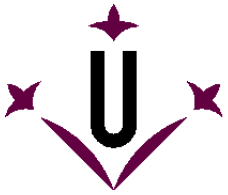
# Despliegue desatendido de GNU/Linux

Dep. de Informática e Ingeniería Industrial

Universitat de Lleida




Adrián Gibanel López

Director: Francesc Solsona Tehàs

Febrero 2009

# Resumen

Las nuevas formas de cómputo en red, mayormente distribuido como los clusters implican mucho mantenimiento. El administrador ha de montar una infraestructura de hardware que le permita con el menor costo posible realizar el mayor trabajo posible. Una vez montada la infraestructura llega el momento de la configuración software de la misma.

Cada nodo de los clusters necesita ser instalado con un sistema operativo. Este sistema operativo ha de ser para trabajar en grupo con el resto de nodos. Para automatizar las instalaciones repetitivas en sistemas Debian tenemos a nuestra disposición el paquete de software: Fully Automatic Installation. FAI no sólo nos permite realizar instalaciones automatizadas sino que también nos permite actualizar las máquinas.

En este proyecto se detallará la instalación y configuración de Fully Automatic Installation para paquetes estándar. Veremos cómo podemos optimizar el sistema instalado para reducir costes. Así mismo se explicará como modificar los paquetes oficiales de Ubuntu 8.10 para que los paquetes de Fully Automatic Installation puedan servir a los nodos del cluster la distribución Ubuntu 8.10 sin ningún problema.

# Índice general

# Índice de figuras

# Capítulo 1

# Introducción

La idea original del proyecto era conseguir un equivalente de la distribución de Linux orientada a clusters Rocks.

Rocks and Rolls es una distribución de Linux con un conjunto de herramientas diseñadas para el despliegue y administración de sistemas de computación, almacenamiento y visualización distribuida. Esta diseñada de manera que la administración sea lo más simple posible a la par que es flexible y personalizable.

Las características de Rocks son las siguientes:

- Ofrece instalación sistemática y masiva del sistema operativo de los nodos basada en el sistema Kickstart de Red Hat.
- Monitoreo del proceso de instalación en el cliente.
- Sincronización en la actualización de los nodos por reinstalación.
- Replicación de la información de autentificación en cada nodo.
- Rolls: Conjuntos paquetes para poder personalizar el perfil de la plataforma.

Para ello necesitamos tener un servidor que nos proporcione por un lado la capacidad de instalación automatizada de los nodos y por otro lado la distribución de cómputo organizada entre los nodos. Este proyecto se encargará de la instalación automatizada de los nodos.

En la instalación automatizada de los nodos se puede diferenciar varias funciones que debe realizar el servidor encargado de la misma.

La primera función es la de gestor de grupos de pcs. En otras palabras el servidor ha de permitir la definición de grupos de pcs en base al identificativo de cada pc y asignarle unas determinadas tareas o programas.

Para que estos equipos inicien la instalación automatizada necesitarán un arranque por red. Así pues el servidor realizará también la función de servir el arranque. Es decir, servirá un cargador de arranque de red con su configuración adecuada a las tareas que le hemos asignado a cada uno de los nodos.

Para poder servir una instalación automatizada necesitamos primero generarla. Así pues el servidor debe poder generar todo un sistema de instalación automático por red que contemple las necesidades de cada nodo.

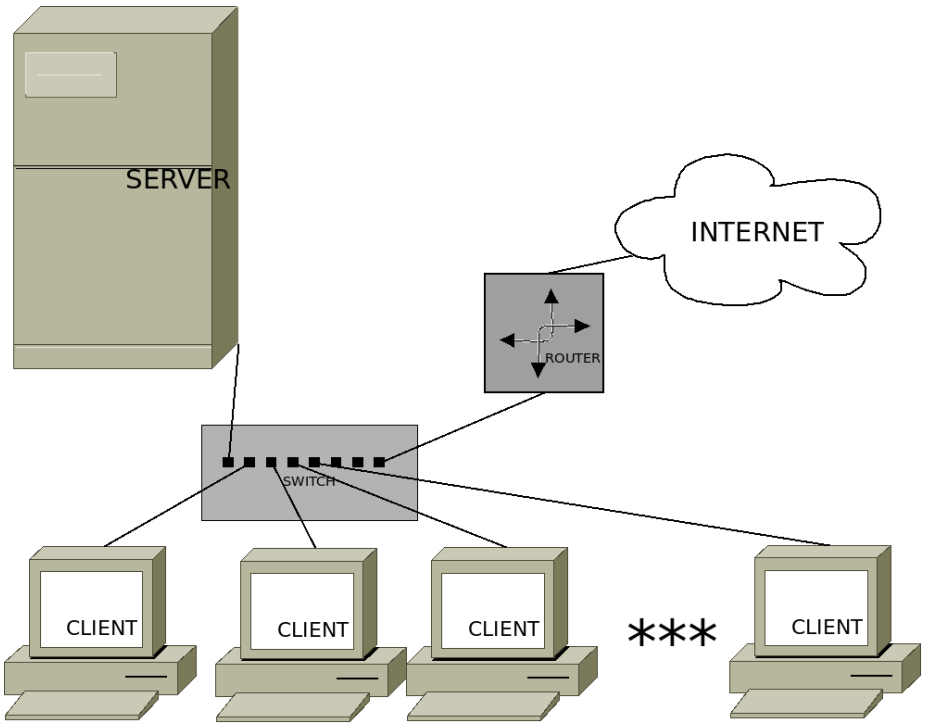



Figura 1.1: Esquema del cluster prototipo del proyecto

El sistema al estar basado en una distribución tipo Debian implica que los paquetes a instalar los nodos serán de tipo Debian y por tanto deberemos conseguirlos de un repositorio. El servidor ha de ser un repositorio ya que si esto no fuera así todos los nodos irían a buscar los mismos paquetes al exterior.

Finalmente todo sistema de software repite un ciclo de desarrollo en la que su actualización acaba tomando la mayor parte del tiempo. Así pues el sistema también debe proporcionar una actualización adecuada que contemple las particularidades de cada nodo.

Todos estos requisitos son en su mayoría suplidos mediante el paquete de software Fully Automatic Installation que está presente en la distribución Ubuntu 8.10. El resto de los requisitos se pueden fácilmente implementar con herramientas de la misma distribución.

Una vez presentadas las necesidades y el software que las soluciona, se estudiará el sistema dónde se quiere probar. El cluster prototipo del proyecto está representado en la figura 1.1. Un ordenador que hace de servidor de trabajos de cómputo, realiza también el servicio de instalaciones automatizadas. Gracias a un switch se conecta a la red local de ordenadores de cómputo. Finalmente gracias a un router que puede ser el mismo servidor podemos conseguir acceso al exterior del cluster. Este cluster se puede extender a multi-cluster si a los nodos se les sirven diferentes distribuciones como se puede observar en la figura 1.2.

En este capítulo veremos algunos conceptos que nos ayudarán a seguir la lectura de la memoria. En la sección Contexto y motivación veremos el cluster de

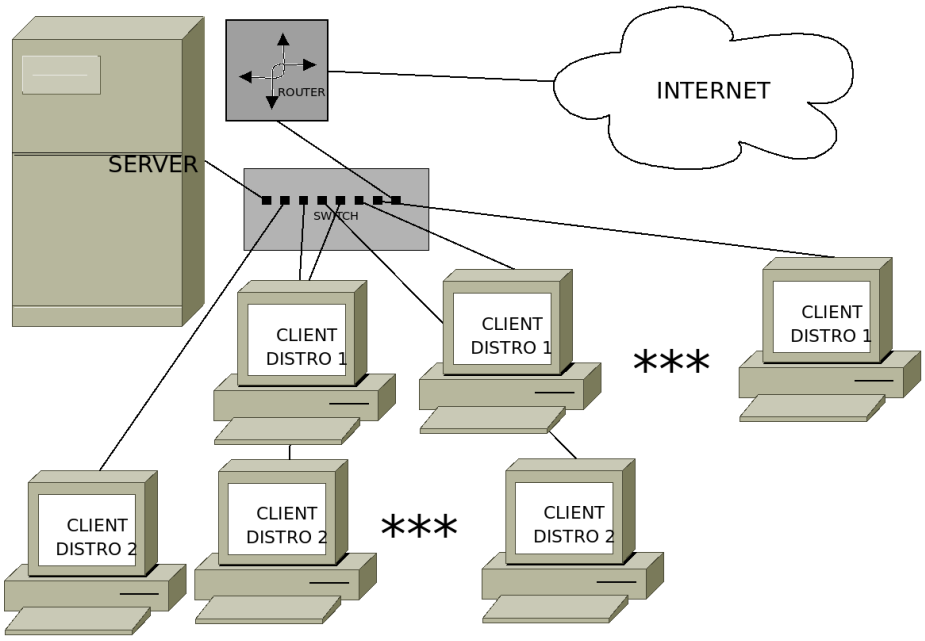



Figura 1.2: Esquema de un multi-cluster

cómputo dónde se originó la idea del proyecto. En los objetivos nos marcaremos unas metas a conseguir centradas en la automatización de la instalación. Por último en el Contenido de la memoria veremos cómo se ha estructurado las diferentes partes del proyecto dentro de la memoria.

## 1.1. Conceptos Previos

### 1.1.1. Fully Automatic Installation (FAI)

Después de buscar diferentes paquetes de software orientados a instalaciones automatizadas que cumplieran los requisitos antes descritos se optó por la herramienta Fully Automatic Installation (FAI). FAI es un conjunto de paquetes software para distribuciones basadas en Debian orientado a la instalación automática de distribuciones GNU/Linux. FAI será la base principal de Desdeslin. Para más información véase el manual de FAI en [1].

### 1.1.2. Cluster

Un cluster es un conjunto de ordenadores que trabajan en red con un fin común. Normalmente se dedican a realizar un cómputo que con una sola máquina tardaría mucho más tiempo a completarse. La estructura de los cluster suele ser servidor-cliente. El servidor manda trabajos como subdivisiones del trabajo principal a cada uno de los nodos que responden con la salida del trabajo que ellos ejecutan.

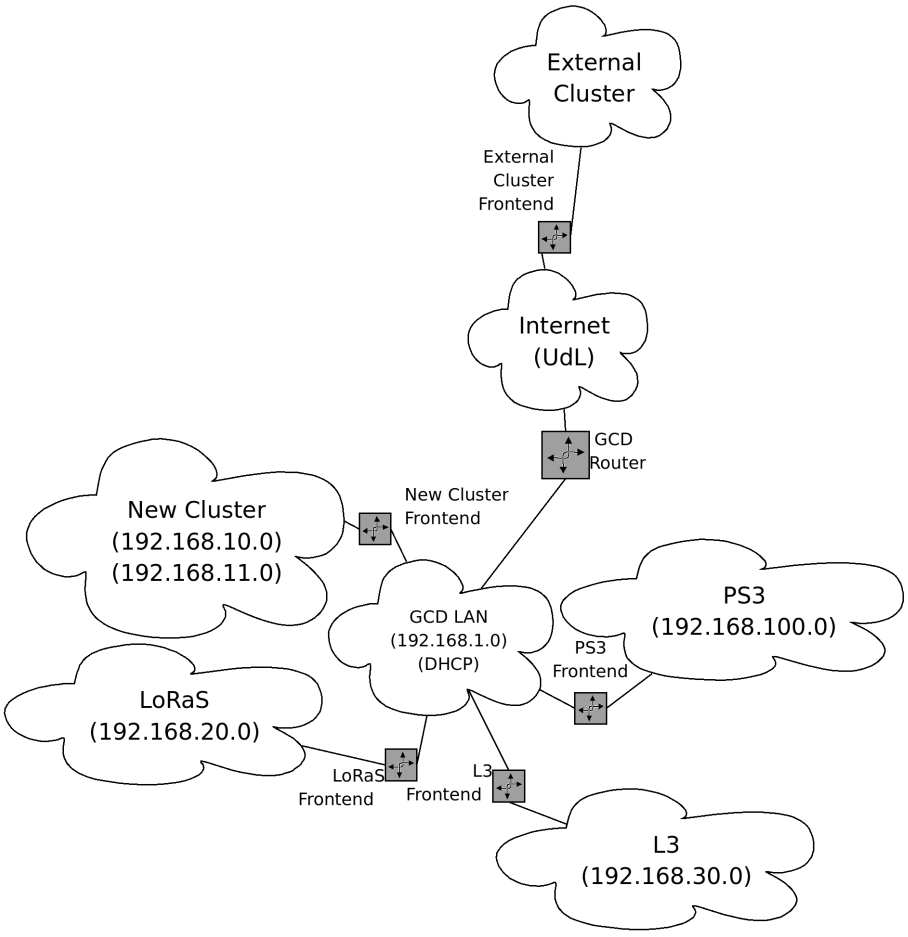



Figura 1.3: Esquema del multicluster del grupo de computación distribuida de la UdL.

