Pasos para crear un cluster

Sergio González González

sergio.gonzalez@hispalinux.es

Documento en el que se explican los pasos necesarios, sin entrar en profundidad, para montar dos tipos de clusters: *de alta disponibilidad* (HA) y *de alto rendimiento* (HP).

1. Introducción

La finalidad de este documento es explicar de forma breve cómo montar dos tipos de clusters: aquellos destinados a la alta disponibilidad y los destinados al alto rendimiento. El primer tipo utilizará las tecnologías: *Ultra Monkey (http://www.ultramonkey.org/)*: (*HeartBeat (http://www.linux-ha.org/heartbeat/)*, *LVS (http://www.linuxvirtualserver.org/)*, *Ldirectord (http://www.vergenet.net/linux/ldirectord/)*, *MON (http://www.kernel.org/software/mon/)*) y *NTP (http://www.ntp.org/)*, entre otras; y el segundo: OpenMosix (http://openmosix.sf.net/).

Este documento está basado en la rama de desarrollo de la distribución Debian GNU/Linux (http://www.debian.org/), más conocida como *Sid (http://www.debian.org/releases/sid/)*. Aunque los pasos que aquí se detallan son fácilmente adaptables a otras distribuciones de GNU/Linux.

También se destaca que para leer este documento se han de poseer unos conocimientos avanzados en administración de sistemas GNU/Linux, ya sean para configurar aspectos como la red, el núcleo Linux (http://www.kernel.org/) o distintas partes del sistema. Aspectos que no entran dentro de este artículo y de los cuales existe una documentación muy extensa, como la que se lista en el apartado: Documentación general sobre GNU/Linux de la Bibliografía.

1.1. Tipos de clusters: conceptos básicos

Normalmente, al diseñar un cluster se piensa en solucionar alguno de los siguientes problemas:

- Mejora de rendimiento
- Abaratamiento del coste
- Distribución de factores de riesgo del sistema
- Escalabilidad

Para solucionar este tipo de problemas, se han implementado distintas soluciones. Aquí veremos dos de ellas: cluster de alta disponibilidad y de alto rendimiento.

Nota: Los clusters de *alta disponibilidad* son bastante ortogonales a los clusters de alto rendimiento, en lo relativo a funcionalidad. Los clusters de alta disponibilidad pretenden dar servicios 24*7, de cualquier tipo, son clusters donde la principal funcionalidad es estar controlando y actuando para que un servicio, o varios, se encuentren activos durante el máximo período de tiempo posible.

Nota: Los clusters de *alto rendimiento* han sido creados para compartir el recurso más valioso de un ordenador: el tiempo de proceso. Generalmente se utilizan en ambientes científicos o en grandes empresas, donde se utilizan para la compilación o renderización. Cualquier operación que necesite altos tiempos de CPU y millones de operaciones, puede ser utilizada en un cluster de alto rendimiento, siempre que se encuentre un algoritmo que sea paralelizable. Existen clusters que pueden ser denominados de alto rendimiento tanto a nivel de sistema como a nivel de aplicación. A nivel de sistema tenemos openMosix (el que trataremos en esta documentación), mientras que a nivel de aplicación se encuentran otros como MPI, PVM, Beowulf y otros muchos. En cualquier caso, estos clusters hacen uso de la capacidad de procesamiento que pueden tener varias máquinas.

2. Cluster de alta disponibilidad, *Ultra Monkey*

Actualmente existen muchos proyectos destinados a proveer de alta disponibilidad a un sistema, uno de ellos es Ultra Monkey (http://www.ultramonkey.org/) (es el que se ha utilizado como base para esta documentación). Ultra Monkey es un proyecto que integra distintas herramientas de Software Libre para conseguir balanceo de carga y alta disponibilidad en redes de área local. Estas herramientas son: LVS, HearBeat, Ldirectord y MON, que se definirán en los siguientes apartados.

2.1. Componentes de Ultra Monkey

2.1.1. LVS (Linux Virtual Server)

LVS se implementa como un conjunto de parches al kernel Linux y un programa de espacio de usuario denominado *ipvsadm*. El sistema que tiene instalado LVS es denominado director o balanceador de carga, cuya función no es otra que balancear las peticiones de red que recibe entre un conjunto de servidores reales que se encuentran detrás de él.

