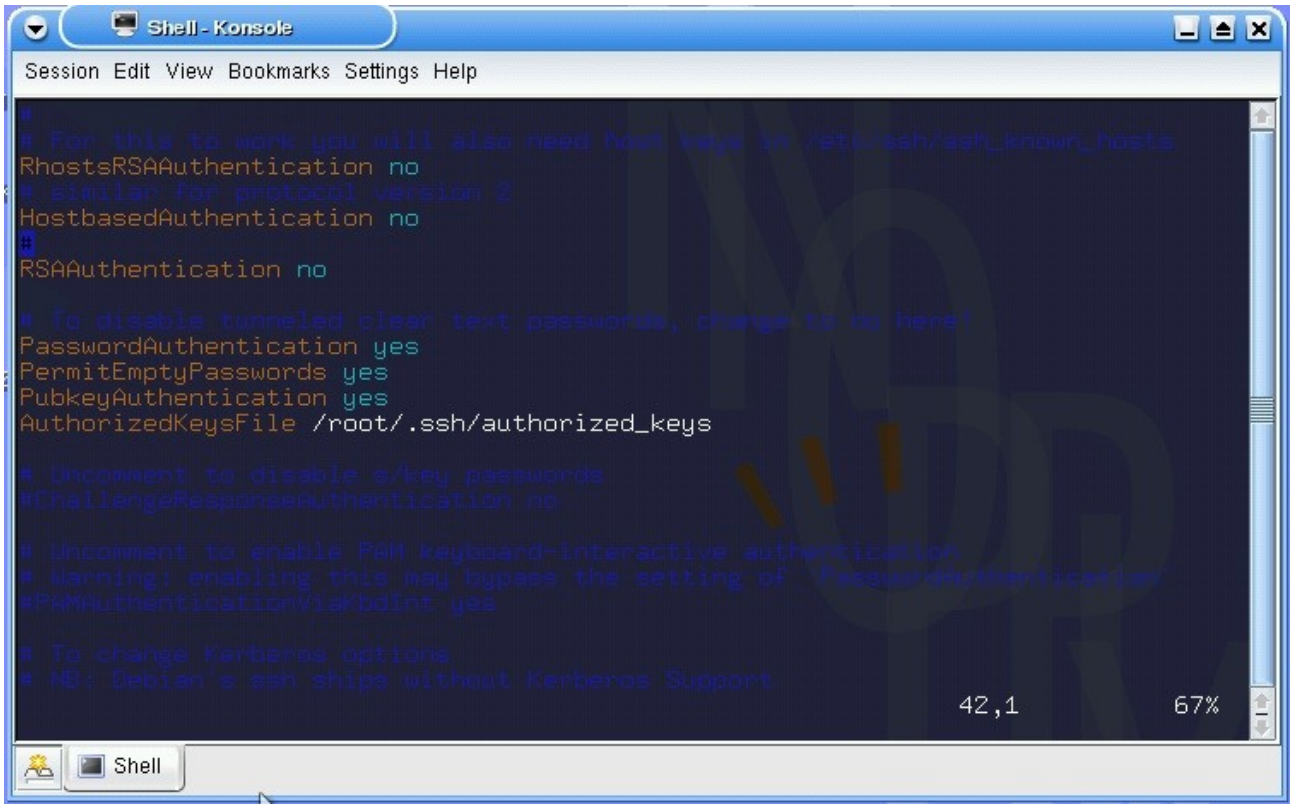
7. Verificamos desde el navegador que está públicada la clave:



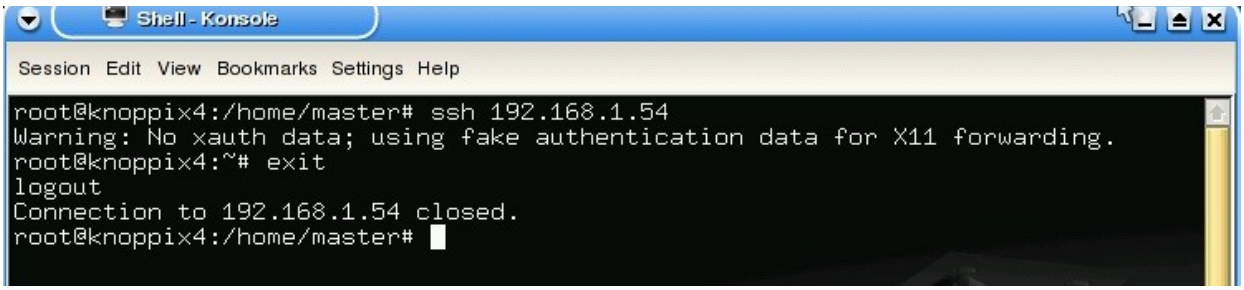
8. Ahora hay que configurar el fichero SSHD para que utilice la utenticación por claves al usar conexiones ssh con el siguiente comando:

```
root@knoppix4:/# vim /etc/ssh/sshd_config
```

9. En el fichero agregamos las siguientes líneas:
   - **RSAAuthentication no**: no realiza autenticaciones de llaves RSA.
   - **PasswordAuthentication yes**: requiere autenticación de claves.
   - **PubkeyAuthentication yes**: realiza autenticación de claves públicas.
   - **PermitEmptyPasswords yes**: permite el ingreso de claves vacías (sin caracteres).
   - **AutorizedKeysFile /root/.ssh/authorized_keys**: Ruta del archivo con las claves públicas.



10. Comprobamos que se realizan login remotos sin pedir clave:



Esta configuración de SSH permite que las conexiones del clúster siempre se hagan de forma cifrada. Evitando que intrusos se cuelen en ella, así como que nodos no autorizados accedan a los procesos y estos puedan llegar a ser alterados.

Ya tenemos la configuración ssh creada ahora ya solo nos falta ejecutar el clúster y realizar pruebas sobre él para ver si responde de forma correcta.

Monitorizando el clúster con Openmosix y sus herramientas

Con esto ya tendremos nuestro clúster configurado para que funcione. Ahora pasaremos a explicar las distintas herramientas para monitorizar el clúster y posteriormente pasaremos a verlo funcionando con distintas pruebas de ejecución de proceso; así como algunas pruebas de estrés para ver como funciona cuando se le mete carga masiva. Todas estas herramientas están incluidas en el sistema de clúster-knoppix por lo que no habrá que instalarlas posteriormente.

Pasemos a ver las distintas herramientas. Openmosix usa un total de 6 aplicaciones controladas por una principal, que son las siguientes:

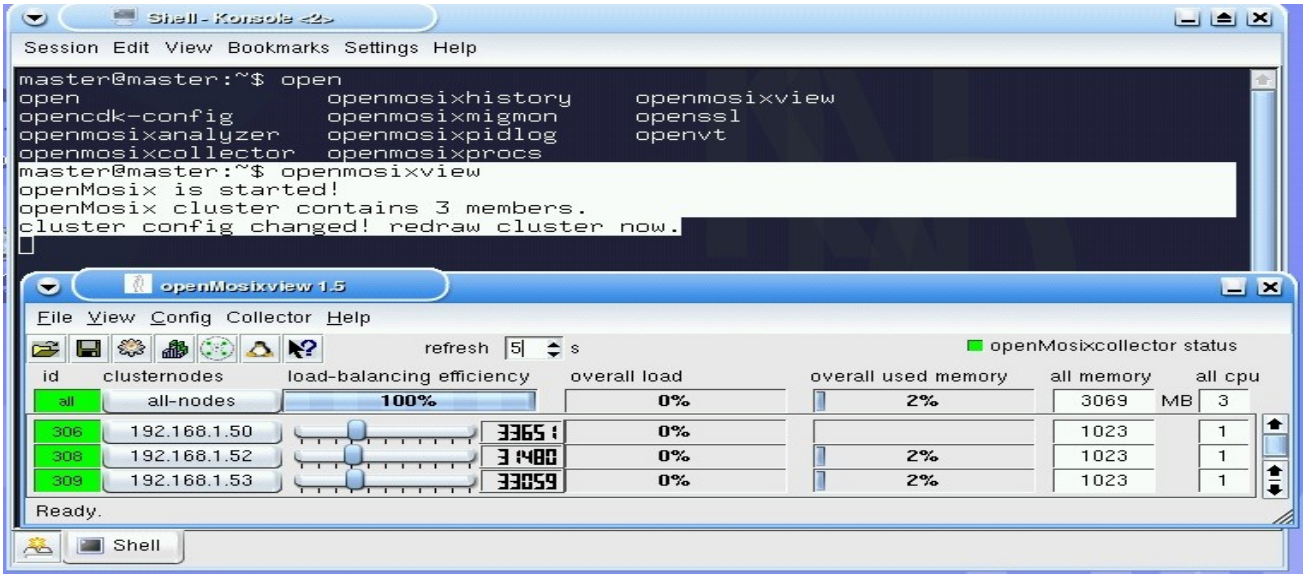
- OpenMosixView: principal herramienta de monitorizacióny administración del clúster.
- OpenMosixProcs: aplicación para la administración de procesos.
- OpenMosixAnalyzer: analizador de la información capturada por OpenMosixCollector.
- OpenMosixHistory: proporciona un historial de los procesos ya ejecutados en el clúster.
- OpenMosixMigmon: visor que representa como se lleva a cabo la migración de los procesos entre los distintos nodos.
- Mosmon: visor para la monitorización de datos.