### 1.1.3. Multicluster

Un multicluster es un conjunto de clusters interconectados por una red dedicada que pueden o no compartir aplicaciones de cómputo entre ellos para obtener una mejor eficiencia. Un ejemplo puede multicluster puede ser el multi cluster montado por el grupo de computación distribuida de la Universitat de Lleida. Este cluster está representado en la figura 1.3. Otro ejemplo puede ser considerar la diferenciación de los clusters no en cuánto a qué red se encuentran sino en el conjunto de tareas que se le asigna a cada nodo. Así pues aunque un conjunto de nodos esté en una misma red y con un mismo servidor pueden diferenciarse subconjuntos de nodos con tareas diferentes. Podemos ver un ejemplo de esta otra forma de multicluster en la figura 1.2.

## 1.2. Contexto y Motivación

El grupo de investigación: Grupo de Computación Distribuida de la Universitat de LLeida se propone a llevar a cabo el desarrollo e implementación de la tecnología de cómputo distribuido peer-to-peer en clusters. Una de las labores indispensables para llevar a cabo esa investigación es automatizar la instalación de las máquinas que conformarán esos clusters. El sistema actual basado en Rocks tiene varias deficiencias como no ser lo suficiente flexible o usar un kernel de Linux muy antiguo. Además al no estar basado en Debian no se puede aprovechar al máximo toda la flexibilidad y potencia del sistema de empaquetado de Debian.

## 1.3. Objetivos

Entre las características de ROCKS que hemos comentado antes nos interesa conseguir las siguientes:

- Instalación sistemática y masiva del sistema operativo de los nodos.
  - De una sola distribución.
  - De varias distribuciones.
- Sincronización en la actualización de los nodos por reinstalación.

## 1.4. Contenido de la memoria

El contenido de esta memoria es el siguiente:

**Capítulo** 2: Este capítulo intenta dar una visión detallada de la arquitectura y funcionamiento de Desdeslin. Así mismo servirá de guía para futuros desarrolladores que deseen ampliar funcionalidades del sistema.

**Capítulo** 3: Es el manual de usuario final de la aplicación. Explicará la instalación, configuración y ejecución del sistema Desdeslin.

**Capítulo** 4: En este capítulo podemos ver las modificaciones que se necesitaron aplicar a los paquetes oficiales de Ubuntu para que pudieran servir una instalación automatizada de Ubuntu 8.10 sin ningún problema.

**Capítulo** 5: Este capítulo expone las conclusiones de este trabajo y las posibles mejoras que se le podrían aplicar al sistema para darle continuidad a la aplicación en el proyecto de cómputo distribuido peer-to-peer.

**Apéndice** A: Muestra la estructura y el contenido del CDROM adjunto al trabajo.

**Apéndice** B: Podemos ver el código fuente de los scripts y ficheros de configuración que proporciona el proyecto.

**Apéndice** C: Licencia de documentación libre de GNU.

# Capítulo 2

# Desdeslin

En este capítulo veremos Desdeslin. Empezaremos con la arquitectura del sistema dónde se describen cada uno de los demonios que ese ejecutan tanto en el servidor como en los nodos para poder llevar a cabo la instalación automatizada. A continuación veremos cómo funciona esa instalación. Se describirá la interacción entre los diferentes demonios y los resultados de dichas interacciones para configurar el inicio de la instalación automatizada. Finalmente se detallará cada uno de los pasos de la instalación automatizada propiamente dicha. Desdeslin será la base de un futuro cluster basado en Ubuntu 8.10 pensado para reemplazar las funciones del sistema operativo Rocks especializado en clusters.

## 2.1. Arquitectura del Sistema

El sistema tiene una arquitectura basada en el modelo servidor-cliente. Así pues el servidor ofrecerá unos servicios al cliente que le servirán a este último para instalarse automáticamente. En primer lugar el cliente necesita arrancar, para ello usa la BIOS como cliente de dhcp (o de arranque). En la parte del servidor encontramos el servidor de dhcp que proporcionará los archivos necesarios para el arranque. Se necesitará un cliente y un demonio en el servidor para obtener por el protocolo tftp varios archivos esenciales. También necesitaremos servir desde el servidor y montar desde el cliente un sistema de ficheros por red como nfs para iniciar los scripts de instalación automatizada. Finalmente necesitaremos un servidor de repositorios para servir paquetes. El cliente tendrá a su vez un módulo de software capaz de recibir e instalar esos paquetes. La arquitectura del sistema se sintetiza en la figura 2.1.

### 2.1.1. Servidor

Como su nombre indica el servidor servirá a los nodos de instalación la información necesaria para llevar a cabo con éxito las instalaciones.

El servidor está compuesto por un demonio dhcp, un demonio del protocolo tftp, un demonio del protocolo nfs y finalmente un repositorio de paquetes Debian.

El servidor de dhcp proporcionará los archivos necesarios para el arranque de los nodos. El demonio tftp proporcionará varios archivos esenciales mediante

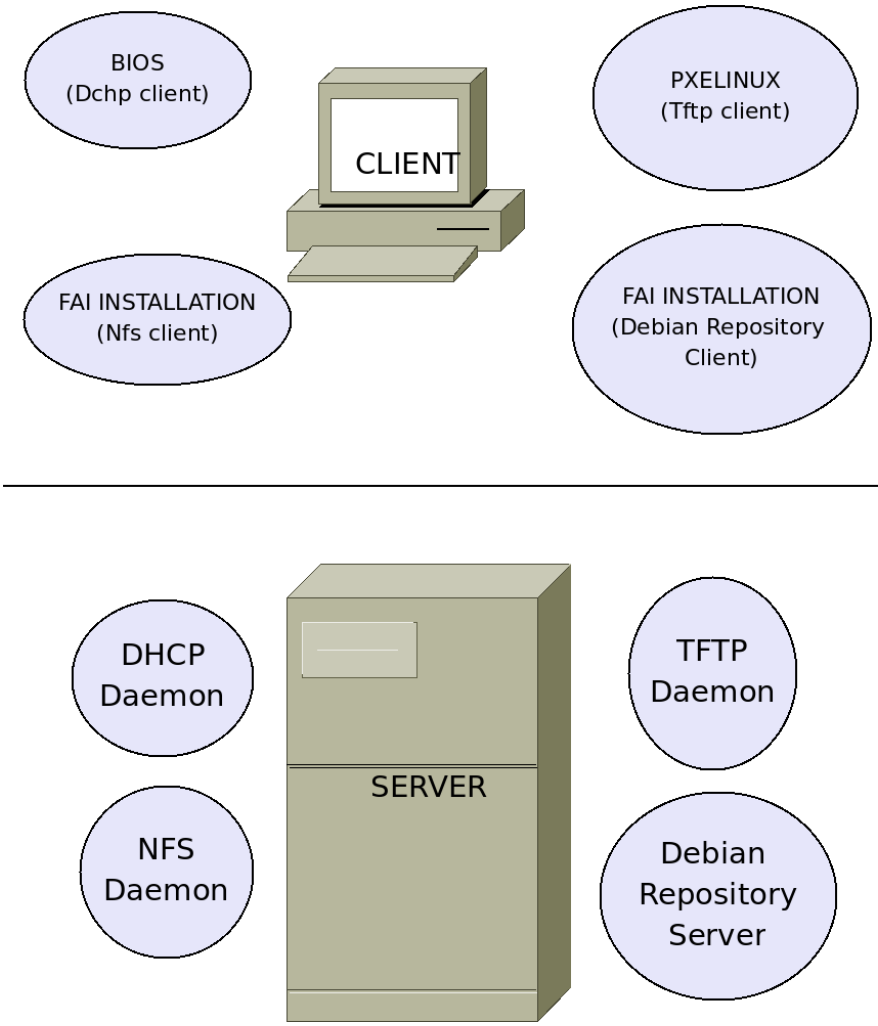



Figura 2.1: Arquitectura de Desdeslin

el protocolo tftp al nodo. También necesitaremos servir un sistema de ficheros por red nfs con los scripts de instalación automatizada. Finalmente un servidor de repositorios podrá servir los paquetes de los programas que el nodo necesite.

A continuación se explica cada uno de los componentes del servidor por separado.

#### 2.1.1.1. Dhcp Daemon

El demonio de dhcp se encarga de proporcionar a cada uno de los nodos un nombre de host y una dirección ip. Estos datos se asignan previamente en un fichero de configuración que los identifica a la MAC del nodo. La asignación bien puede ser dinámica (sin depender de la MAC) o estática. Para el caso de los clusters es mejor estática para poder estructurar de antemano los posibles clusters.

#### 2.1.1.2. Tftp Daemon

El demonio tftp se encarga de proporcionar acceso a los ficheros de arranque. Primero sirve el cargador de arranque pxelinux y después sirve el kernel y el initrd al cargador de arranque para que los pueda ejecutar.

#### 2.1.1.3. Nfs Daemon

El demonio nfs se encarga de proporcionar acceso a varios sistemas de ficheros montados por red mediante el protocolo nfs. Estas rutas serán empleadas por FAI para realizar iniciar su instalación automática. Además a partir de esas rutas se especificarán las características de cada nodo.

#### 2.1.1.4. Debian Repository Server

Una parte importante del proceso de instalación de FAI consiste en la instalación de programas. Los programas se encuentran almacenados en forma de paquetes en el servidor. Para que los nodos tengan acceso a esos paquetes se debe configurar un repositorio en el servidor normalmente servido por el protocolo http.

### 2.1.2. Cliente

#### 2.1.2.1. Bios (Dhcpd Client)

La BIOS del nodo se configura para arranque por red PXE. Gracias a esto manda un paquete especial broadcast de petición de arranque. La BIOS ejecutará el cargador de arranque por red después de obtenerlo mediante el protocolo dhcp.

#### 2.1.2.2. PxeLinux (Tftp Client)

Pxelinux es el cargador de arranque. Este hace uso del protocolo tftp para obtener los ficheros que necesita. En primer lugar pedirá su fichero de configuración. A continuación pedirá el kernel y el initrd especificados para cargar en el fichero de configuración.

#### 2.1.2.3. FAI Installation (Nfs Client)

La instalación automática de FAI será cliente de nfs. Con el montaje de nfs se podrán hacer uso de los scripts de instalación automática. También se podrá obtener la configuración particular de cada nodo. Por último de forma opcional también se puede tener acceso a un repositorio de paquetes Debian.

#### 2.1.2.4. FAI Installation (Debian Repository Client)

La instalación automática de FAI será cliente del repositorio de Debian. Esto permitirá poder instalar los paquetes escogidos para cada uno de los nodos con las ventajas del sistema de paquetería de Debian. Por ejemplo, podremos instalar un paquete y automáticamente se seleccionarán los paquetes en los que depende ese paquete para poder funcionar.

## 2.2. Funcionamiento del sistema de instalación automática

La instalación automática tiene dos procesos diferenciados. El primero es el arranque y configuración de un sistema Linux preparado para contar con todos los requisitos (paquetes, scripts de instalación, etc.) que necesita la instalación automatizada de FAI. El segundo proceso consiste en la instalación automatizada del sistema requerido en el nodo dónde ya sí se interpreta la configuración a aplicar y se procede a la instalación de todos los paquetes de software especificados.

### 2.2.1. Arranque de la instalación automatizada.

En la figura 2.2 podemos ver el esquema de arranque de un nodo para su instalación automática. Como podemos ver desde el cliente se van pidiendo desde imágenes de arranque, kernel, initrd hasta el montaje de un sistema de ficheros nfs con toda la información necesaria para poder iniciar la instalación. Cada uno de los pasos de detallarán a continuación.

1. El cargador de arranque se carga en RAM.
   La BIOS realiza una petición de broadcast de DHCP. El servidor de DHCP responde con la dirección IP del nodo, su mascara de red, su nombre de host y el fichero a cargar en RAM. Todos estos datos son servidos por el servidor dhcp en base a las MAC del nodo. El fichero a cargar en RAM será el cargador de arranque Pxelinux. La BIOS le pasará el control del procesador una vez lo haya cargado totalmente en la RAM.

2. El cargador de arranque lee su configuración.
   El cargador de arranque realiza varias peticiones tftp para conseguir su fichero de configuración. El nombre del mismo se calcula a partir de la dirección MAC del nodo. En el fichero de configuración se especifica el kernel a cargar, el disco RAM de inicio de apoyo para el kernel (initrd) y los parámetros de arranque del kernel.