LVS funciona a nivel TCP/IP, lo que se conoce como un conmutador de nivel 4. Lo que ve LVS son direcciones y puertos de origen y destino, y toma decisiones para balancear la carga con esta información. LVS toma las decisiones cuando se abre una conexión (SYN), manteniendo una tabla de conexiones, para saber a que servidor real[1] enviar un paquete perteneciente a una conexión ya establecida. Por lo tanto, el balanceo de carga que realiza LVS tiene, en principio, granularidad[2] a nivel de conexión.

LVS permite balancear muchos protocolos distintos, en principio puede balancear cualquier protocolo que trabaje en un solo puerto, y puede trabajar con protocolos que usen varios puertos, mediante persistencia o marcas de firewall.

Cuando se usan servicios persistentes, cada entrada en la tabla de LVS ya no corresponde a una conexión TCP (direcciones y puertos de origen y destino), sino que sólo usa las direcciones para identificar una conexión (se pierde granularidad).

Se puede usar iptables (http://www.iptables.org/) o ipchains para marcar los paquetes pertenecientes a un servicio virtual (con una marca de firewall) y usar esa marca para que LVS identifique los paquetes pertenecientes al servicio virtual.

LVS se ha usado con HTTP, HTTPS, Telnet, FTP, Squid, servidores de streaming QT, Real y Windows Media, incluso se ha empezado a añadirle soporte para IPSec (FreeSWAN (http://www.freeswan.org/)).

LVS realiza balanceo de carga y facilita la alta disponibilidad entre los servidores reales (si alguno deja de funcionar, se elimina del cluster mediante ipvsadm; cuando vuelva a estar operativo, se añade de nuevo con ipvsadm). Sin embargo, el balanceador de carga pasa a ser un SPOF[3], si se quiere alta disponibilidad se tiene que añadir un balanceador de respaldo y usar software de alta disponibilidad que le permita tomar el papel del balanceador de carga principal, esto lo conseguimos con HearBeat.

2.1.2. HearBeat

Esta tecnología implementa *heartbeats*, cuya traducción directa sería: «latidos de corazón». Funciona enviando periódicamente un paquete, que si no llegara, indicaría que un servidor no está disponible, por lo tanto se sabe que el servidor ha caído y se toman las medidas necesarias.

Dichos latidos se pueden enviar por una linea serie, por UDP o por PPP/UDP. De hecho los desarrolladores de HeartBeat recomiendan el uso de puertos serie por varias razones, entre las que destacan que están aislados de las tarjetas de red.

También incluye toma de una dirección IP y un modelo de recursos, incluyendo grupos de recursos. Soporta múltiples direcciones IP y un modelo servidor primario/secundario. Se ha probado satisfactoriamente en varias aplicaciones, como son: servidores DNS, servidores proxy de caché, servidores web y servidores directores de LVS. El proyecto LVS recomienda HeartBeat para aumentar la disponibilidad de su solución, pero no es parte de LVS.

En Linux-HA (http://www.linux-ha.org/) Heartbeat es un servicio de bajo nivel. Cuando un ordenador se une al cluster, se considera que el ordenador se ha unido al canal de comunicaciones, por lo tanto «late»; cuando sale, implica que ha dejado el canal de comunicaciones.

Cuando un ordenador deja de «latir» y se considera muerto, se hace una transición en el cluster. La mayoría de los mensajes de manejo del cluster que no son heartbeats se realizan durante estas transiciones.

Los mensajes de Heartbeat se envían por todas las lineas de comunicación a la vez, de esta manera, si una linea de apoyo cae, se avisará de ese problema antes de que la linea principal caiga y no haya una línea secundaria para continuar el servicio.

Heartbeat también se preocupa por la seguridad, permitiendo firmar los paquetes con CRC de 32 bits, MD5 y

SHA1. Esto puede evitar el desastre que podría provocarse si un nodo no miembro se enmascarase como nodo miembro del cluster. El problema es que el entorno donde se ejecuta Heartbeat no debe parar nunca y con suerte ese entorno se mantendrá comunicado y funcionando durante años.