Pasemos a verlas una a una.

OpenMosixView

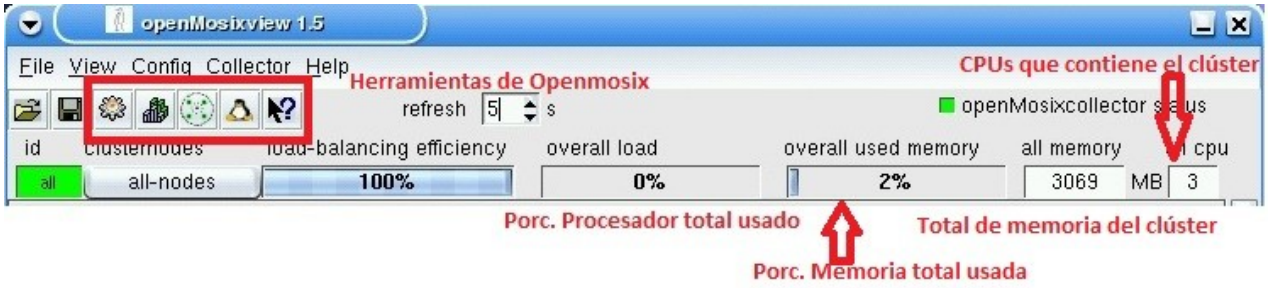
Esta es la herramienta principal de administración del clúster. Con ella podemos gestionar cuál es la memoria usada, el porcentaje de procesador de cada nodo; así como la capacidad de cómputo y memoria total que tiene nuestro clúster.

Para ejecutarla solo hay que abrir una consola y teclear el comando siguiente:

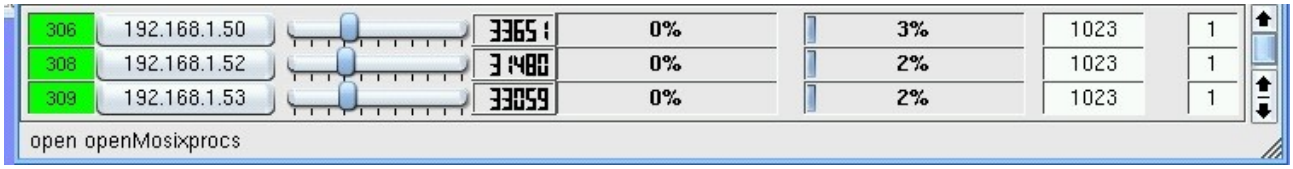


Como se observa este comando se ha ejecutado como usuario master, aunque lo ideal es que se haga como usuario root para tener los privilegios necesarios sobre las aplicaciones de la suite de OpenMosix.

Como también se observa en la captura de arriba, openmosix nos indica que el demonio omdiscd ha encontrado dos nodos, a parte de donde se está ejecutando que ha añadido al clúster.



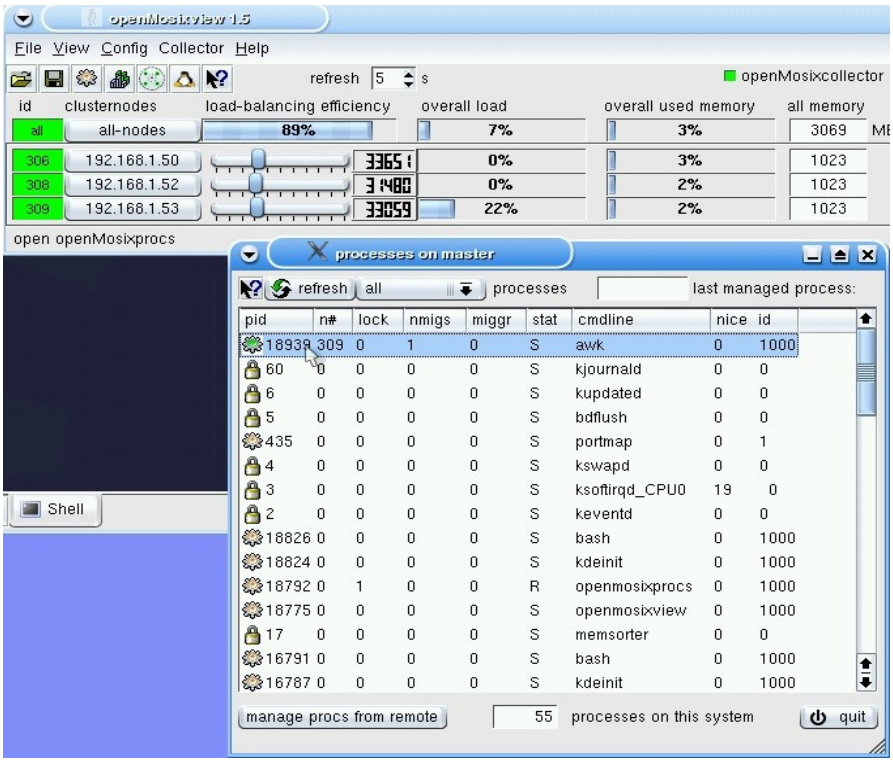
Aquí vemos las distintas partes de la pantalla de adminsitración. Como vemos nuestro clúster ya está funcionando como un solo equipo con 3 procesadores y 3GB de RAM.



En esta captura se ve las 3 máquinas que ahora mismo forman parte del clúster. Lo primero es el número de nodo que OpenMosix le asigna (306, 308 y 309). Lo segundo son las Ips de cada uno de los nodos. La barra que se ve en tercer lugar es la eficiencia del nodo, podemos configurarla a nuestro gusto aunque por defecto OpenMosix la asignará en función de como esté cada nodo. El número siguiente es la velocidad de procesamiento del nodo. El resto es lo dicho anteriormente la RAM, el procesador y el número de CPUs del nodo.