3. El cargador de arranque carga el Kernel e Initrd.
   El cargador de arranque lee su fichero de configuración. Este especifica que

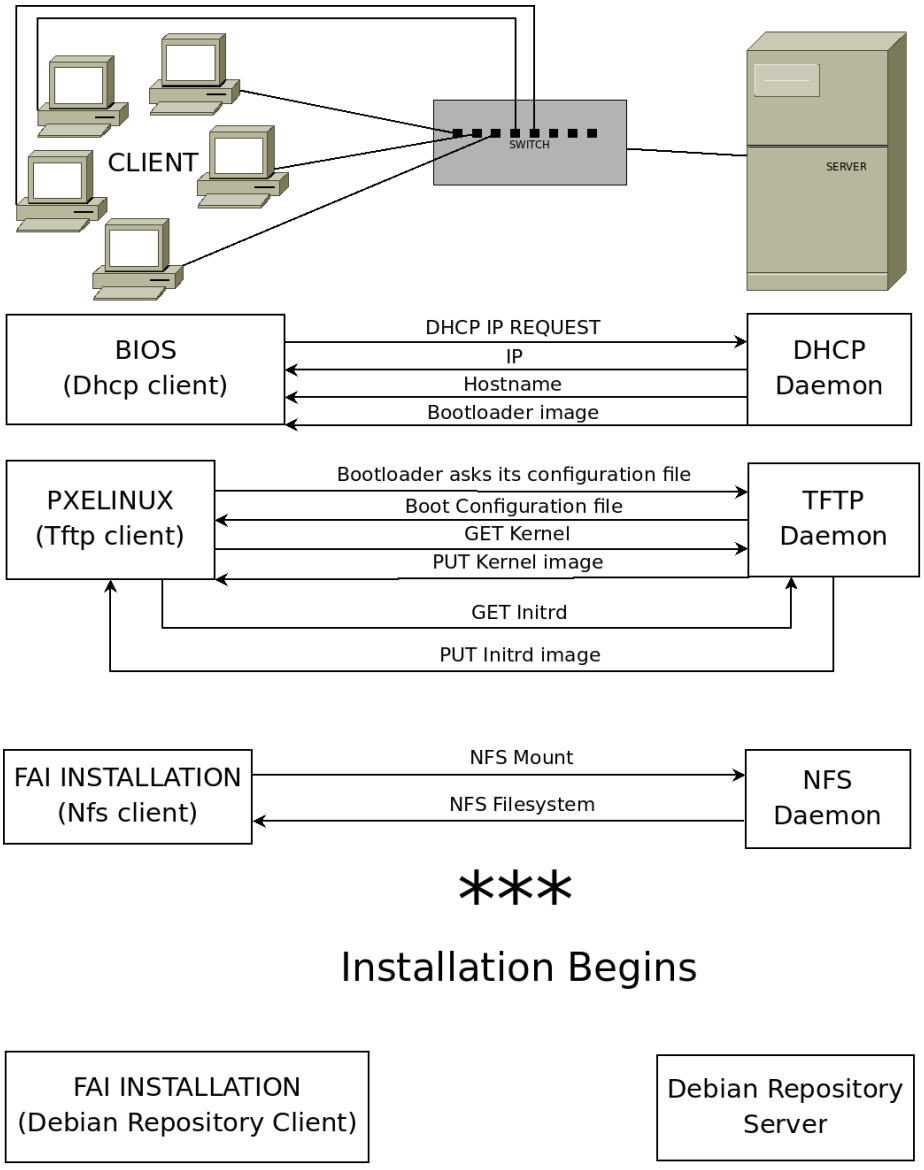



Figura 2.2: Esquema del arranque de un nodo para instalación automatizada

deben cargarse en RAM el kernel y el initrd así como que al kernel ha de pasarse el control del procesador. La ruta tanto del kernel como del initrd están en el servidor. Así pues el cargador de arranque hará dos peticiones tftp al servidor de tftp quien servirá ambos archivos. Una vez obtenidos y cargados en RAM el proceso de arranque seguirá con la ejecución del Kernel.

4. Montaje de la instalación mínima.
   En este punto el kernel interpreta los scripts del disco RAM de inicio de apoyo (initrd). Estos especifican que se ha de montar por nfs un sistema de ficheros. Este sistema de ficheros contendrá el verdadero sistema de instalación.

5. La instalación empieza.
   Una vez el sistema proporcionado por nfs se configura y carga correctamente se inicia la instalación automática.

### 2.2.2. Instalación automática

La instalación automática empieza con la ejecución del script /usr/sbin/fai, la definición de clases para poder diferenciar las particularidades del nodo. El proceso sigue con el particionamiento de los discos duros locales y su formateo para poder seguidamente instalar los paquetes software del nodo. Esta instalación podrá ser completada gracias a la llamada de unos scripts específicos. Por último después de guardar los ficheros de registro podremos reiniciar la máquina para poder arrancarla directamente. Todos estos pasos de la instalación automática serán descritos con mayor detalle a continuación.

1. Configuración inicial de FAI.
   Una vez el nodo de instalación ha arrancado sólo el script /usr/sbin/fai se ejecuta. Este controlará el resto de pasos de la instalación. La configuración de fai se monta via nfs en el directorio /fai. Se crean los terminales virtuales y si se ha pedido se arranca un demonio de shell segura.

2. Definición de clases.
   Se ejecuta el fichero fai-class que define las clases. Esto permite diferenciar la instalación en el nodo dependiendo de su nombre de host así como permite diferenciar otros pasos de la instalación. Podemos hacernos una idea del algoritmo usado en este punto con el diagrama de flujo de la figura 2.3.

3. Particionamiento de discos locales.
   A partir de un sólo fichero de configuración de disco localizado en */fai/disk_ config* (usando la tecnología de clases) se particionan los discos locales. Además cada partición será formateada con un sistema de ficheros. En este mismo fichero también se especifica en que ruta del sistema de ficheros raíz deben montarse los sistemas de ficheros definidos.

4. Instalación de paquetes software.
   Una vez creados los sistemas de ficheros todos están vacíos. Para instalar

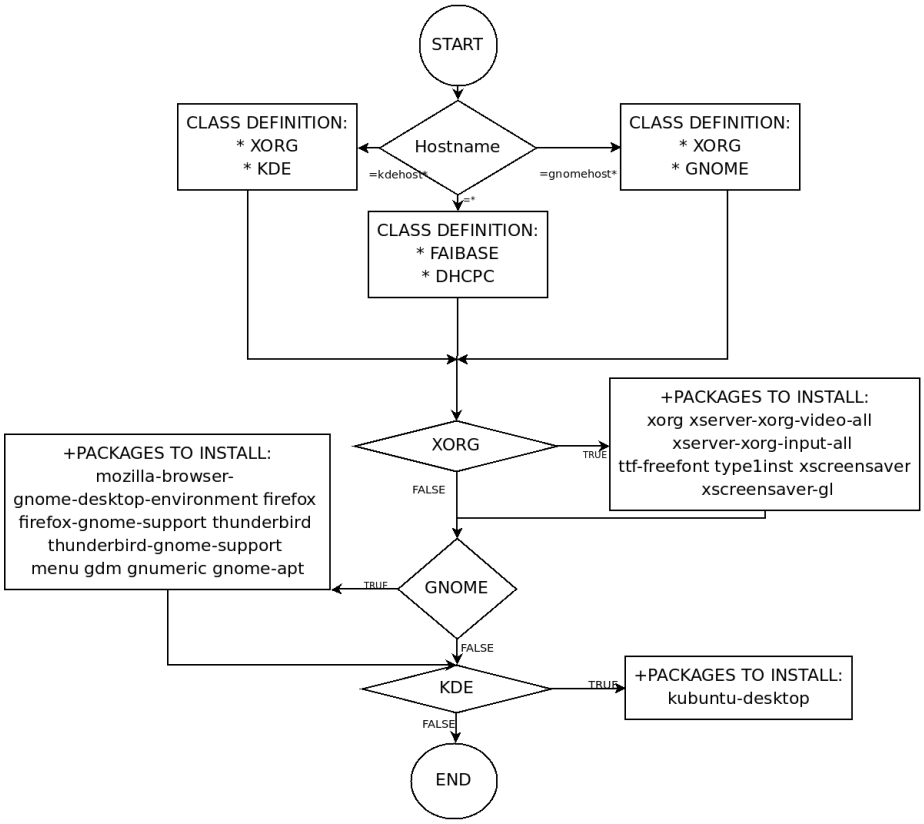



Figura 2.3: Diagrama de flujo del tratamiento de clases en la instalación de FAI

el sistema básico de la distribución FAI hace uso del comando debootstrap. Ese sistema básico contiene el sistema mínimo de soporte para la gestión de paquetes. Finalmente los paquetes especificados en los ficheros de */fai/package_config* se instalan.

5. Llamadas a scripts específicos.
   Una vez instalados los paquetes se puede afinar aún más la configuración llamando a scripts. Estos scripts serán específicos de las clases ya definidas.

6. Salvaguarda de ficheros de registro.
   Cuando las tareas de instalación se terminan los ficheros de registro se escriben en: */var/log/fai/$HOSTNAME/install* en el nuevo sistema. Si definimos la variable *$LOGUSER* en */etc/fai/fai.conf* también se guardará en esa cuenta en el servidor.

7. Reinicio del nuevo sistema instalado.
   Por último el nodo instalado se reinicia siempre y cuando *reboot* fuera añadido a la variable *FAI_FLAGS*. Para evitar otra instalación por red podemos cambiar el valor de *FAI_FLAGS* gracias al comando *fai-chboot*.

# Capítulo 3

# Manual de Usuario

El manual de usuario de Desdeslin comprende los requisitos previos (3.1 Requisitos previos) dónde veremos que se necesita para ponerlo en marcha. Entre estos requisitos esta la distribución Ubuntu 8.10 y su configuración para que use los paquetes de Desdeslin en lugar de los oficiales. También veremos cuál es su uso normal una vez configurado (3.2 Operativa diaria de FAI) que consiste básicamente en ordenar instalaciones automáticas o dejar que el nodo arranque directamente (cuando ya esté instalado correctamente). Otro aspecto importante que abordaremos es cómo adecuarlo a nuestras necesidades(3.3 Personalización de FAI). Como complemento al manual veremos qué paquetes modificados se proporcionan (3.5 Paquetes modificados). No menos útil será aprender cómo automatizar nuestras tareas con el sistema (3.6 Scripts útiles). Finalmente veremos una relación de todos los ficheros de configuración importantes (3.7 Ficheros de configuración) como una guía más de referencia.

## 3.1. Requisitos previos

Para poder usar Desdeslin de forma cómoda deberemos en primer lugar instalar una distribución Ubuntu 8.10, configurarla para usar los paquetes proporcionados por el proyecto Desdeslin, instalar todos los paquetes relacionados con FAI y adecuar la configuración a nuestro cluster. A continuación tendremos que configurar la red para así poder generar el sistema de ficheros preparado para la instalación automatizada que se servirá por nfs.

Este documento cubrirá los requisitos previos para tres configuraciones distintas, la configuración Laboratorio 1 malgasta recursos buscando los paquetes directamente a internet, la configuración Laboratorio 2 hace uso de un proxy de paquetes para cachear los mismos. Finalmente se propone la configuración Virtualbox como red de pruebas virtual en un solo equipo físico.

### 3.1.1. Configuraciones de requisitos previos

#### 3.1.1.1. Configuración Laboratorio 1

Esta configuración se realizó en un laboratorio de la UdL. Podemos ver un esquema de la misma en la figura 3.1.