Hay varias operaciones de mantenimiento de seguridad que necesitan ser efectuadas en ese tiempo, como pueden ser cambio de claves y de protocolos de autentificación. Heartbeat está preparado para esos cambios disponiendo de ficheros para la configuración.

Heartbeat tiene el problema, si no se dispone de una línea dedicada, aunque ésta sea una línea serie, al tener un tráfico que aunque pequeño es constante, suele dar muchas colisiones con otros tráficos que puedan ir por la misma red. Por ejemplo, openMosix y Heartbeat en una misma red que no tenga gran ancho de banda no funcionan bien, sobre todo si hay bastantes nodos, pues los heartbeats se envían de cualquier nodo a cualquier nodo, por lo que podrían llegar a ser un tráfico voluminoso.

2.1.3. Ldirectord

Pensado especialmente para ser usado junto con LVS, utiliza Heartbeat. Monitoriza que los servidores reales sigan funcionando periódicamente, enviando una petición a una url conocida y comprobando que la respuesta contenga una cadena concreta. Si un servidor real falla, entonces el servidor es quitado del conjunto de servidores reales y será reinsertado cuando vuelva a funcionar correctamente. Si todos los servidores fallan, se insertará un servidor de fallos, que será quitado una vez que los servidores vuelvan a funcionar. Típicamente, este servidor de fallos es el propio host desde el que se realiza el monitoraje.

2.1.4. MON (Service Monitoring Daemon)

Mon es un software para la monitorización del sistema. Mon permite definir una serie de alarmas y acciones a ejecutar cuando un servicio deja de funcionar.

Mon se utiliza ampliamente como componente de monitorización de recursos para Heartbeat.

Mon se compone de dos partes:

- *Monitores:* Son programas (escritos normalmente en Perl) que se ejecutan periódicamente para comprobar el estado de un servicio. Devuelven éxito o fallo. Hay muchos monitores escritos, y para una gran variedad de servicios, y también se pueden escribir monitores nuevos.
- *El demonio mon:* Lee un fichero de configuración, que especifica los nodos/servicios que hay que monitorizar y con que frecuencia. También especifica las acciones (alertas en la terminología de mon) a realizar cuando un nodo/servicio deja de responder o se recupera. Estas alertas también suelen ser scripts en Perl.

2.2. Consideraciones previas, el problema de los datos

En un cluster que ofrezca algún servicio en red se supone que cada servidor debe poseer los mismos datos, por ejemplo, una granja de servidores web debería compartir las mismas páginas web, un cluster de servidores POP debería compartir los mismos mensajes de correo, etc.

Esto crea un problema ¿Cómo se hace que los nodos compartan los mismos datos, sin dar lugar a conflictos?

Como en todo, para este problema van a existir diversas soluciones. La mejor solución dependerá de cuál sea el problema concreto. Por ejemplo, si tenemos un sitio web con un contenido que no cambia a menudo, puede ser suficiente hacer mirroring cada cierto tiempo, si tenemos varios sitios web que cambian continuamente de contenido, esta solución puede no ser tan buena.

De todas formas, para el ejemplo que aquí mostraremos, vamos a suponer que cada servidor real posee sus propios datos.

2.3. Instalación de Ultra Monkey

Los pasos que vamos a realizar para instalar Ultra Monkey son los siguientes:

- Configurar las fuentes de *APT*
- Actualizar la configuración de los módulos
- Actualizar el *kernel*
- Actualizar el gestor de arranque
- Reiniciar
- Instalar los paquetes restantes
- Configurar el sistema

2.3.1. Configurar las fuentes de APT

Partiendo de una distribución Debian GNU/Linux correctamente instalada, el primer paso que hemos de realizar, es añadir las siguientes fuentes a nuestro `/etc/apt/sources.list`:

```
deb http://www.ultramonkey.org/download/2.0.1/ sid main
deb-src http://www.ultramonkey.org/download/2.0.1 sid main
```

Estas dos líneas nos van a proveer de los paquetes (tanto en formato fuente como en binario) necesarios para poner en marcha un sistema de alta disponibilidad con Ultra Monkey.