OpenMosixProcs

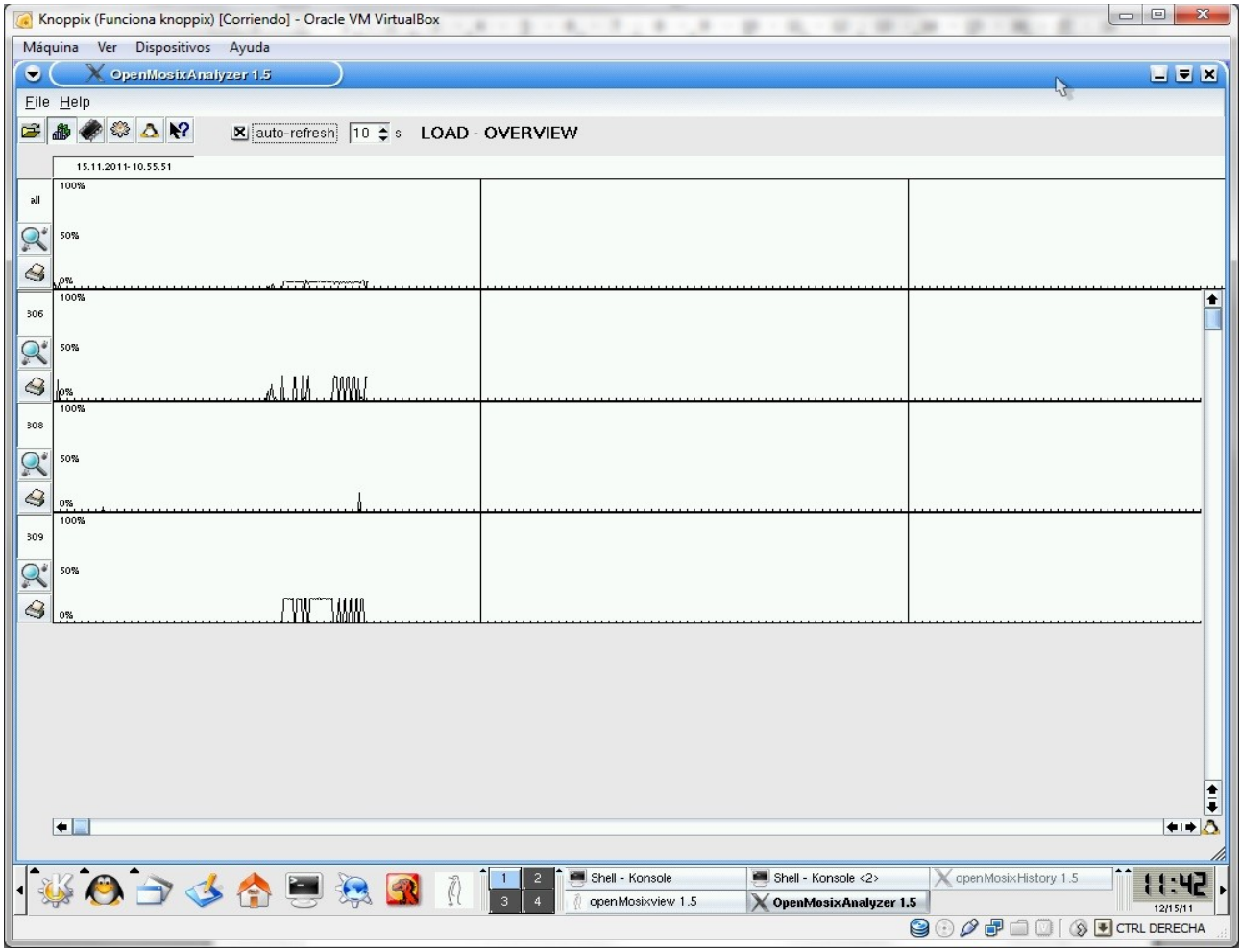
Esta herramienta es un monitor de procesos del clúster. Nos indica cuál es el número del proceso y en qué nodo se está ejecutando dicho proceso:



Los procesos que aparecen con un icono verde, como el señalado, son los procesos que han sido migrados a otros nodos del clúster. También aparece el número de migraciones que se ha hecho de ese proceso.

OpenMosixAnalyzer

Con OpenMosixAnalyzer tendremos un historial continuo de nuestro clúster en tiempo real. Los historiales generados por OpenMosixCollector (recoge información capturada por los demonios del clúster) se mostrarán de forma gráfica y continua, lo que nos permitirá ver la evolución del rendimiento y demás parámetros de nuestro clúster a través del tiempo. Si queremos analizar el rendimiento antiguo podemos abrir backup viejos y verlos en él:

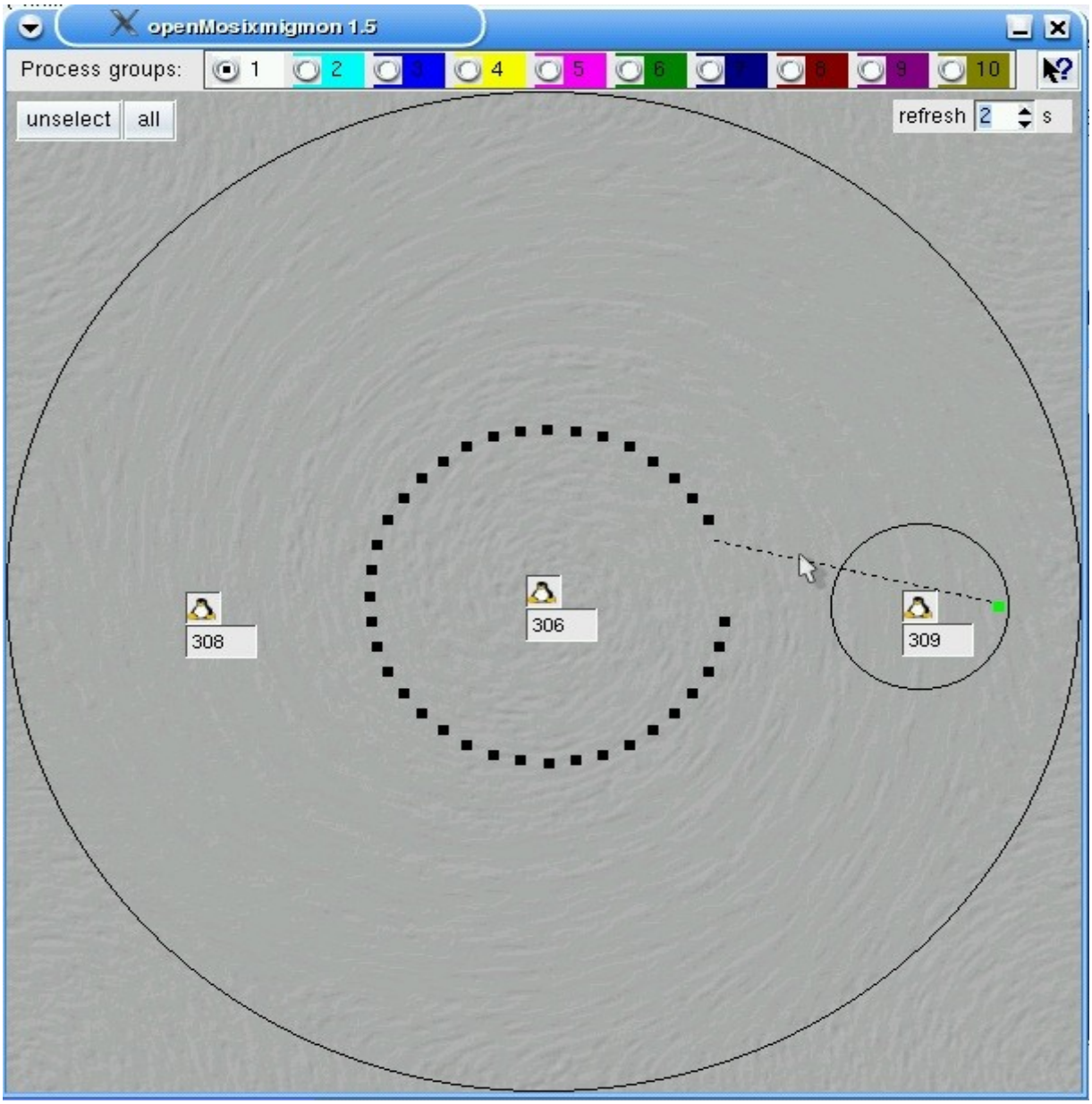


Podemos ver el historial tanto del procesador como de la memoria, así como acceder de nuevo a OpenMosixProcs para ver los procesos que se están ejecutando.

Como se observa, nos da la información total, así como de cada uno de los nodos.

OpenMosixMigmon

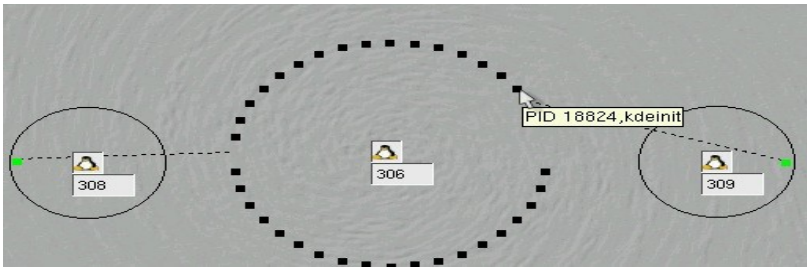
Monitoriza las migraciones de procesos que se hacen dentro del clúster.