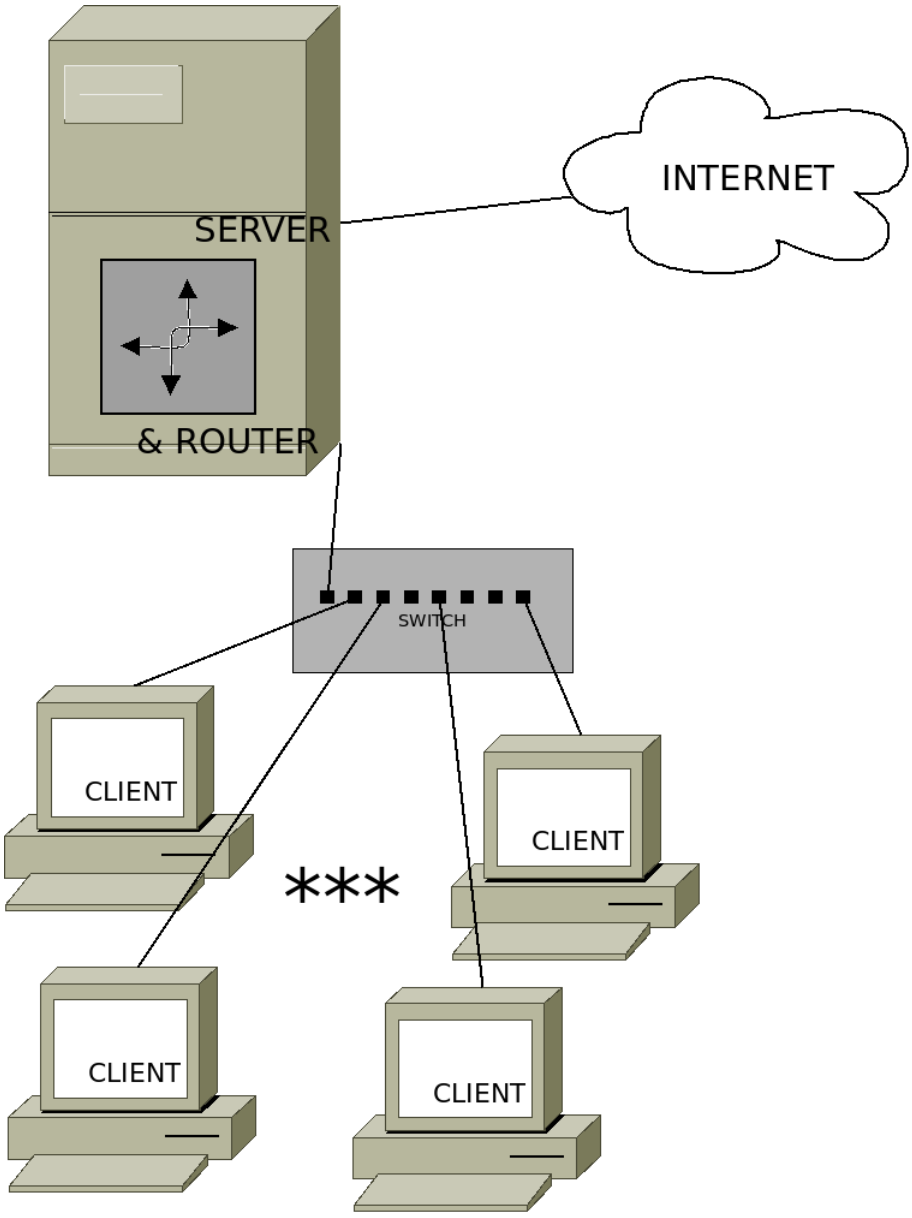



Figura 3.1: Esquema de la configuración de laboratorio 1

La configuración de FAI habitual consiste en poner un mirror de los paquetes de Ubuntu en el servidor o bien montar un proxy de apt para solamente bajarse los paquetes solicitados. En este caso se optó por que los nodos obtuvieran los paquetes de internet. Como el mismo servidor hace de router deberemos configurar sus tablas iptables para que redireccione las peticiones a internet desde la red de los nodos. Los ficheros de configuración de paquetes (sources.list) tanto en el nodo como en el servidor deberán buscar los paquetes oficiales de Ubuntu en internet.Para poder probar Desdeslin sin necesidad de configurar ni un mirror de los paquetes ni un proxy de paquetes es una opción valida.

#### 3.1.1.2. Configuración Laboratorio 2

Esta configuración se realizó en un laboratorio de la UdL. Es una mejora de la configuración laboratorio 1 (3.1.1.1). Esta configuración se encuentra esquematizada en la figura 3.2.

Usaremos un proxy de paquetes. Esto nos permitirá que los paquetes sean obtenidos por los nodos directamente del servidor. Los paquetes serán descargados de internet sólo una vez. Así mismo no necesitaremos que el servidor haga de router. Podremos ver esta idea mejor en el esquema lógico de la configuración que se encuentra en la figura 3.3.

Por lo tanto no deberemos alterar las reglas de iptable en el servidor. En este caso deberemos adecuar los ficheros de configuración de los paquetes (sources.list) tanto del servidor como de los nodos para que busquen los paquetes oficiales en el puerto 9999 del servidor que es dónde está configurado el apt-proxy.

#### 3.1.1.3. Configuración Virtualbox

Esta configuración se realizó en un ordenador offline. El servidor alberga gracias a Virtualbox una máquina virtual. Esta máquina virtual está configurada como un nodo del cluster conectada a la red del servidor. Podemos ver reflejada la configuración Virtualbox en la figura 3.4.

Esto es posible gracias a un bridge. Además el repositorio de Ubuntu se alberga entero en el servidor, así pues, los ficheros de configuración de los paquetes (sources.list) tanto del servidor como de los nodos tendrán que ser modificados para buscar en el servidor los paquetes oficiales de Ubuntu.

### 3.1.2. Requisitos previos para Configuración Laboratorio 1

#### 3.1.2.1. Distribución Ubuntu 8.10

La versión de FAI propuesta por el proyecto Desdeslin funciona en Ubuntu 8.10. Funcionó desde una instalación básica de Ubuntu 8.10 Server Edition. Si se usan otras versiones de FAI se puede elegir otra distribución basada en Debian.

#### 3.1.2.2. Configuración de paquetes propios

Al instalar la distribución ésta tendrá configuradas las fuentes de paquetes (repositorios) por defecto. Primero copiaremos el repositorio propuesto de Desdeslin en un directorio del sistema como pudiera ser */home/ubuntu_ mirrors/*