> **Importante:** Aunque las instalación la realizaremos sobre Sid, la distribución en desarrollo de Debian, los mismos pasos se pueden seguir para Woody, la distribución estable de Debian; la única diferencia son las fuentes a añadir. En caso de utilizar la distribución estable, hemos de añadir las siguientes fuentes a nuestro `/etc/apt/sources.list`:
>
> ```
> deb http://www.ultramonkey.org/download/2.0.1/ woody main
> deb-src http://www.ultramonkey.org/download/2.0.1 woody main
> ```

Ahora actualizamos la base de datos de paquetes de nuestra distribución con el siguiente comando:

```
# apt-get update
```

2.3.2. Actualizar la configuración de los módulos

Como el núcleo que instalaremos a continuación usa una imagen initrd, que provee los módulos necesarios para arrancar el sistema, es especialmente importante que la configuración de los módulos de su sistema está al día. En particular, es necesario que cualquier módulo necesario para arrancar el sistema, como los controladores SCSI, estén presentes en su `/etc/modules`. Una vez añadidos los módulos necesarios al archivo anterior, ejecute **update-modules** antes de instalar el núcleo.

2.3.3. Actualizar el kernel

Ultra Monkey pone a su disposición LVS (Linux Virtual Server), que no está disponible en el núcleo por defecto de la distribución Debian Sid. Por este motivo, se necesita un núcleo modificado.

Es recomendable que instale este núcleo tanto en los ordenadores destinados a ser directores, como en los servidores reales. Este núcleo provee la posibilidad de ocultar sus interfaces para que no respondan a las peticiones arp[4].

La forma más fácil de hacer esto es instalar uno de los núcleos empaquetados que provee Ultra Monkey para Debian GNU/Linux. Estos se pueden instalar con **apt-get** ejecutando uno de los siguientes conjuntos de comandos[5]:

- Arquitectura 386:

```
# apt-get install kernel-image-2.4.20-3-ipvs-386
# apt-get install kernel-headers-2.4.20-3-ipvs-386
# apt-get install kernel-pcmcia-modules-2.4.20-3-ipvs-386
```

- Arquitectura Pentium:

```
# apt-get install kernel-image-2.4.20-3-ipvs-586tsc
# apt-get install kernel-headers-2.4.20-3-ipvs-586tsc
# apt-get install kernel-pcmcia-modules-2.4.20-3-ipvs-586tsc
```

- Arquitecturas Pentium Pro, Celeron, Pentium II, Pentium II o Pentium IV:

```
# apt-get install kernel-image-2.4.20-3-ipvs-686
# apt-get install kernel-headers-2.4.20-3-ipvs-686
# apt-get install kernel-pcmcia-modules-2.4.20-3-ipvs-686
```

- Arquitecturas SMP Pentium Pro, Celeron, Pentium II, Pentium II y Pentium IV:

```
# apt-get install kernel-image-2.4.20-3-ipvs-686-smp
# apt-get install kernel-headers-2.4.20-3-ipvs-686-smp
# apt-get install kernel-pcmcia-modules-2.4.20-3-ipvs-686-smp
```

- Arquitecturas AMD K6, K6-II o K6-III:

```
# apt-get install kernel-image-2.4.20-3-ipvs-k6
# apt-get install kernel-headers-2.4.20-3-ipvs-k6
# apt-get install kernel-pcmcia-modules-2.4.20-3-ipvs-k6
```

- Arquitecturas AMD Duron o Athlon:

  \# **apt-get install kernel-image-2.4.20-3-ipvs-k7**

  \# **apt-get install kernel-headers-2.4.20-3-ipvs-k7**

  \# **apt-get install kernel-pcmcia-modules-2.4.20-3-ipvs-k7**

- Arquitecturas SMP AMD Duron o Athlon:

  \# **apt-get install kernel-image-2.4.20-3-ipvs-k7-smp**

  \# **apt-get install kernel-headers-2.4.20-3-ipvs-k7-smp**

  \# **apt-get install kernel-pcmcia-modules-2.4.20-3-ipvs-k7-smp**

De todas maneras, si desea compilar su propio núcleo, ha de tener en cuenta que necesita los parches *IPVS* y *Hidden Interface* para ejecutar Ultra Monkey. Estos parches se pueden instalar desde paquetes *deb* usando el comando **apt-get**. Para ello teclee:

\# **apt-get install kernel-patch-2.4-hidden-interface**

\# **apt-get install kernel-patch-2.4-ipvs**

Los pasos necesarios para parchear y compilar en núcleo para obtener el soporte de LVS no se verán en esta documentación. De todas formas, si está familiarizado con este tipo de actividades, no le será muy difícil obtenerlo.