El nodo que actúa como master, donde se está ejecutando la aplicación, aparece como un pigüino en el centro del círculo. Los pequeños cuadrados que hay a su alrededor son los procesos que se están ejecutando. Los demás pingüinos son el resto de nodos.

Cuando un proceso es migrado aparece una línea entre el cuadradito del proceso y el nodo al que ha sido migrado. (En este caso ha sido migrado al nodo número 309 desde el que actúa de máster que es el 306).

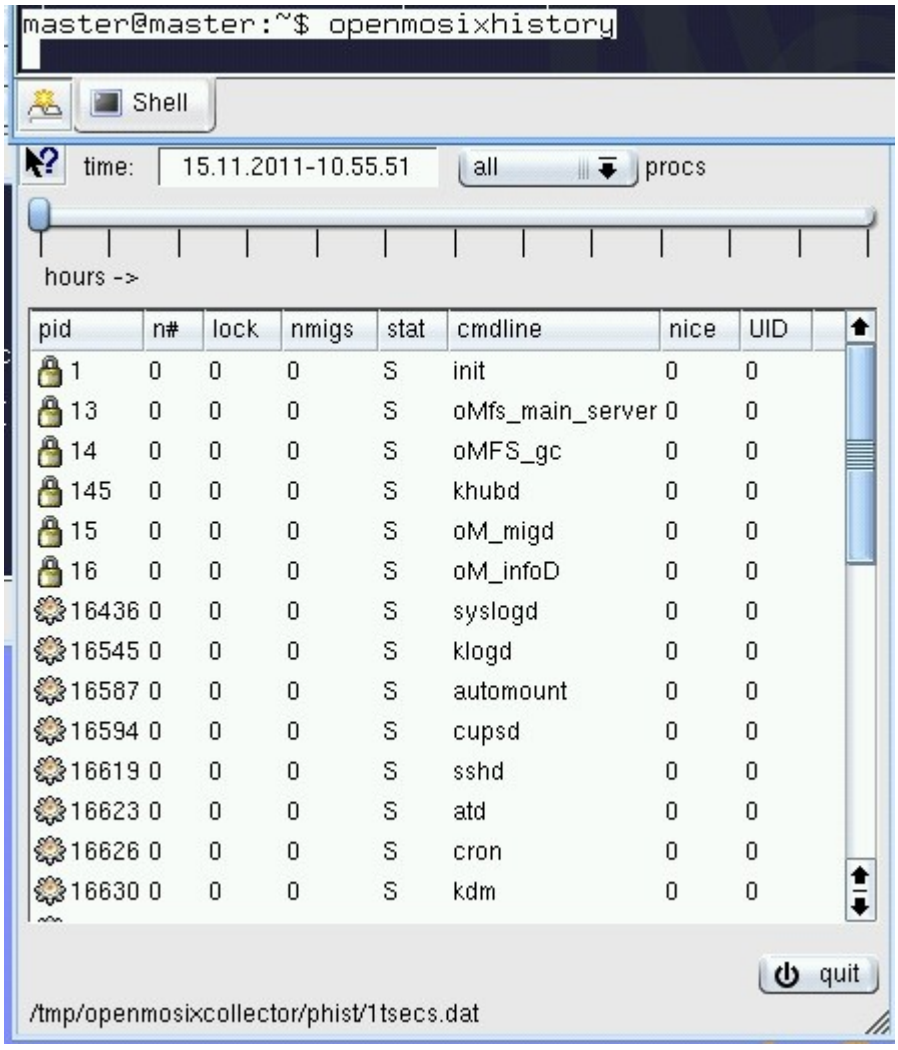
Si mantenemos el ratón sobre uno de los procesos nos mostrará su PID y la descripción del comando:



Con OpenMosixMigmon podemos migrar procesos de forma manual. Solo tenemos que arrastrar un proceso hacia uno de los nodos, y en ese momento el proceso será migrado hacia dicho nodo. Para dejar de migrarlo solo tenemos que dar doble click sobre el proceso en el nodo remoto y volverá de forma automática a la máquina que lo generó.

OpenMosixHistory

Con esta herramienta podemos acceder a la lista de procesos ejecutados en el pasado. Nos da la lista de los procesos que fueron ejectuados en cada nodo. Para iniciarlo ejecutamos el siguiente comando en una consola de usuario:

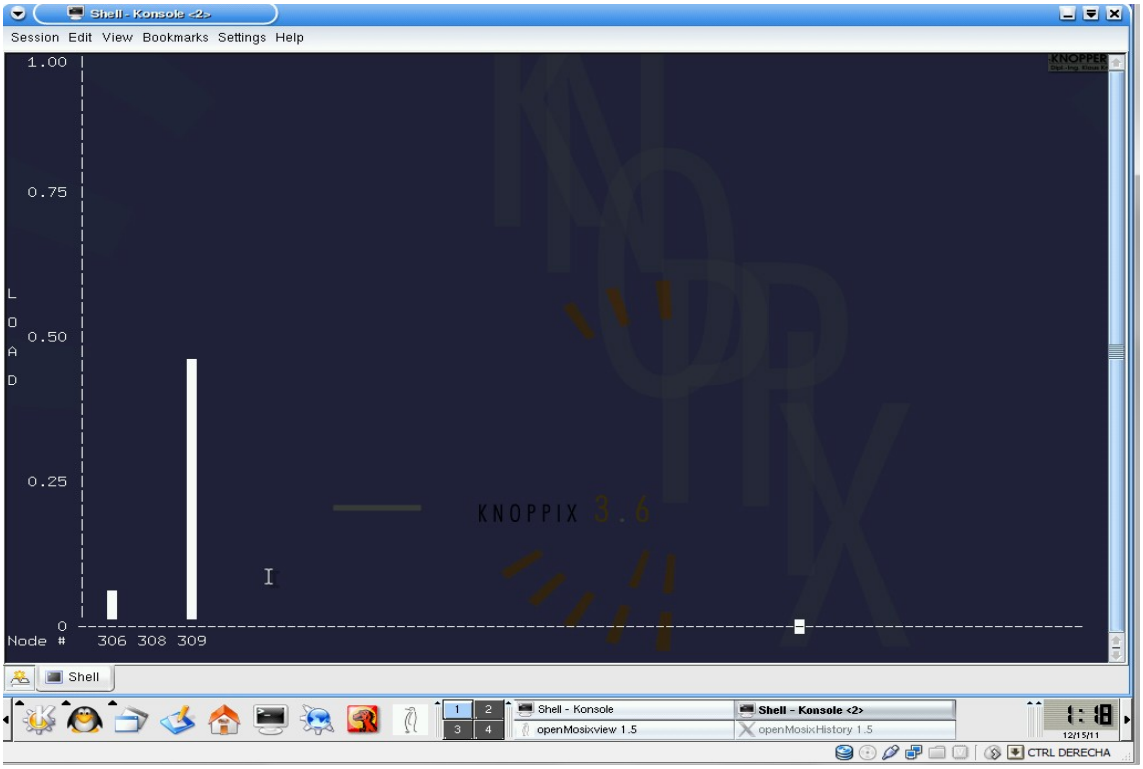


Aquí podemos ver el comportamiento de nuestro clúster en un momento específico. Con la barra de la parte superior podemos movernos por las distintas horas que el clúster ha estado funcionando. Si la máquina es reiniciada los contadores pasan a estar a 0.

Como OpenMosixProcs, ofrece información del código del proceso, la descripción y en que nodo fue ejecutado así como el número de migraciones realizadas para el proceso.

Mosmon

Esta herramienta monitorea la carga en el Clúster, la memoria disponible, memoria que fue utilizada y otras cosas en tiempo real:



Estas son todas las herramientas para la gestión y monitorización del clúster. Y ahora, ¿como podemos probar que nuestro clúster funciona?. Veámoslo.