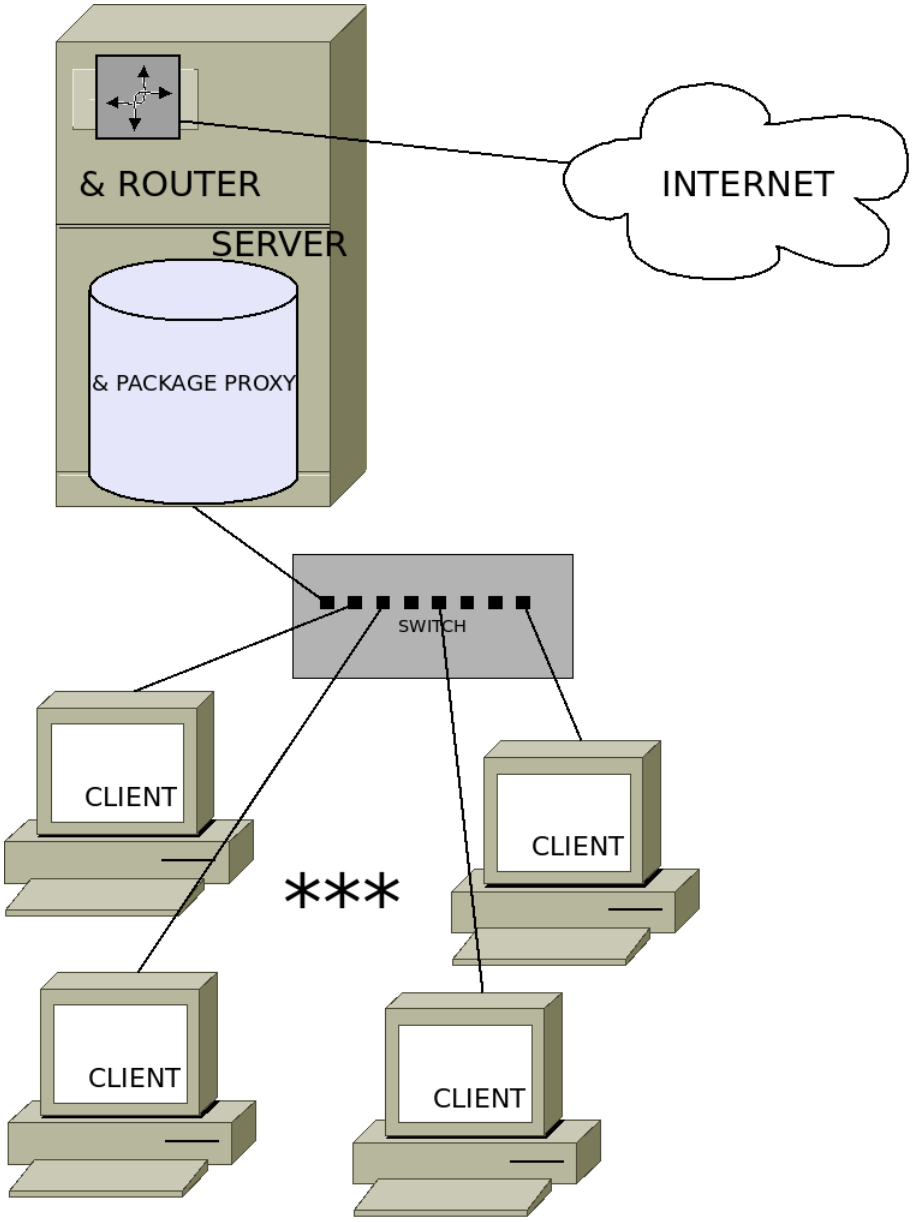



Figura 3.2: Esquema físico de la configuración de laboratorio 2

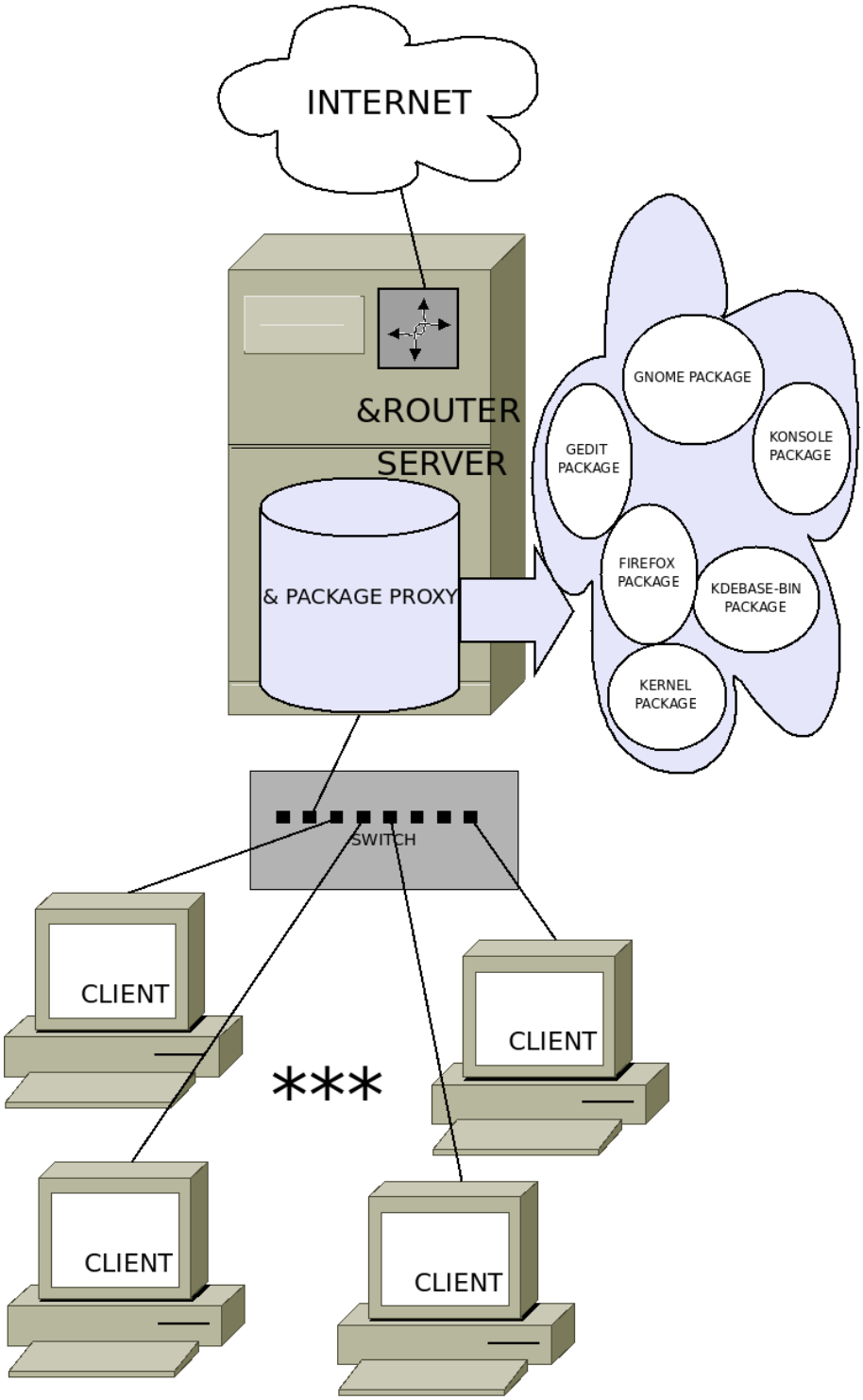



Figura 3.3: Esquema lógico de la configuración de laboratorio 2

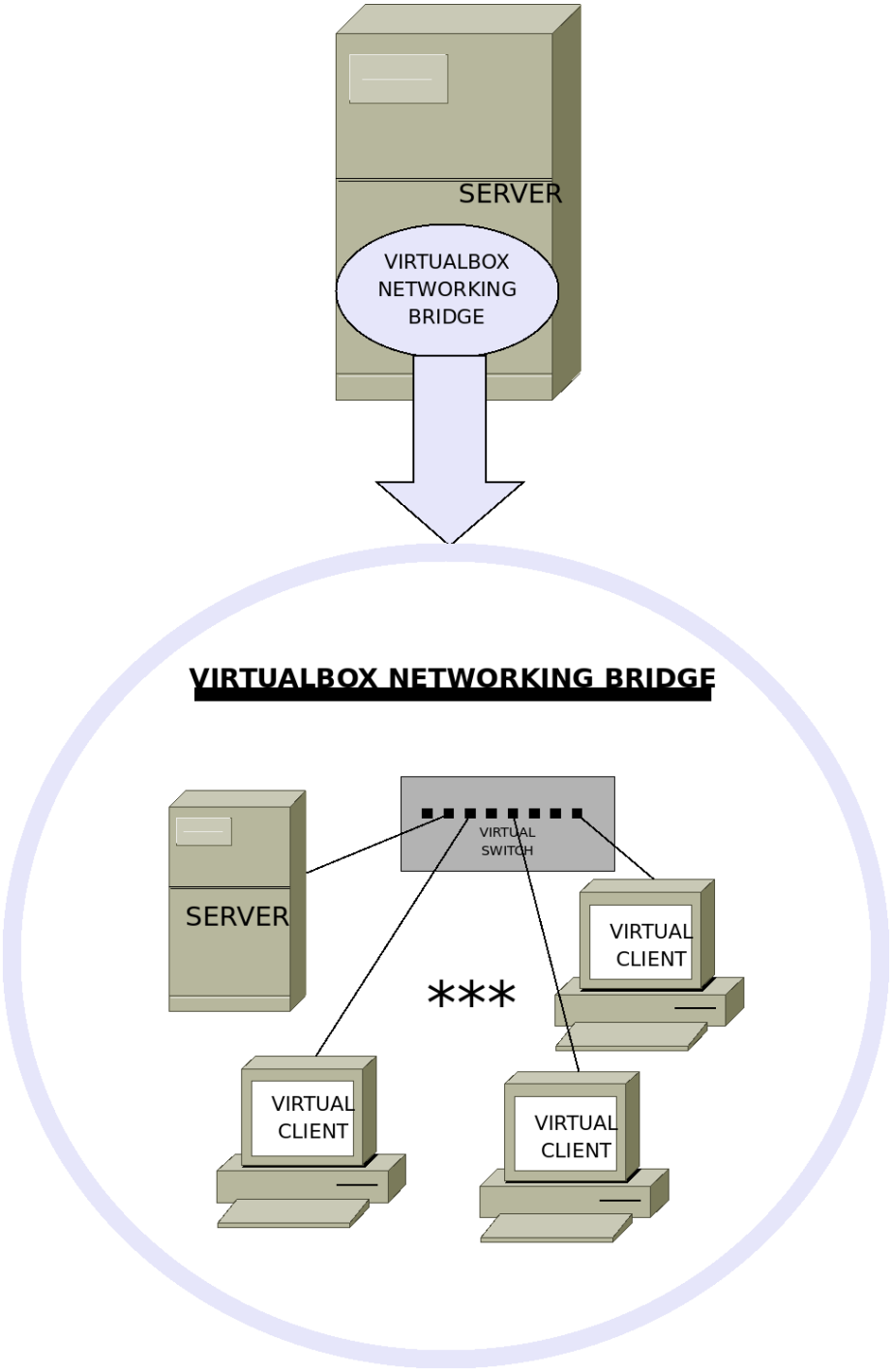



Figura 3.4: Esquema lógico de la configuración Virtualbox

y editaremos el fichero */etc/apt/sources.list* para que este nuevo repositorio se tenga en cuenta en primer lugar. El fichero propuesto por Desdeslin contiene:

```
deb file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
deb-src file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
```

Si suponemos que el sources.list propuesto se encuentra en: */media/disk/configuracion_paquetes_propios/etc/sources.list* para actualizar el mismo podemos hacer:

```
cp /etc/apt/sources.list /etc/apt/sources.list.original
cat /media/disk/configuracion_paquetes_propios/etc/sources.list
/etc/apt/sources.list.original >/etc/apt/sources.list
```

#### 3.1.2.3. Configuración de la preferencia de los paquetes

Al tener en paralelo las versiones oficiales de los paquetes que hemos modificado y los paquetes modificados el sistema ha de decidir qué paquetes ha de instalar. Los paquetes no oficiales han sido definidos como pertenecientes a la sección *codip2p* (en lugar de main, universe o multiverse). Gracias al fichero */etc/apt/preferences* propuesto que se muestra a continuación forzaremos que el sistema prefiera los paquetes de la sección *codip2p* y por tanto los modificados.

```
Package: *
Pin: release c=codip2p
Pin-Priority: 1001
Package: *
Pin: release c=main restricted universe multiverse
Pin-Priority: 100
```

Así pues copiamos el fichero propuesto y actualizamos el sistema de paquetería con la nueva configuración:

```
cp /media/disk/configuracion_preferencia_paquetes/etc/apt/preferences
/etc/apt/preferences
apt-get update
```

#### 3.1.2.4. Instalación de FAI y otros paquetes

Una vez configurados los repositorios y el sistema de instalación de los paquetes procederemos a instalar los paquetes necesarios para instalación correcta de FAI.

- Demonio de dhcp
  ```
  apt-get install dhcp3-server
  ```
- Sistema básico de FAI
  ```
  apt-get install fai-quickstart
  ```

- Paquetes sugeridos por FAI

  *apt-get install debmirror mknbi apt-move mkisofs grub aptitude*

- Actualización de initramfs-tools con nuestro paquete modificado

  *apt-get install initramfs-tools*

#### 3.1.2.5. Configuración de FAI para proyecto codip2p

El paquete FAI adaptado para Ubuntu 8.10 no contempla todas las particularidades del proyecto codip2p.

A continuación se detallan todas las configuraciones y binarios específicos para codip2p.

- Fichero: */etc/fai/fai.conf*: Se añade la linea:

  *FAI_DEBMIRROR=desdeslinserver:/srv/ubuntumirror*

  para poder acceder al repositorio de paquetes modificados via nfs desde los nodos (se monta en */media/mirror*).

- Fichero: */etc/fai/apt/sources.list*: Se combinan los dos repositorios desde el punto de vista del nodo:

```
# Repositorio con nuestros paquetes modificados
# El servidor sirve: /srv/ubuntumirror
# El nodo lo monta en: /media/mirror
deb file:/media/mirror/codip2p/ intrepid codip2p
deb-src file:/media/mirror/codip2p/ intrepid codip2p
# Repositorios de Ubuntu de España para las secciones
habituales.
deb http://es.archive.ubuntu.com/ubuntu/ intrepid main restricted
deb-src http://es.archive.ubuntu.com/ubuntu/ intrepid main restricted
deb http://es.archive.ubuntu.com/ubuntu/ intrepid-updates main restricted
deb-src http://es.archive.ubuntu.com/ubuntu/ intrepid-updates main restricted
deb http://es.archive.ubuntu.com/ubuntu/ intrepid universe
deb-src http://es.archive.ubuntu.com/ubuntu/ intrepid universe
deb http://es.archive.ubuntu.com/ubuntu/ intrepid-updates universe
deb-src http://es.archive.ubuntu.com/ubuntu/ intrepid-updates universe
deb http://es.archive.ubuntu.com/ubuntu/ intrepid multiverse
deb-src http://es.archive.ubuntu.com/ubuntu/ intrepid multiverse
deb http://es.archive.ubuntu.com/ubuntu/ intrepid-updates multiverse
deb-src http://es.archive.ubuntu.com/ubuntu/ intrepid-updates multiverse
deb http://security.ubuntu.com/ubuntu intrepid-security main restricted
deb-src http://security.ubuntu.com/ubuntu intrepid-security main restricted
deb http://security.ubuntu.com/ubuntu intrepid-security universe
deb-src http://security.ubuntu.com/ubuntu intrepid-security universe
deb http://security.ubuntu.com/ubuntu intrepid-security multiverse
deb-src http://security.ubuntu.com/ubuntu intrepid-security multiverse
```

- Fichero: */srv/fai/config/scripts/last/srv/50-misc*: Se descomenta la linea que monta el repositorio Debian via nfs.

```
[ "$FAI_DEBMIRROR" ] && echo "$FAI_DEBMIRROR $MNTPOINT nfs ro 0 0" >>$target/etc/fstab
```

Si tenemos los ficheros propuestos por Desdeslin todos estos pasos se pueden ahorrar haciendo una copia de los ficheros proporcionados tal que:

```
cp -r /media/disk/configuracion_fai_codip2p/etc /media/disk/configuracion_fai/codip2p/srv /
```

#### 3.1.2.6. Configuración del sistema restante

Aunque no es especifico de FAI hay algunas configuraciones que han de ser establecidas en el sistema para que este funcione. A continuación se detallan todas las configuraciones especificas para este propósito.

- Fichero: */etc/dhcp3/dhcpd.conf*: A partir de la MAC de cada nodo se define su ip y su nombre de host.

```
ddns-update-style ad-hoc;
# Definimos la mascara de red de una clase C
option subnet-mask 255.255.255.0;
option option-128 code 128 = string;
option option-129 code 129 = text;
get-lease-hostnames true;
group {
  # Activamos que los nodos reconozcan como router al
propio servidor para tener acceso al exterior.
  option routers 192.168.1.4;
   # Activamos el envío de nombres (hostname) además de
ips.
    use-host-decl-names on;
  # Definimos la subred 192.168.1.0 con máscara de red
255.255.255.0.
  subnet 192.168.1.0 netmask 255.255.255.0 {
    # Definimos el nodo con nombre: demohost
    host demohost {
      # Con su MAC lo identificamos.
       hardware ethernet 00:07:E9:94:B9:27; # CLIENTE #1 Laboratorio
      # Y le asignamos una dirección fija.
       fixed-address 192.168.1.5;
           # Para finalmente servirle la imagen de arranque:
/fai/pxelinux.0
             # Esta imagen contiene el cargador de arranque:
Pxelinux.
       filename "/fai/pxelinux.0";
    }
  }
}
```

- Fichero: */etc/exports*: Exportamos /srv/ubuntumirror para poder ser accesible desde la instalación de los nodos.

```
/srv/fai/config\
                    192.168.1.4/255.255.255.0(async,ro,no_subtree_check)
/srv/fai/nfsroot\
          192.168.1.4/255.255.255.0(async,ro,no_subtree_check,no_root_squash)
/srv/ubuntumirror        192.168.1.4/255.255.255.0(async,ro,no_subtree_check)
```

- Fichero: */etc/hosts*: Definimos el nombre del servidor así como todos los nombres de los nodos asignándoles una ip.

```
# Ip para la red de bucle local
127.0.0.1 localhost
# Ip del servidor al cual llamaremos: desdeslinserver
192.168.1.4 desdeslinserver
# Ip del nodo de demostración que llamaremos: demohost
192.168.1.5 demohost
```

- Se define el arranque del demonio tftp.

```
# Definimos que el superdemonio inet controle al programa
tftp que funciona con el protocolo udp.
# Para ello este se ejecutará como root y servirá el
contenido del directorio /srv/tftp
# En /srv/tftp se encuentra la imagen del cargador de
arranque, su configuración, el kernel y el initrd.
tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s
/srv/tftp
```

- Creación del enlace simbólico en /srv/ubuntumirror para simplificar las rutas en la configuración.

```
cd /srv
ln -s ../home/ubuntu_mirrors/ ubuntumirror
```

- Reinicio del demonio de dhcp

```
/etc/init.d/dhcp3-server restart
```

- Reinicio del demonio inetd

```
/etc/init.d/openbsd-inetd restart
```

Como en otros pasos de la configuración previa podemos obviar la edición de ficheros antes explicada usando los ficheros propuestos por Desdeslin:

```
cp -r /media/disk/configuracion_sistema_restante/etc /
```

#### 3.1.2.7. Configuración de la red

En este paso hemos de configurar la red para poder terminar de configurar FAI.

Si configuramos los nodos para que los paquetes los consigan de internet deberemos definir a nuestro servidor como router de nuestros nodos. Para ello deberemos definir unas reglas iptable para lo mismo y además deberemos cerciorarnos de que el servidor dhcp sólo funciona en la interfaz de red de nuestra red interna y no en la de acceso a internet.

A continuación se detallan los ficheros a modificar y los comandos a ejecutar para llevar a cabo esta configuración.