2.3.4. Actualizar el gestor de arranque

El núcleo instalado hace uso de una imagen initrd para proveer los módulos en arranque del sistema. Debido a esto, debe asegurarse que su archivo de configuración de LILO: `/etc/lilo.conf`, posea una línea para el initrd y que se haya actualizado para el núcleo actual. Un ejemplo de un archivo `/etc/lilo.conf` que cumple estas características se puede ver a continuación:

```
lba32
boot=/dev/sda
root=/dev/sda3
install=/boot/boot-menu.b
map=/boot/map
delay=20
vga=normal

default=Linux

image=/boot/vmlinuz-2.4.20-3-ipvs-686
        label=Linux
        read-only
        initrd=/boot/initrd.img-2.4.20-3-ipvs-686

image=/vmlinuz
        label=LinuxOLD
        read-only
        optional
```

Una vez que ha actualizado su archivo `/etc/lilo.conf`, ejecute:

```
# /sbin/lilo -v
```

2.3.5. Reiniciar

Como se ha instalado un nuevo núcleo, necesitará reiniciar el sistema para que los cambios tengan efecto. Para ello ejecute:

```
# /sbin/shutdown -r now
```

2.3.6. Instalar los paquetes restantes

Ultra Monkey viene con algunos paquetes adicionales. Estos paquetes sólo son necesarios para aquellos sistemas que van a ejecutar HearBeat y pueden ser obtenidos usando **apt-get**, como se muestra a continuación:

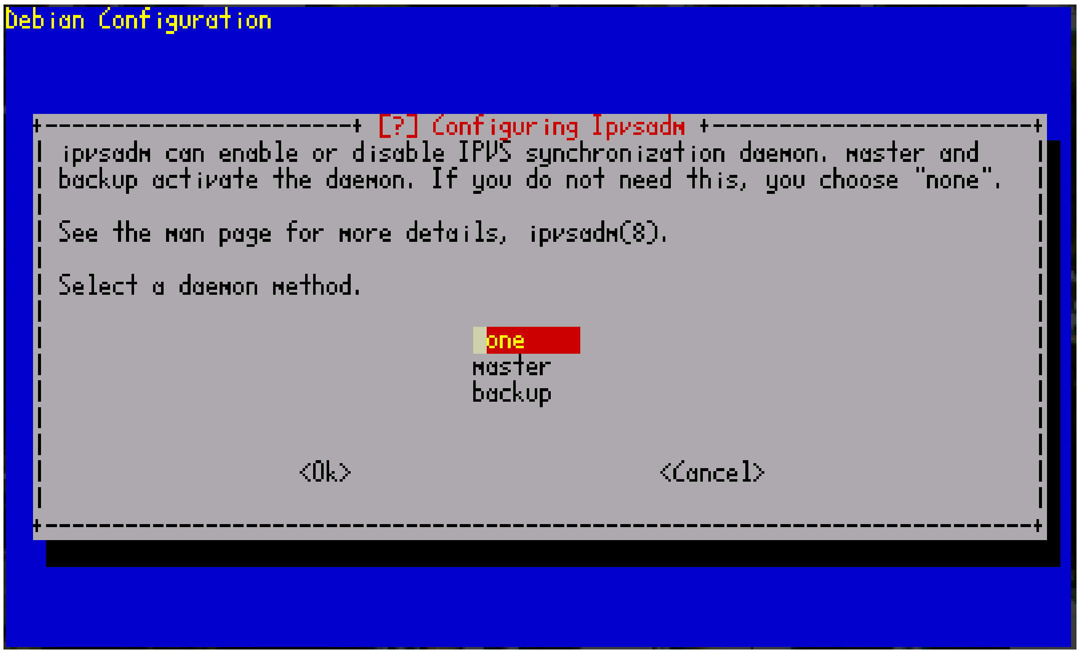
```
# apt-get install ultramonkey
```

Durante la instalación de los paquetes necesarios, se realizarán una serie de preguntas con el objeto de configurar su sistema. Ha de responderlas de forma que se adapten a sus requerimientos y sistema[6].