Probando clúster con OpenMosix

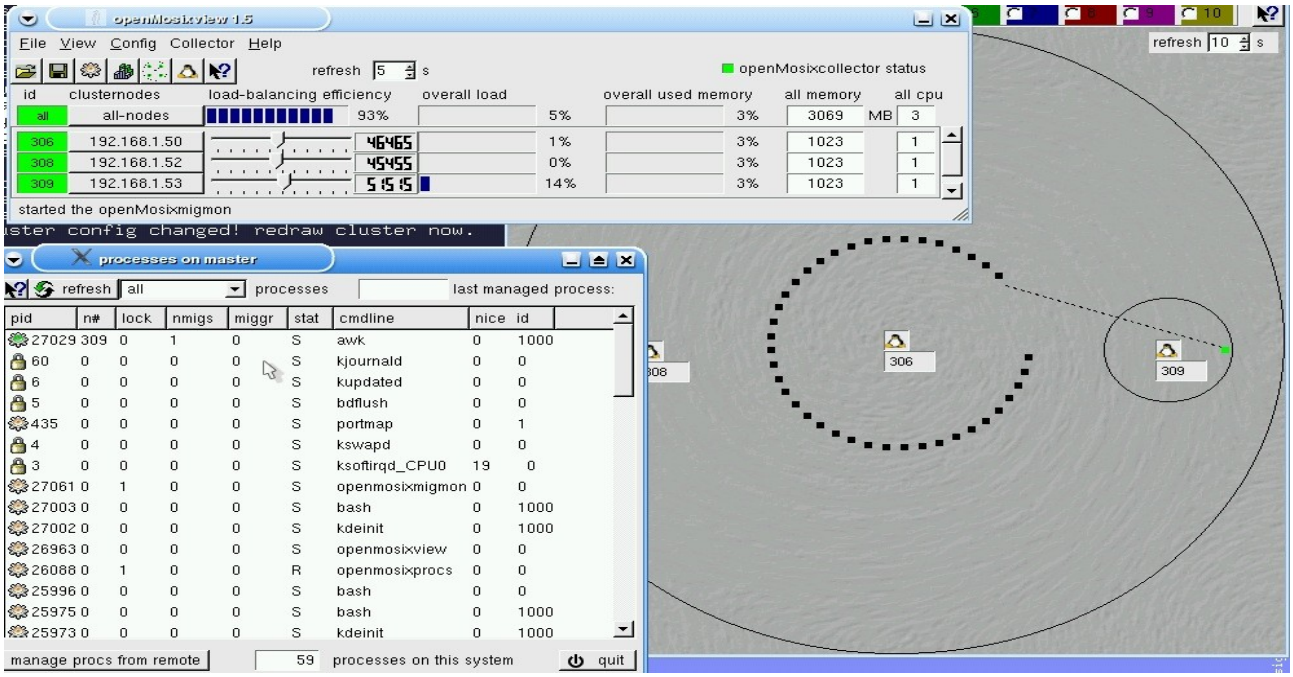
Para probar el funcionamiento de nuestro clúster lo único que hemos de hacer es ejecutar un programa cuya necesidad de proceso de cómputo sea alta. Veamos tres pruebas:

a) **Prueba simple**: para probar de forma sencilla nuestro clúster hemos creado un scrip sencillo con el siguiente código:

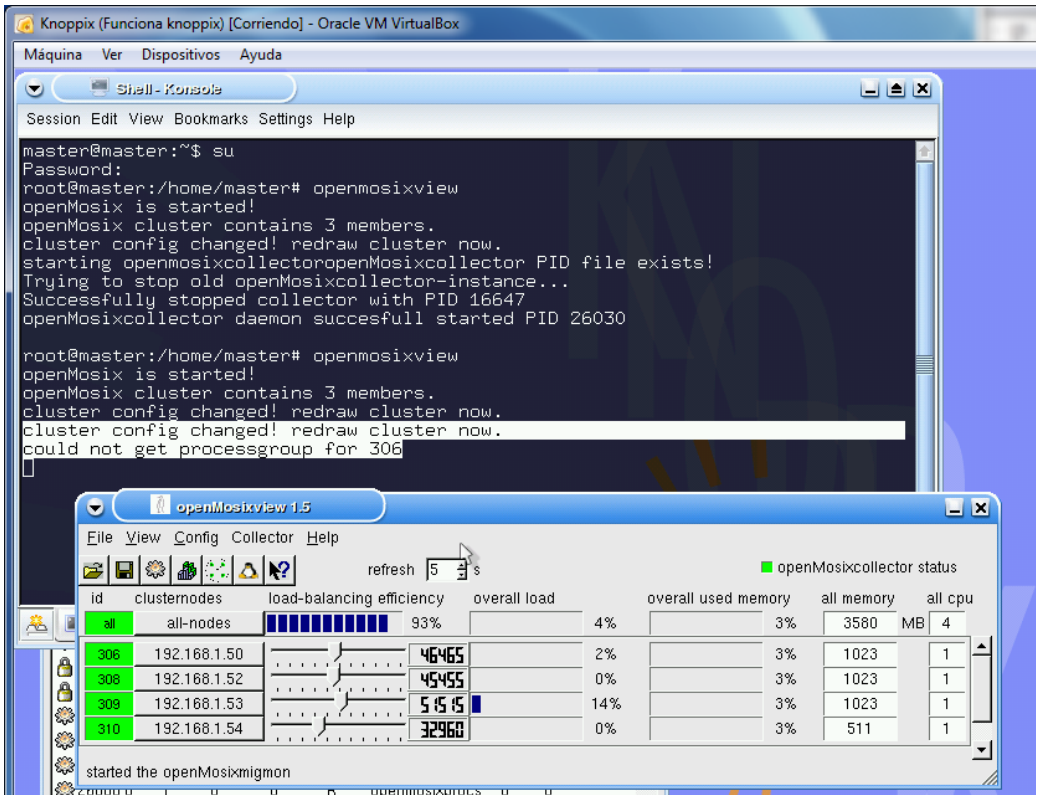
awk 'BEGIN' {for(i=0;i<100000;i++)for(j=0;j<100000;j++);}' &

Con esto solo estamos diciendo que realice un for dentro de otro for y que cuente hasta llegar 100.000 elevado a 100.000 veces. Con esta sencilla prueba nuestro clúster empezaría a funcionar y migrar procesos de forma automática.

Clúster de alto rendimiento con OpenMosix



Como se puede apreciar en la captura, el proceso con PID:27029 ha sido migrado al nodo 309 con ip: 192.168.1.53; y hasta el momento de tomarse la imagen solo había sido migrado una sola vez. Si esperáramos algo más veríamos que el proceso sería migrado a otros de los nodos para aprovechar el mayor rendimiento posible. Nuestro clúster esta funcionando de forma correcta.



Para poder saber si el demonio de OpenMosix estaba escuchando en caso de añadir un nuevo nodo, hemos levantado la máquina que habiamos creado (Knoppix-4) y

como se observa la ha reconocido de forma automática y la ha añadido al clúster.

b) **Prueba de estrés con OpenMosixTest**: El proyecto Openmosix contiene una serie de tests para que probemos el funcionamiento de nuestro clúster con el fin determinar si es correcto y ver como responde bajo carga, generando un informe con los resultados obtenidos.

Para hacerlo basta con abrir el navegador en uno de los nodos y teclear la siguiente dirección: http://openmosixview.com/omtest/

Aquí nos muestra toda la información referente al testeo del clúster como qué programas va a ejecutar y que hacen cada uno de ellos. Pichamos donde nos indica para descargar el archivo y lo hacemos.