- Fichero /etc/default/*dhcp3-server: Se configura para que sólo escuche por la interfaz de la red interna.*

```
INTERFACES="eth0"
```

- Fichero */etc/init.d/iptables_codip2p*: Fichero que contendrá las reglas iptables a ejecutar al inicio. Está tomado prestado de [2].

```
# Para poder portar fácilmente el script a otras
distribuciones de Linux
# definimos las rutas absolutas de los comandos utilizados
#
# Usaremos el programa iptables para modificar las tablas
ip del Kernel
IPTABLES=/sbin/iptables
#Con awk podremos extraer información de la salida de
comandos
AWK=/usr/bin/awk
#Gracias a ifconfig podremos obtener información de las
interfaces de redes
IFCONFIG=/sbin/ifconfig
# Definición de la interfaz externa (Hacia internet)
EXTIF="eth1"
# Dirección Ip Externa (Detectada automáticamente)
EXTIP="`$IFCONFIG $EXTIF | $AWK /$EX-
TIF/'{next}//{split($0,a,":");split(a[2],a," ");print a[1];exit}'`"
# Definicion de la interfaz interna (Red local del cluster)
INTIF="eth0"
# Dirección Ip interna (Notación CIDR)
# Como trabajamos con una clase destinamos 24 bits para
definir la red.
# Para el servidor que será también el router hemos elegido
# la dirección 192.168.1.4
INTIP="192.168.1.4/24"
# Dirección de red interna (Notación CIDR)
INTNET="192.168.1.0/24"
# Dirección por defecto (Notación CIDR)
UNIVERSE="0.0.0.0/0"
# Informamos al usuario de que configuraciones de red se
han detectado
#
#Mostraremos la interfaz de red externa y su ip.
echo "External: [Interface=$EXTIF] [IP=$EXTIP]"
# Tambien mostraremos la interfaz de red interna y su ip.
Además veremos su dirección de red.
echo "Internal: [Interface=$INTIF] [IP=$INTIP] [Net-
work:$INTNET]"
echo
echo -n "Cargando reglas..."
# Modificamos el sistema de ficheros proc para activar el
forwarding de IP
echo 1 >/proc/sys/net/ipv4/ip_forward
# Las reglas de filtrado INPUT, OUTPUT, y FORWARD
# activan su política a DROP (descartar paquetes)
# Así mismo se borran las reglas ya existentes.
$IPTABLES -P INPUT DROP
$IPTABLES -F INPUT
$IPTABLES -P OUTPUT DROP
$IPTABLES -F OUTPUT
$IPTABLES -P FORWARD DROP
$IPTABLES -F FORWARD
$IPTABLES -F -t nat
# Borrar todas las reglas de los usuarios
$IPTABLES -X
# Finalmente reseteamos los contadores de las iptables.
$IPTABLES -Z
```

```
#######
# INPUT: Reglas para el tráfico de entrada desde varias
interfaces #
#######
# Definimos la interfaz de bucle local como valida
$IPTABLES -A INPUT -i lo -s $UNIVERSE -d $UNIVERSE -j AC-
CEPT
# Tanto la interfaz local, como las maquinas locales pueden
ir a cualquier dirección
$IPTABLES -A INPUT -i $INTIF -s $INTNET -d $UNIVERSE -j
ACCEPT
# Anulamos el IP Spoofing impidiendo que en la interfaz
de red de salida a internet se reciban paquetes con
direcciones de las maquinas locales
$IPTABLES -A INPUT -i $EXTIF -s $INTNET -d $UNIVERSE -j
REJECT
# Autorizamos el tráfico ICMP desde cualquier fuente en la
interfaz externa.
$IPTABLES -A INPUT -i $EXTIF -p ICMP -s $UNIVERSE -d $EX-
TIP -j ACCEPT
# Permitimos que cualquier tráfico relacionado vuelva al
servidor MASQ.
$IPTABLES -A INPUT -i $EXTIF -s $UNIVERSE -d $EXTIP -m
state –state ESTABLISHED,RELATED -j ACCEPT
# Activamos el tráfico DHCP en la interfaz interna.
$IPTABLES -A INPUT -i $INTIF -p tcp –sport 68 –dport 67 -j AC-
CEPT
$IPTABLES -A INPUT -i $INTIF -p udp –sport 68 –dport 67 -j AC-
CEPT
# Autorizamos el tráfico HTTP/HTTPS en la interfaz externa.
$IPTABLES -A INPUT -i $EXTIF -m state –state
NEW,ESTABLISHED,RELATED -p tcp -s $UNIVERSE -d $EX-
TIP –dport 80 -j ACCEPT
$IPTABLES -A INPUT -i $EXTIF -m state –state
NEW,ESTABLISHED,RELATED -p tcp -s $UNIVERSE -d $EX-
TIP –dport 443 -j ACCEPT
# Autorizamos el tráfico SSH en la interfaz externa.
$IPTABLES -A INPUT -i $EXTIF -m state –state
NEW,ESTABLISHED,RELATED -p tcp -s $UNIVERSE -d $EX-
TIP –dport 22 -j ACCEPT
# Cualquier otro paquete para INPUT debe ser rechazado.
$IPTABLES -A INPUT -s $UNIVERSE -d $UNIVERSE -j REJECT
```

```
#######
# OUTPUT: Tráfico que sale de varias interfaces #
#######
# Esto es un truco para evitarnos un error en netfilter
$IPTABLES -A OUTPUT -m state -p icmp –state INVALID -j DROP
# Definimos la interfaz de bucle local como válida.
$IPTABLES -A OUTPUT -o lo -s $UNIVERSE -d $UNIVERSE -j
ACCEPT
# En la interfaces locales cualquier fuente yendo a la red
local es válida.
$IPTABLES -A OUTPUT -o $INTIF -s $EXTIP -d $INTNET -j AC-
CEPT
# En la interfaz local, la fuente del servidor MASQ que va
a la red local es válida.
$IPTABLES -A OUTPUT -o $INTIF -s $INTIP -d $INTNET -j AC-
CEPT
# Se deniega poder ir a la red local a través de la
interfaz externa.
$IPTABLES -A OUTPUT -o $EXTIF -s $UNIVERSE -d $INTNET -j
REJECT
# Cualquier otra cosa que va hacia fuera en la interfaz
externa es valida.
$IPTABLES -A OUTPUT -o $EXTIF -s $EXTIP -d $UNIVERSE -j
ACCEPT
#En la interfaz interna el tráfico DHCP se autoriza.
$IPTABLES -A OUTPUT -o $INTIF -p tcp -s $INTIP –sport 67 -d
255.255.255.255 –dport 68 -j ACCEPT
$IPTABLES -A OUTPUT -o $INTIF -p udp -s $INTIP –sport 67 -d
255.255.255.255 –dport 68 -j ACCEPT
# Toda otra regla se considerada denegada y se registra.
$IPTABLES -A OUTPUT -s $UNIVERSE -d $UNIVERSE -j RE-
JECT
```

```
#######
# Redireccionamiento de Paquetes / NAT #
#######
# Aceptamos paquetes tcp solicitados.
$IPTABLES -A FORWARD -i $EXTIF -o $INTIF -m state –state ES-
TABLISHED,RELATED -j ACCEPT
# Permitir paquetes a través de la interfaz interna.
$IPTABLES -A FORWARD -i $INTIF -o $INTIF -j ACCEPT
# Activamos el redireccionamiento de la red interna a
Internet.
$IPTABLES -A FORWARD -i $INTIF -o $EXTIF -j ACCEPT
# Regla para rechazar todo.
$IPTABLES -A FORWARD -j REJECT
# Enmascaramiento de IP
$IPTABLES -t nat -A POSTROUTING -o $EXTIF -j SNAT –to $EX-
TIP
# Informamos al usuario del final del script.
echo " Hecho."
```

- Fichero */etc/network/interfaces*: Fichero que nos define la función de las interfaces de redes. En nuestro caso la interfaz eth0 será la conectada a la red interior del cluster y, por ejemplo, será definida de forma estática como una clase C. La interfaz eth1 está conectada la red exterior del cluster para poder acceder a internet y otros servicios y en nuestra red se define via dhcp.

```
# La interfaz de bucle local
auto lo
iface lo inet loopback
# La interfaz de red externa
# Se configura por dhcp.
auto eth1
iface eth1 inet dhcp
# La interfaz de red interna (red cluster)
# Definimos una red de clase C
# El servidor tendrá la dirección 192.168.1.4.
auto eth0
iface eth0 inet static
address 192.168.1.4
netmask 255.255.255.0
network 192.168.1.0
broadcast 192.168.1.255
```

- Fichero */etc/resolv.conf*: Se propone un fichero de configuración tipo para la resolución de nombres en la red de la Universitat de Lleida por si la asignación de los servidores dns por dhcp fallará.

```
domain udl.net
search udl.net
nameserver 172.16.20.10
```

- Creación del enlace de inicio de las iptables especificas: El fichero */etc/init.d/iptables_ codip2p* antes propuesto tendrá que ejecutarse a cada

inicio del sistema operativo. Para ello deberemos crear un enlace simbólico en: /etc/rc2.d que apunte al mismo empezando con la letra S mayúscula. Como por ejemplo:

```
cd /etc/rc2.d
ln -s ../init.d/iptables_codip2p S99iptables_codip2p
```

- Reinicio de la red: Reiniciaremos la red para contemple la nueva configuración.

```
/etc/init.d/networking restart
```

- Reinicio del servicio dhcp. Hemos configurado el servicio dhcp para que sólo escuche por una interfaz. Deberemos reiniciarlo para que los cambios surtan efecto.

```
/etc/init.d/dhcp3-server restart
```

- Establecimiento de las reglas iptable. Para que el servidor se comporte como router deberemos activar las reglas iptable que hemos definido previamente.

```
/etc/init.d/iptables_codip2p start
```

#### 3.1.2.8. Instalación del nfsroot de Fai

Con:

```
fai-setup -v
```

configuramos la raíz de que servirá de base a FAI para su instalación automática y que será servida por NFS.

### 3.1.3. Requisitos previos para Configuración Laboratorio 2

#### 3.1.3.1. Distribución Ubuntu 8.10

La versión de FAI propuesta por el proyecto Desdeslin funciona en Ubuntu 8.10. Funcionó desde una instalación básica de Ubuntu 8.10 Server Edition. Si se usan otras versiones de FAI se puede elegir otra distribución basada en Debian.

#### 3.1.3.2. Configuración de paquetes propios

Al instalar la distribución está tendrá configuradas las fuentes de paquetes (repositorios) por defecto. Primero copiaremos el repositorio propuesto de Desdeslin en un directorio del sistema como pudiera ser */home/ubuntu_mirrors/* y editaremos el fichero */etc/apt/sources.list* para que este nuevo repositorio se tenga en cuenta en primer lugar. El fichero propuesto por Desdeslin contiene:

```
deb file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
deb-src file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
```

Si suponemos que el sources.list propuesto se encuentra en: */media/disk/configuracion_paquetes_propios/etc/sources.list* para actualizar el mismo podemos hacer:

```
cp /etc/apt/sources.list /etc/apt/sources.list.original
cat /media/disk/configuracion_paquetes_propios/etc/sources.list
/etc/apt/sources.list.original >/etc/apt/sources.list
```

#### 3.1.3.3. Configuración del proxy de paquetes

Ahora instalaremos el proxy de paquetes para ahorrarnos ancho de banda al bajarnos los paquetes desde los nodos directamente del servidor y no desde internet. En el servidor instalaremos el paquete apt-proxy.

```
apt-get install apt-proxy
```

Hemos de configurar el fichero */etc/apt-proxy/apt-proxy-v2.conf* para los paquetes oficiales de Ubuntu. El fichero propuesto por Desdeslin es el siguiente:

```
[default]
;; Definimos la dirección ip del servidor como la de nuestro
servidor: 192.168.1.4
address = 192.168.1.4
;; De forma arbitraria elegimos el puerto 9999 para el apt-proxy.
port = 9999
min_refresh_delay = 1s
timeout = 15
;; Establecemos el directorio dónde se guardará el caché de
archivos
cache_dir = /var/cache/apt-proxy
;; Frecuencia de limpieza del cache
cleanup_freq = 1d
;; Edad máxima del cache
max_age = 120d
;; Numero máximo de versiones de un mismo paquete
max_versions = 3
[ubuntu]
;; Ruta real al archivo de Ubuntu
backends = http://es.archive.ubuntu.com/ubuntu
[ubuntu-security]
;; Ruta real a las actualizaciones de seguridad de Ubuntu
backends = http://security.ubuntu.com/ubuntu
```

Para que surtan los efectos reiniciaremos el servicio de apt-proxy tal que:

```
/etc/init.d/apt-proxy restart
```

Para aprovecharnos de las ventajas del proxy de paquetes en el servidor actualizaremos la configuración de paquetes (sources.list) en el servidor. De esta manera las referencias que antes eran a internet deberán hacerse al puerto 9999 del servidor. Recordamos que la referencia a nuestros paquetes modificados no debe alterarse. El fichero propuesto por Desdeslin es el siguiente:

```
# Paquetes modificados de Ubuntu 8.10 para Desdeslin
deb file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
deb-src file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
# Distribución intrepid. Secciones main y restricted. Binarios.
# Como el servidor se denomina en el /etc/hosts desdeslinserver
# podemos usar su nombre en el URI de la entrada del repositorio
#
# Advertimos también el uso especifico del puerto 999 dónde
# apt-proxy nos sirve los paquetes cacheados
#
deb http://desdeslinserver:9999/ubuntu/ intrepid main restricted
# Distribución intrepid. Secciones main y restricted. Código
fuente.
deb-src http://desdeslinserver:9999/ubuntu/ intrepid main restricted
## Actualizaciones
deb http://desdeslinserver:9999/ubuntu/ intrepid-updates main restricted
deb-src http://desdeslinserver:9999/ubuntu/ intrepid-updates main restricted
# ...
```

Como siempre después de actualizar la configuración de los paquetes debemos pedir al sistema que lea los nuevos cambios con:

```
apt-get update
```

#### 3.1.3.4. Configuración de la preferencia de los paquetes

Al tener en paralelo las versiones oficiales de los paquetes que hemos modificado y los paquetes modificados el sistema ha de decidir qué paquetes ha de instalar. Los paquetes no oficiales han sido definidos como pertenecientes a la sección *codip2p* (en lugar de main, universe o multiverse). Gracias al fichero */etc/apt/preferences* propuesto que se muestra a continuación forzaremos que el sistema prefiera los paquetes de la sección *codip2p* y por tanto los modificados.

```
Package: *
Pin: release c=codip2p
Pin-Priority: 1001
Package: *
Pin: release c=main restricted universe multiverse
Pin-Priority: 100
```

Así pues copiamos el fichero propuesto y actualizamos el sistema de paqueteria con la nueva configuración:

```
cp /media/disk/configuracion_preferencia_paquetes/etc/apt/preferences
/etc/apt/preferences
apt-get update
```

#### 3.1.3.5. Instalación de FAI y otros paquetes

Una vez configurados los repositorios y el sistema de instalación de los paquetes procederemos a instalar los paquetes necesarios para instalación correcta de FAI.