Hemos de tener especial cuidado a la hora de configurar el paquete ipvsadm, durante la instalación del mismo le preguntará para configurar el archivo `/etc/ipvsadm.rules`. Ha de responder <No>, ya que de sino interferirá con la forma que tiene Ultra Monkey de configurar ipvsadm.

Pantalla de configuración de ipvsadm

Otra de las preguntas que le hará el sistema al instalar ipvsadm será para configurar el demonio de sincronización de IPVS. Es recomendable que no seleccione nada para la configuración del demonio de sincronización, ya que en algunos casos interfiere en la forma en que Ultra Monkey ejecuta LVS.



Pantalla de configuración de ipvsadm

Para finalizar, instalaremos el paquete mon y el paquete ntp-simple[7], para lo cual ejecutaremos:

```
# apt-get install mon ntp-simple
```

Una vez finalizada la instalación, «sólo» nos queda configurar MON para adaptarlo a nuestras necesidades. Puede consultar la página web (http://www.kernel.org/software/mon/) de la aplicación, así como leer la documentación que ha instalado el propio paquete (`/usr/share/doc/mon/`) y ver los scripts de ejemplo que provee dicho paquete (`/usr/share/doc/mon/examples`).

2.3.7. Configurar el sistema

La página de Ultra Monkey provee de una gran variedad de ejemplos y formas de configuración de un cluster de alta disponibilidad. Por este motivo, remitimos a la página de topologías (http://www.ultramonkey.org/2.0.1/topologies/) de dicho proyecto para saber más acerca de las distintas posibilidades y formas de configurar nuestro cluster.

3. Cluster de alto rendimiento, *OpenMosix*

Los pasos para montar un cluster de alto rendimiento con OpenMosix son muy simples:

- Parchear, configurar y compilar el núcleo Linux con OpenMosix. Una forma de hacerlo, es instalando el parche de OpenMosix para el núcleo, mediante apt-get. Para ello teclee:

  ```
  # apt-get install kernel-patch-openmosix
  ```

  Una vez instalado el núcleo, hemos de reiniciar el sistema.

- El segundo paso es instalar las utilidades de administración para OpenMosix, para ello tecleamos:

  ```
  # apt-get install openmosix
  ```

- Adicionalmente podemos instalar el programa openMosixview:

  ```
  # apt-get install openmosixview
  ```

Una vez tenemos todas las aplicaciones necesarias, sólo nos queda configurar el cluster para adaptarlo a nuestras necesidades. Como los pasos para realizar esta tarea son muy simples y están perfectamente detallados en el The OpenMosix HOWTO, no los voy a repetir aquí, por lo que le remito a dicho manual para leerlos.

3.1. ClusterKnoppix, un cluster con un *Live CD*

Aun siendo extremadamente sencilla la instalación de OpenMosix, se puede facilitar aun más gracias a Win Vandersmissen. Win ha adaptado una distribución KNOPPIX (http://www.knoppix.org/) [8] para que nada más arrancar, configure y ponga en funcionamiento un cluster con OpenMosix transparentemente al usuario. La distribución se llama ClusterKnoppix.

Para hacer uso de esta distribución, no tenemos más que grabar la imagen ISO en un CD y arrancar el ordenador con este CD en la unidad de CD-ROM. Después de un rato, la distribución habrá detectado el hardware del ordenador, lo habrá configurado y habrá arrancado el cluster OpenMosix, buscando inmediatamente nuevos nodos (transparentemente), que serán añadidos al cluster en caso de ser encontrados.

Aviso

Le recomiendo, que nada más arrancar la distribución, establezca una clave para el usuario *knoppix*. Esto se debe a que la distribución ClusterKnoppix hace uso de SSH para la comunicación entre nodos, y si no se ha establecido una clave para dicho usuario, será imposible la comunicación.