Una vez tengamos descargado el paquete omtest-0.1-4.tar.gz lo descomprimimos con el siguiente comando:

```
master@master:~/Desktop$ gunzip omtest-0.1-4.1.tar.gz
master@master:~/Desktop$
```

Cuando lo hayamos descomprimido pasamos a desempaquetarlo con este comando:

```
master@master:~/Desktop/omtest$ tar -xvf omtest-0.1-4.1.tar
```

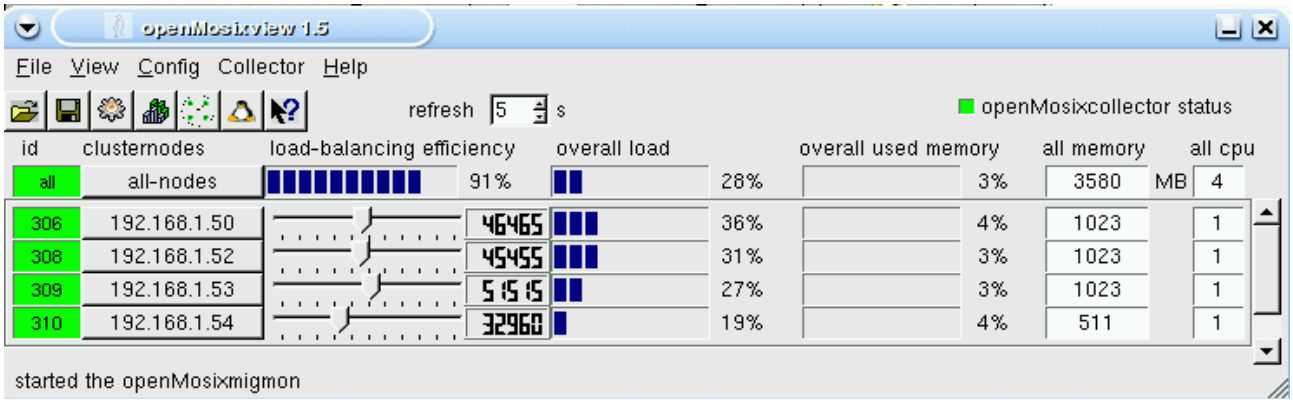
Cuando ya esté desempaquetado accedemos a la carpeta y ejecutamos la compilación e instalacion del paquete con el comando siguiente:

```
master@master:~/Desktop$ cd omtest/
master@master:~/Desktop/omtest$ ./
compile_tests.sh          ltp/                      required/
distkeygen/               make_clean.sh             run_lmbench.sh
eatmem/                   mfstest/                  start_openMosix_test.sh
fork/                     moving/                   timewaster/
lmbench/                  portfolio/
master@master:~/Desktop/omtest$ ./compile_tests.sh
```

Una vez instalado pasamos a ejecutarlo:

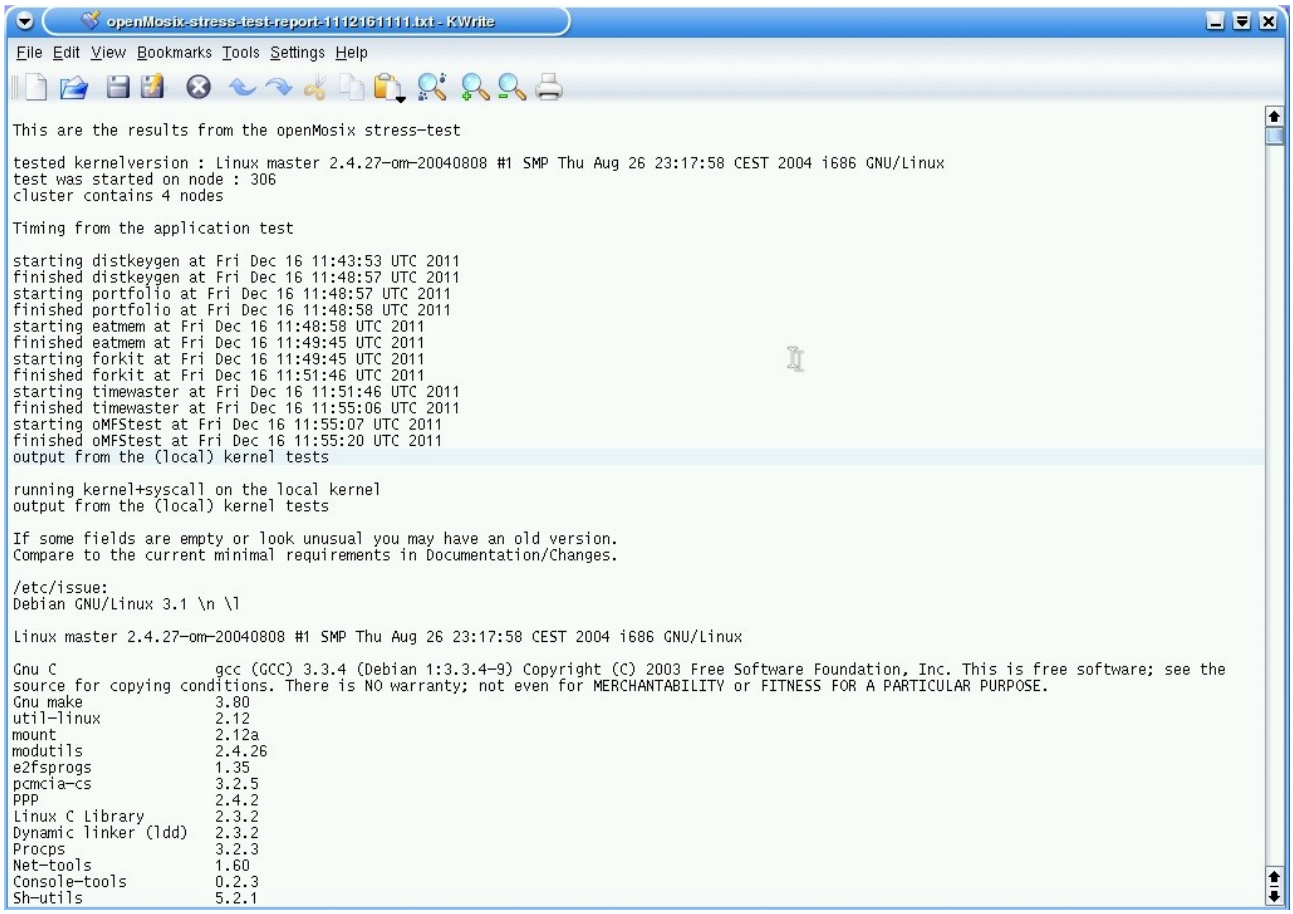
```
Installation finished!
you can run :
./start_openMosix_test.sh
to start the stress test on your cluster now

master@master:~/Desktop/omtest$ ./start_openMosix_test.sh
```

Todo está funcionando de forma correcta:



Los procesos están siendo migrados y repartidos entre los distintos nodos. Nuestro clúster reacciona a la perfección con carga.

Aquí podemos ver un informe sobre los tiempos de procesamiento de los test:

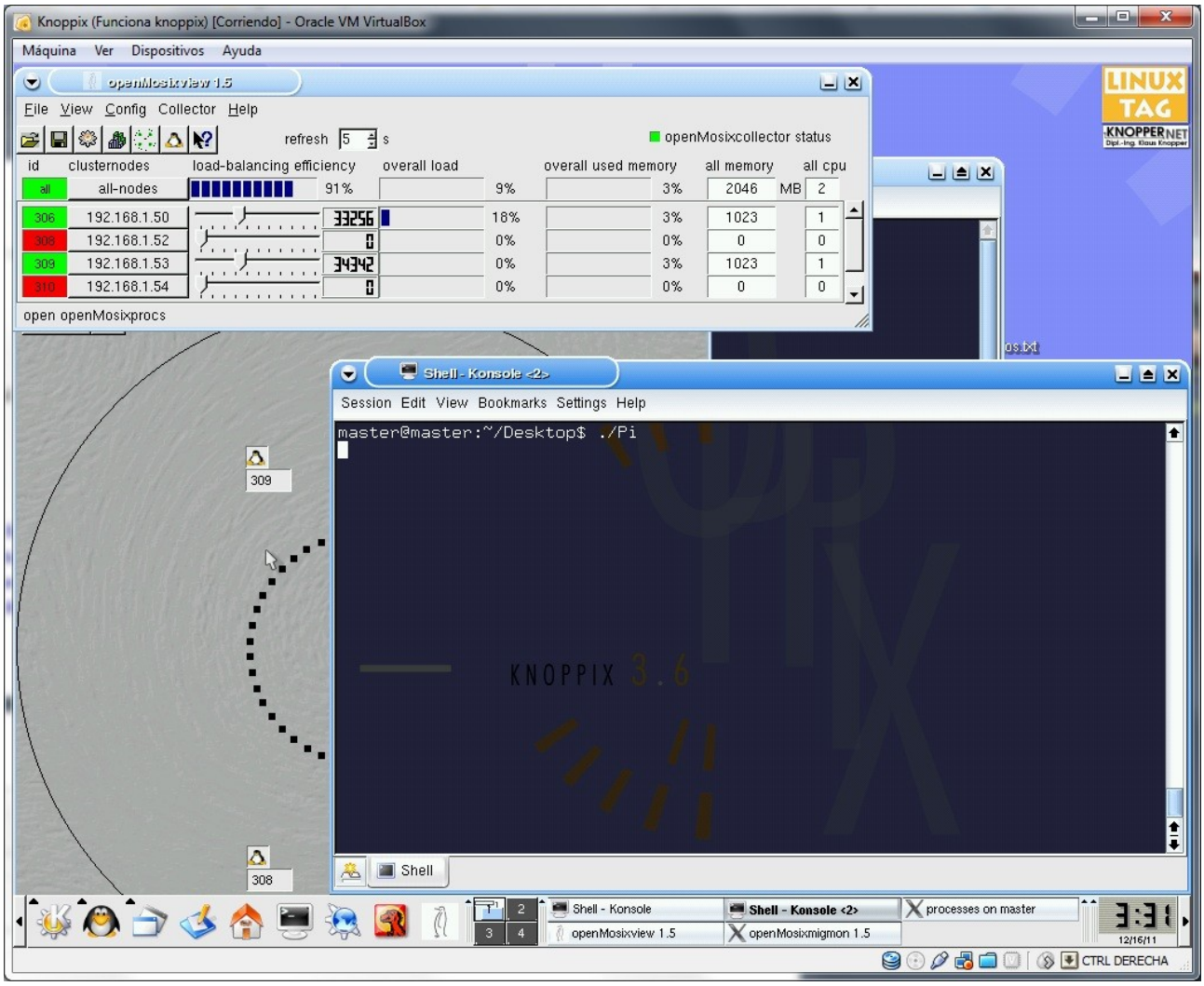


c) **Prueba geek**: Ahora pasaremos a hacer una prueba sobre un caso real. Algunos clúster construidos se dedican para operaciones aritméticas específicas como obtener números primos o sacar decimales de "pi". En nuestro caso vamos a lanzar el clúster para que busque decimales de "pi":

Primero hemos crado un scrip con el algoritmo para que busque los números con lo siguiente:

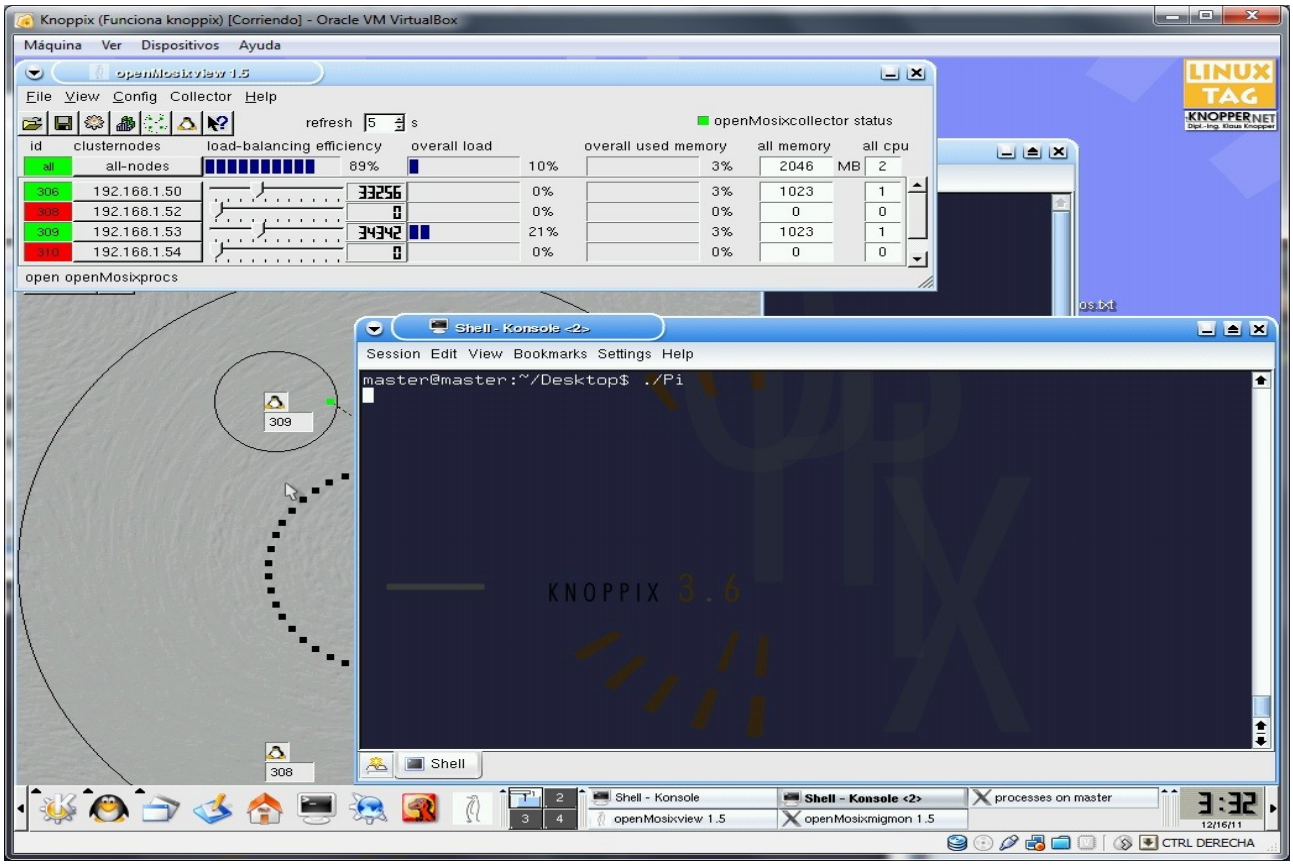
echo -e "scale=5000;4*a(1);" | bc -l

Con ello vamos a decirle al clúster que busque los 5000 primeros decimales de "pi". Veamos como se comporta el clúster al lanzarlo:

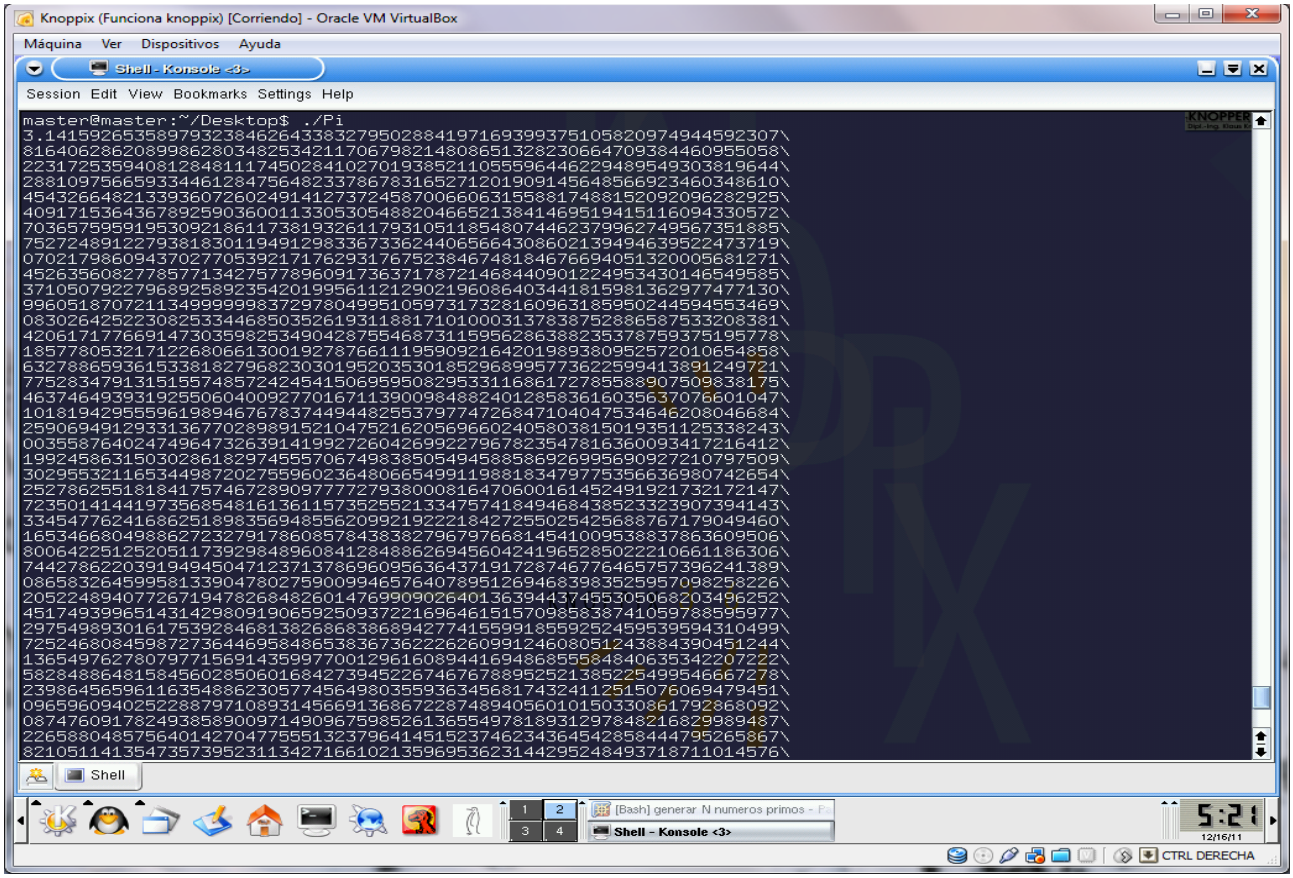


Solo lo vamos a ejecutar con 2 nodos para ver si migra procesos, luego lo haremos con todos los nodos.

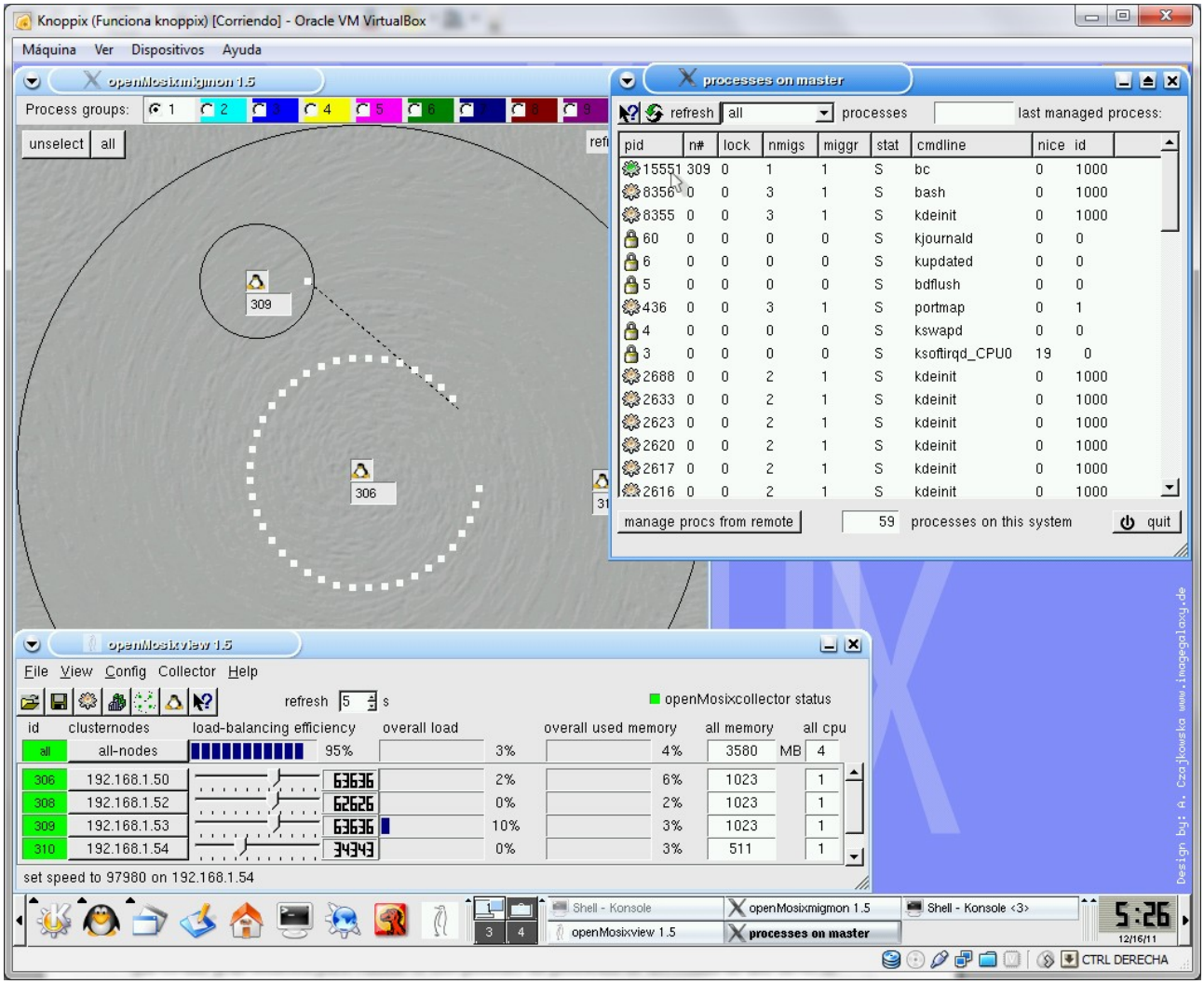
Clúster de alto rendimiento con OpenMosix



Como se observa ha migrado el proceso al nodo 309 que estaba activo. Entre los dos están sacando decimales para hacerlo en un menor tiempo:



Aquí tenemos los 5000 primero decimales de "pi". Este proceso lo hemos calculado y ha tardado 40,81 segundos en realizarlo con dos nodos. Veamos cuanto dura con 4 nodos:



El proceso ha sido migrado a otro de los nodos finalizando con un tiempo de 22,26 segundos.

Conclusiones finales

Tras analizar la construcción y el comportamiento del clúster llegamos a la conclusión de que es una gran solución para cuando se nos presente un problema de necesidad de alto nivel de computación.

Usando el clúster de openmosix tenemos la posibilidad de crear el clúster de forma sencilla, a bajo coste y con alto rendimiento. Además, al ser una aplicación con licencia GPL de software libre, no tendríamos problemas a la hora de reprogramar algunos aspectos que nos sean necesarios específicamente para la actividad que necesitemos procesar.

También con el clúster de openmosix abarcamos dos conceptos necesarios como son la velocidad y la eficencia que es lo que necesitaríamos de un clúster de alto rendimiento.

La **velocidad** o "speeedup" es el tiempo que tarda en ejecutarse un mismo programa en un solo procesador, divido entre el tiempo que toma en ejecutarse en *N* procesadores.

$s=T(1)/T(n)$

Para nosotros la velocidad sería si tardamos 4 minutos en ejecutar un programa:

s=4/1

s=4

Tendríamos una velocidad de 4. Aunque esto depende del nivel de cómputo de cada procesador en caso de que sea un clúster heterogéneo.

La **eficiencia** es la relación entre el costo computacional y el funcionamiento del clúster; y lo que indica es la eficiencia con que se está utilizando el hardware y se expresa de la siguiente forma:

e(n)=T(1)/T(n)n

Siguiendo el ejemplo de antes:

e(4)=4/4

e=1/4

e=0,25

Cuanto más nodos añadamos al clúster mayor velociad y eficiencia tendremos.

Bibliografía y webs

- http://sistemasdistribuidos4xiezar.blogspot.com/2007/03/cluster-express-con-clusterknoppix.html
- http://www.taringa.net/posts/info/928120/ClusterKnoppix_-convertir-una-red-de-PCs-en-una-superPC.html
- http://www.taringa.net/posts/linux/1784930/Creacion-de-una-super-computadora-basandose-en-cpu_s-comunes.html
- http://www.youtube.com/watch?v=ZbYo5P31vVs&feature=related
- https://forums.virtualbox.org/viewtopic.php?f=6&t=35106
- http://www.sadikhov.com/forum/index.php?/topic/149819-clusterknoppix-building-my-supercomputer-an-ongoing-project/
- http://doc.zentyal.org/es/appendix-b.html
- http://www.ubuntu-es.org/node/125232
- http://www.esdebian.org/foro/15736/alguien-pobado-clusterknoppix
- https://bbs.archlinux.org/viewtopic.php?id=114983
- http://www.eslomas.com/2005/03/como-cambiar-el-hostname-en-linux/
- http://www.ubuntu-es.org/node/78309