- Demonio de dhcp
  *apt-get install dhcp3-server*
- Sistema básico de FAI
  *apt-get install fai-quickstart*
- Paquetes sugeridos por FAI
  *apt-get install debmirror mknbi apt-move mkisofs grub aptitude*
- Actualización de initramfs-tools con nuestro paquete modificado
  *apt-get install initramfs-tools*

#### 3.1.3.6. Configuracion de FAI para proyecto codip2p

El paquete FAI adaptado para Ubuntu 8.10 no contempla todas las particularidades del proyecto codip2p.

A continuación se detallan todas las configuraciones y binarios específicos para codip2p.

- Fichero: */etc/fai/fai.conf*: Se añade la linea:
  *FAI_DEBMIRROR=desdeslinserver:/srv/ubuntumirror*
  para poder acceder al repositorio de paquetes modificados via nfs desde los nodos (se monta en */media/mirror*).
- Fichero: */etc/fai/apt/sources.list*: Se combinan los dos repositorios desde el punto de vista del nodo:

```
# Repositorio con nuestros paquetes modificados
# El servidor sirve: /srv/ubuntumirror
# El nodo lo monta en: /media/mirror
deb file:/media/mirror/codip2p/ intrepid codip2p
deb-src file:/media/mirror/codip2p/ intrepid codip2p
# Distribución intrepid. Secciones main y restricted.
Binarios.
# Como el servidor se denomina en el /etc/hosts
desdeslinserver
# podemos usar su nombre en el URI de la entrada del
repositorio
#
# Advertimos también el uso especifico del puerto 999
dónde
# apt-proxy nos sirve los paquetes cacheados
#
deb http://desdeslinserver:9999/ubuntu/ intrepid main restricted
# Distribución intrepid. Secciones main y restricted.
Código fuente.
deb-src http://desdeslinserver:9999/ubuntu/ intrepid main restricted
## Actualizaciones
deb http://desdeslinserver:9999/ubuntu/ intrepid-updates main restric-
ted
deb-src http://desdeslinserver:9999/ubuntu/ intrepid-updates main res-
tricted
# ...
```

- Fichero: */srv/fai/config/scripts/last/srv/50-misc*: Se descomenta la linea que monta el repositorio Debian via nfs.

```
[ "$FAI_DEBMIRROR" ] && echo "$FAI_DEBMIRROR $MNT-
POINT nfs ro 0 0" >>$target/etc/fstab
```

Si tenemos los ficheros propuestos por Desdeslin todos estos pasos se pueden ahorrar haciendo una copia de los ficheros proporcionados tal que:

```
cp -r /media/disk/configuracion_fai_codip2p/etc /me-
dia/disk/configuracion_fai/codip2p/srv /
```

#### 3.1.3.7. Configuración del sistema restante

Aunque no es especifico de FAI hay algunas configuraciones que han de ser establecidas en el sistema para que este funcione. A continuación se detallan todas las configuraciones especificas para este propósito.

- Fichero: */etc/dhcp3/dhcpd.conf*: A partir de la MAC de cada nodo se define su ip y su nombre de host.

```
ddns-update-style ad-hoc;
# Definimos la mascara de red de una clase C
option subnet-mask 255.255.255.0;
option option-128 code 128 = string;
option option-129 code 129 = text;
get-lease-hostnames true;
group {
  # Activamos que los nodos reconozcan como router al
propio servidor para tener acceso al exterior.
  option routers 192.168.1.4;
   # Activamos el envío de nombres (hostname) además de
ips.
    use-host-decl-names on;
  # Definimos la subred 192.168.1.0 con máscara de red
255.255.255.0.
  subnet 192.168.1.0 netmask 255.255.255.0 {
    # Definimos el nodo con nombre: demohost
    host demohost {
      # Con su MAC lo identificamos.
       hardware ethernet 00:07:E9:94:B9:27; # CLIENTE #1 Labora-
torio
      # Y le asignamos una dirección fija.
       fixed-address 192.168.1.5;
           # Para finalmente servirle la imagen de arranque:
/fai/pxelinux.0
             # Esta imagen contiene el cargador de arranque:
Pxelinux.
       filename "/fai/pxelinux.0";
    }
  }
}
```

- Fichero: */etc/exports*: Exportamos /srv/ubuntumirror para poder ser accesible desde la instalación de los nodos.

```
/srv/fai/config\
                        192.168.1.4/255.255.255.0(async,ro,no_subtree_check)
/srv/fai/nfsroot\
          192.168.1.4/255.255.255.0(async,ro,no_subtree_check,no_root_squash)
/srv/ubuntumirror       192.168.1.4/255.255.255.0(async,ro,no_subtree_check)
```

- Fichero: */etc/hosts*: Definimos el nombre del servidor así como todos los nombres de los nodos asignándoles una ip.

```
# Ip para la red de bucle local
127.0.0.1 localhost
# Ip del servidor al cual llamaremos: desdeslinserver
192.168.1.4 desdeslinserver
# Ip del nodo de demostración que llamaremos: demohost
192.168.1.5 demohost
```

- Fichero: */etc/inetd.conf*: Se define el arranque del demonio tftp.

```
# Definimos que el superdemonio inet controle al programa
tftp que funciona con el protocolo udp.
# Para ello este se ejecutará como root y servirá el
contenido del directorio /srv/tftp
# En /srv/tftp se encuentra la imagen del cargador de
arranque, su configuración, el kernel y el initrd.
tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s
/srv/tftp
```

- Creación del enlace simbólico en /srv/ubuntumirror para simplificar las rutas en la configuración.

```
cd /srv
ln -s ../home/ubuntu_mirrors/ ubuntumirror
```

- Reinicio del demonio de dhcp

```
/etc/init.d/dhcp3-server restart
```

- Reinicio del demonio inetd

```
/etc/init.d/openbsd-inetd restart
```

Como en otros pasos de la configuración previa podemos obviar la edición de ficheros antes explicada usando los ficheros propuestos por Desdeslin:

```
cp -r /media/disk/configuracion_sistema_restante/etc /
```

#### 3.1.3.8. Configuración de la red

En este paso hemos de configurar la red para poder terminar de configurar FAI.

A continuación se detallan los ficheros a modificar y los comandos a ejecutar para llevar a cabo esta configuración.

- Fichero /etc/default/*dhcp3-server: Se configura para que sólo escuche por la interfaz de la red interna.*

```
INTERFACES="eth0"
```

- Fichero */etc/network/interfaces*: Fichero que nos define la función de las interfaces de redes. En nuestro caso la interfaz eth0 será la conectada a la red interior del cluster y, por ejemplo, será definida de forma estática como una clase C. La interfaz eth1 está conectada la red exterior del cluster para poder acceder a internet y otros servicios y en nuestra red se define via dhcp.

```
# La interfaz de bucle local
auto lo
iface lo inet loopback
# La interfaz de red externa
# Se configura por dhcp.
auto eth1
iface eth1 inet dhcp
# La interfaz de red interna (red cluster)
# Definimos una red de clase C
# El servidor tendrá la dirección 192.168.1.4.
auto eth0
iface eth0 inet static
address 192.168.1.4
netmask 255.255.255.0
network 192.168.1.0
broadcast 192.168.1.255
```

- Fichero */etc/resolv.conf*: Se propone un fichero de configuración tipo para la resolución de nombres en la red de la Universitat de Lleida por si la asignación de los servidores dns por dhcp fallará.

```
domain udl.net
search udl.net
nameserver 172.16.20.10
```

- Reinicio de la red: Reiniciaremos la red para contemple la nueva configuración.

```
/etc/init.d/networking restart
```

- Reinicio del servicio dhcp. Hemos configurado el servicio dhcp para que sólo escuche por una interfaz. Deberemos reiniciarlo para que los cambios surtan efecto.

```
/etc/init.d/dhcp3-server restart
```

#### 3.1.3.9. Instalación del nfsroot de FAI

Con:

```
fai-setup -v
```

configuramos la raíz de que servirá de base a FAI para su instalación automática y que será servida por NFS.

### 3.1.4. Requisitos previos para Configuración Virtualbox

#### 3.1.4.1. Distribución Ubuntu 8.10

La versión de FAI propuesta por el proyecto Desdeslin funciona en Ubuntu 8.10. Funcionó desde una instalación básica de Ubuntu 8.10 Server Edition. Si se usan otras versiones de FAI se puede elegir otra distribución basada en Debian.

#### 3.1.4.2. Configuración de paquetes propios

Al instalar la distribución está tendrá configuradas las fuentes de paquetes (repositorios) por defecto. Primero copiaremos el repositorio propuesto de Desdeslin en un directorio del sistema como pudiera ser */home/ubuntu_mirrors/*. Después suponemos que hemos obtenido un mirror de los paquetes oficiales de Ubuntu y que también los guardamos en */home/ubuntu_mirrors/*. *Habrá que* editar el fichero */etc/apt/sources.list* para que tanto el repositorio proporcionado por Desdeslin como el mirror se tengan en cuenta. El fichero propuesto por Desdeslin contiene:

```
# Paquetes modificados de Ubuntu 8.10 para Desdeslin
# El URI empieza por file para establecer un recurso accesible
# desde el sistema de ficheros
deb file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
deb-src file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
# Distribución intrepid. Secciones main y restricted. Binarios.
# En este caso el mirror se ha descargado en el
# directorio us.archive.ubuntu.com
deb file:/home/ubuntu_mirrors/us.archive.ubuntu.com/ubuntu/ intrepid main
restricted
# Distribución intrepid. Secciones main y restricted. Código
fuente.
deb-src file:/home/ubuntu_mirrors/us.archive.ubuntu.com/ubuntu/ intrepid
main restricted
# ...
```

Si suponemos que el sources.list propuesto se encuentra en: */media/disk/configuracion_virtualbox/etc/sources.list* podemos emplearlo directamente para sustituir nuestro sources.list.

```
cp /media/disk/configuracion_virtualbox/etc/sources.list /etc/apt/sources.list
```

#### 3.1.4.3. Configuración de la preferencia de los paquetes

Al tener en paralelo las versiones oficiales de los paquetes que hemos modificado y los paquetes modificados el sistema ha de decidir qué paquetes ha de instalar. Los paquetes no oficiales han sido definidos como pertenecientes a la sección *codip2p* (en lugar de main, universe o multiverse). Gracias al fichero */etc/apt/preferences* propuesto que se muestra a continuación forzaremos que el sistema prefiera los paquetes de la sección *codip2p* y por tanto los modificados.

```
Package: *
Pin: release c=codip2p
Pin-Priority: 1001
Package: *
Pin: release c=main restricted universe multiverse
Pin-Priority: 100
```

Así pues copiamos el fichero propuesto y actualizamos el sistema de paquetería con la nueva configuración:

```
cp /media/disk/configuracion_preferencia_paquetes/etc/apt/preferences
/etc/apt/preferences
apt-get update
```

#### 3.1.4.4. Instalación de FAI y otros paquetes

Una vez configurados los repositorios y el sistema de instalación de los paquetes procederemos a instalar los paquetes necesarios para instalación correcta de FAI.

- Demonio de dhcp
  ```
  apt-get install dhcp3-server
  ```
- Sistema básico de FAI
  ```
  apt-get install fai-quickstart
  ```
- Paquetes sugeridos por FAI
  ```
  apt-get install debmirror mknbi apt-move mkisofs grub aptitude
  ```
- Actualización de initramfs-tools con nuestro paquete modificado
  ```
  apt-get install initramfs-tools
  ```

#### 3.1.4.5. Configuración de FAI para proyecto codip2p

El paquete FAI adaptado para Ubuntu 8.10 no contempla todas las particularidades del proyecto codip2p.

A continuación se detallan todas las configuraciones y binarios específicos para codip2p.