Para realizar esto, abra una consola y teclee lo siguiente:

```
$ sudo bash
# passwd knoppix
```

Tras lo cual, ya puede teclear una nueva clave para dicho usuario. Repita este procedimiento para cada nodo.

Para ver el estado del cluster, puede hacer uso de openMosixview, aplicación que le permite configurar y administrar el cluster de una manera muy cómoda (vea el El manual para el clustering con openMosix para más información). Arranque la aplicación desde una consola, para ver los mensajes que «lanza» y para poder teclear la clave del usuario *knoppix*en las conexiones SSH que realiza.

A. Sobre este documento

Bibliografía

Documentación sobre clusters

[El manual para el clustering con openMosix (http://alumnes.eup.udl.es/~b4767512/07.openMosix/oM_como.html)] Miquel Catalán i Coït.

<mikel@akamc2.net>

[The OpenMosix HOWTO (http://howto.ipng.be/openMosix-HOWTO/)] Kris Buytaert.

<buytaert@stone-it.be>

[Documentación sobre un Web Farm para la Herramienta de E-Learning de Hispalinux (no disponible «en línea»)] Pablo de la Red Blanco, Alfonso Escribano Merino, César Estébanez.

<pablo@delared.com>

<alfons@est.unileon.es>

<cesar@lacaja.net>

[Documentación sobre clusters para servicio web (Webfarms) (no disponible «en línea»)] Luis Javier Merino Morán, Alfonso Escribano Merino.

<ninjalj@lycos.com>

<alfons@est.unileon.es>

Documentación general sobre GNU/Linux

[The Linux Documentation Project (http://www.tldp.org/)]

[TLDP-ES/LuCAS (http://es.tldp.org)]

Programas utilizados

[FreeSWAN (http://www.freeswan.org/)]

[HeartBeat (http://www.linux-ha.org/heartbeat/)]

[Iptables (http://www.iptables.org/)]

[Ldirectord (http://www.vergenet.net/linux/ldirectord/)]

[Linux-HA (http://www.linux-ha.org/)]

[LVS (Linux Virtual Server) (http://www.linuxvirtualserver.org/)]

[MON (http://www.kernel.org/software/mon/)]

[NTP (http://www.ntp.org/)]

[OpenMosix (http://openmosix.sf.net/)]

[openMosixview (http://www.openmosixview.com/)]

[Ultra Monkey (http://www.ultramonkey.org/)]

Distribuciones

[ClusterKnoppix (http://bofh.be/clusterknoppix/)] Win Vandersmissen, 2003.

`<clusterknoppix@bofh.be>`

[Debian (http://www.debian.org/)]

Notas

1. Los servidores reales son aquellos ordenadores que atienden a los clientes una vez el nodo director les ha pasado el «trabajo». Estos ordenadores son los que poseerán el demonio HTTP, FTP, etc.
2. El término granularidad se usa como el mínimo componente del sistema que puede ser preparado para ejecutarse de manera paralela
3. *Single Point Of Failure* (Punto Simple de Fallo): cualquier elemento cuyo fallo provoca el fallo de todo el sistema, por ejemplo la alimentación. Una forma de evitarlo es la duplicación del servicio.

4. Para más detalles sobre esta cuestión, lea la sección dedicada a LVS (http://alumnes.eup.udl.es/~b4767512/07.openMosix/manual_html/node17_mn.html#SECTION00725000000000000000) del El manual para el clustering con openMosix.
5. Los comandos segundo y tercero, que instalan las cabeceras y los módulos pcmcia, respectivamente, son opcionales.
6. Una vez más, recomiendo la lectura del El manual para el clustering con openMosix para comprender mejor las preguntas realizadas y cómo se ha de configurar el sistema.
7. NTP es un demonio destinado a mantener los relojes del sistema sincronizados, la única opción de configuración que hemos de establecer al instalarlo es el servidor ntp desde el cual actualizaremos nuestros equipos.
8. KNOPPIX es una distribución de GNU/Linux que se ejecuta completamente desde un CD-ROM. Está basada en Debian GNU/Linux, y contiene aplicaciones como OpenOffice.org, The Gimp!, Mozilla, KDE y miles de aplicaciones libres más. Toda esta información (2 GB aprox.) entra en un CD de 700MB, gracias al uso de compresión.