- Fichero: */etc/fai/fai.conf*: Se añade la linea:
  ```
  FAI_DEBMIRROR=desdeslinserver:/srv/ubuntumirror
  ```
  para poder acceder al repositorio de paquetes modificados via nfs desde los nodos (se monta en */media/mirror*).
- Fichero: */etc/fai/apt/sources.list*: Se combinan los dos repositorios desde el punto de vista del nodo:

```
# El servidor sirve: /srv/ubuntumirror
# El nodo lo monta en: /media/mirror
# Repositorio con nuestros paquetes modificados
deb file:/media/mirror/codip2p/ intrepid codip2p
deb-src file:/media/mirror/codip2p/ intrepid codip2p
# Mirror de los repositorios de Ubuntu de EE.UU. para las
secciones habituales.
deb file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid main
restricted
deb-src file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid
main restricted
deb file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid-updates
main restricted
deb-src file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid-
updates main restricted
deb file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid univer-
se
deb-src file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid
universe
deb file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid-updates
universe
deb-src file:/media/mirror/us.archive.ubuntu.com/ubuntu/ intrepid-
updates universe
#...
```

- Fichero: */srv/fai/config/scripts/last/srv/50-misc*: Se descomenta la linea que monta el repositorio Debian via nfs.

```
[ "$FAI_DEBMIRROR" ] && echo "$FAI_DEBMIRROR $MNT-
POINT nfs ro 0 0" >>$target/etc/fstab
```

Si tenemos los ficheros propuestos por Desdeslin todos estos pasos se pueden ahorrar haciendo una copia de los ficheros proporcionados tal que:

```
cp -r /media/disk/configuracion_fai_codip2p/etc /me-
dia/disk/configuracion_fai/codip2p/srv /
```

#### 3.1.4.6. Configuración del sistema restante

Aunque no es especifico de FAI hay algunas configuraciones que han de ser establecidas en el sistema para que este funcione. A continuación se detallan todas las configuraciones especificas para este propósito.

- Fichero: */etc/dhcp3/dhcpd.conf*: A partir de la MAC de cada nodo se define su ip y su nombre de host.

```
ddns-update-style ad-hoc;
# Definimos la mascara de red de una clase C
option subnet-mask 255.255.255.0;
option option-128 code 128 = string;
option option-129 code 129 = text;
get-lease-hostnames true;
group {
  # Activamos que los nodos reconozcan como router al
propio servidor para tener acceso al exterior.
  option routers 192.168.1.4;
   # Activamos el envío de nombres (hostname) además de
ips.
    use-host-decl-names on;
  # Definimos la subred 192.168.1.0 con máscara de red
255.255.255.0.
  subnet 192.168.1.0 netmask 255.255.255.0 {
    # Definimos el nodo con nombre: demohost
    host demohost {
      # Con su MAC lo identificamos.
       hardware ethernet 00:07:E9:94:B9:27; # CLIENTE #1 Labora-
torio
      # Y le asignamos una dirección fija.
       fixed-address 192.168.1.5;
           # Para finalmente servirle la imagen de arranque:
/fai/pxelinux.0
             # Esta imagen contiene el cargador de arranque:
Pxelinux.
       filename "/fai/pxelinux.0";
    }
  }
}
```

- Fichero: */etc/exports*: Exportamos /srv/ubuntumirror para poder ser accesible desde la instalación de los nodos.

```
/srv/fai/config\
                        192.168.1.4/255.255.255.0(async,ro,no_subtree_check)
/srv/fai/nfsroot\
          192.168.1.4/255.255.255.0(async,ro,no_subtree_check,no_root_squash)
/srv/ubuntumirror       192.168.1.4/255.255.255.0(async,ro,no_subtree_check)
```

- Fichero: */etc/hosts*: Definimos el nombre del servidor así como todos los nombres de los nodos asignándoles una ip.

```
# Ip para la red de bucle local
127.0.0.1 localhost
# Ip del servidor al cual llamaremos: desdeslinserver
192.168.1.4 desdeslinserver
# Ip del nodo de demostración que llamaremos: demohost
192.168.1.5 demohost
```

- Fichero: */etc/inetd.conf*: Se define el arranque del demonio tftp.

```
# Definimos que el superdemonio inet controle al programa
tftp que funciona con el protocolo udp.
# Para ello este se ejecutará como root y servirá el
contenido del directorio /srv/tftp
# En /srv/tftp se encuentra la imagen del cargador de
arranque, su configuración, el kernel y el initrd.
tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s
/srv/tftp
```

- Creación del enlace simbólico en /srv/ubuntumirror para simplificar las rutas en la configuración.

```
cd /srv
ln -s ../home/ubuntu_mirrors/ ubuntumirror
```

- Reinicio del demonio de dhcp

```
/etc/init.d/dhcp3-server restart
```

- Reinicio del demonio inetd

```
/etc/init.d/openbsd-inetd restart
```

Como en otros pasos de la configuración previa podemos obviar la edición de ficheros antes explicada usando los ficheros propuestos por Desdeslin:

```
cp -r /media/disk/configuracion_sistema_restante/etc /
```

#### 3.1.4.7. Configuración de la red

En este paso hemos de configurar la red para poder terminar de configurar FAI.

A continuación se detallan los ficheros a modificar y los comandos a ejecutar para llevar a cabo esta configuración.

- Fichero */etc/network/interfaces*: Fichero que nos define la función de las interfaces de redes. En nuestro caso la interfaz eth0 será la conectada a la red interior del cluster y, por ejemplo, será definida de forma estática como una clase C. No obstante para poder comunicarnos con la red de ordenadores de Virtualbox tendremos que definir una interfaz de red bridge.

```
# La interfaz de bucle local
auto lo
iface lo inet loopback
# La interfaz de red interna (red cluster)
# la definimos en el nuevo dispositivo de red: br0
# Definimos una red de clase C
# El servidor tendrá la dirección 192.168.1.4.
# En el caso de esta red el router tiene dirección
192.168.1.1
# aunque esto no tiene la menor importancia porque
# el equipo no tiene salida a internet.
auto br0
iface br0 inet static
address 192.168.1.4 network 192.168.1.0
netmask 255.255.255.0
broadcast 192.168.1.255
gateway 192.168.1.1
bridge_ports all
bridge_maxwait 5
dns-nameservers 192.168.1.1
# El dispositivo físico eth0 lo configuramos como estático
y con dirección 0.0.0.0.
# De esa manera actúa correctamente como bridge.
auto eth0
iface eth0 inet static
address 0.0.0.0
```

- Necesitamos instalar el paquete bridge-utils que nos permitirá establecer el puente entre la tarjeta de red física y las tarjetas de red virtuales.

```
apt-get install bridge-utils
```

- Reinicio de la red: Reiniciaremos la red para contemple la nueva configuración.

```
/etc/init.d/networking restart
```

#### 3.1.4.8. Configuración de Virtualbox

- Instalamos las cabeceras del Kernel.

```
apt-get install linux-headers-‘uname -r‘
```

- Instalamos el paquete de Virtualbox.

```
apt-get install virtualbox-ose
```

- Creamos una máquina virtual de tipo genérico.
- Configuramos la red de Virtualbox en */etc/vbox/interfaces* con el fichero propuesto:

```
# Configuración estática de la interfaz en la red del host
#
# <interfaz><usuario><bridge>
vbox0 usuario br0
vbox1 usuario br0
vbox2 usuario br0
```

Dónde usuario es el usuario que usará Virtualbox para probar las máquinas nodos.

- Reiniciamos Virtualbox con:

```
/etc/init.d/vboxdrv restart
/etc/init.d/virtualbox-ose restart
```

- Terminamos de configurar la máquina virtual con las siguientes opciones:
  - Configuración, Red, Adaptador 1, Attached to: Interfaz anfitrión.
  - Configuración, Red, Adaptador 1, Mac address: Podemos especificar ahí su MAC para poderla usar más adelante.
  - Configuración, Red, Adaptador 1,Configuración de interface anfitrión, Interface Name : vbox0. (Cuando añadamos otro nodo tenemos que poner otra interfaz de las definidas en */etc/vbox/interfaces* como vbox1.
- Introducimos a nuestro usuario de Virtualbox en el grupo vboxusers para evitar problemas relacionados con permisos.

```
usermod -a -G vboxusers usuario
```

#### 3.1.4.9. Instalación del nfsroot de FAI

Con:

```
fai-setup -v
```

configuramos la raíz de que servirá de base a FAI para su instalación automática y que será servida por NFS.

## 3.2. Operativa diaria de FAI

Aquí veremos el uso del comando *fai-chboot* que nos cambia el arranque por defecto de los nodos. Veremos cómo hacer que arranque la instalación automatizada sin reinicio para hacer pruebas. Cómo hacer que el reinicio después de la instalación sea automático. Y finalmente cómo establecer que el nodo arranque de forma local una vez esté instalado correctamente.

Los ejemplos propuestos hablan de un solo host aunque podemos usar comodines para definir más de un host. Si por ejemplo tenemos los nodos: *nodop2p001, nodop2p002, nodop2p003, nodop2p004* podemos abstraerlos con: *nodop2p**.

Por último aconsejamos seguir el siguiente patrón de acción para evitarnos problemas con continuas re-instalaciones automáticas:

- Definir arranque de instalación automatizada en terminal con reinicio.

- Arrancar todos los nodos a instalar.
- Esperar un tiempo prudencial como 20 segundos para que todos los nodos arranquen.
- Definir arranque local de terminal.

### 3.2.1. Arranque de instalación automatizada en terminal sin reinicio

Usaremos:

| *fai-chboot -IFv nombre_de_host* |
|---|

para especificar que el host con nombre *nombre_de_host* haga una instalación automatizada sin reinicio al final de la misma. La opción F establece la variable *FAI_FLAGS* a: *verbose,sshd,createvt.*

### 3.2.2. Arranque de instalación automatizada en terminal con reinicio

Usaremos:

| *fai-chboot -IBv nombre_de_host* |
|---|

para especificar que el host con nombre *nombre_de_host* haga una instalación automatizada sin reinicio al final de la misma. La opción B establece la variable *FAI_FLAGS* a: v*erbose,sshd,reboot.*

### 3.2.3. Arranque local de terminal

Usaremos:

| *fai-chboot -ov nombre_de_host* |
|---|

para especificar que el host con nombre *nombre_de_host* haga un arranque local. Normalmente esto arrancará el disco duro especificado en la segunda entrada de arranque de la BIOS.

## 3.3. Personalización de FAI

Para personalizar FAI tenemos varias opciones. El concepto de clase (3.3.1 Cómo funcionan las clases )es el más potente. A partir de las características de los nodos podremos encuadrarlos en una clase u otra y aplicarle las particularidades de instalación de una clase determinada. También podemos definir cómo va a realizarse el particionado en los nodos (3.3.2 Particionamiento y sistemas de ficheros), ya sea para dedicar más espacio a una determinada partición o bien para preservar el contenido de otra. Una opción importante para poder realizar el multi-cluster es la de especificar qué paquetes instalar (3.3.3 Paquetes a instalar) en cada serie de nodos. Por último podremos afinar la instalación mediante la ejecución de scripts (3.3.4 Scripts específicos).

### 3.3.1. ¿Cómo funcionan las clases?

#### 3.3.1.1. ¿Qué es una clase?

Una clase es un atributo de un nodo. Este atributo servirá para especificar cómo ha de comportarse la instalación de FAI en cada uno de los nodos. Un nodo puede tener una o varias clases. Las clases pueden asignarse a un nodo, definirse en un fichero o generarse dinámicamente al ejecutar un script.

#### 3.3.1.2. Clases por defecto

Las clases por defecto que tiene cualquier nodo son: DEFAULT, LAST y el nombre de host del nodo.

#### 3.3.1.3. Definición de clases

En el fichero: */srv/fai/config/class/50-host-classes* encontramos la definición de clases a partir del nombre de host de las máquinas. En el ejemplo dado vemos como todos los nombres de host que empiecen por xubuntudesktop contienen entre otras la clase XUBUNTUDESKTOP.

```
#! /bin/bash
# assign classes hosts
# use a list of classes for our demo machine
case $HOSTNAME in
faiserver)
echo "FAIBASE DEMO FAISERVER" ;;
demohost)
echo "FAIBASE DHCPC DEMO" ;;
gnomehost)
echo "FAIBASE DHCPC DEMO XORG GNOME";;
xubuntudesktop*)
echo "FAIBASE DHCPC XUBUNTUDESKTOP";;
ubuntudesktop*)
echo "FAIBASE DHCPC UBUNTUDESKTOP";;
kubuntudesktop*)
echo "FAIBASE DHCPC KUBUNTUDESKTOP";;
atom*)
echo "FAIBASE DHCPC DEMO" ;;
*)
echo "FAIBASE DHCPC" ;;
esac
(ifclass I386 || ifclass AMD64) && echo GRUB
exit 0
```

#### 3.3.1.4. Prioridad de las clases

Las clases se definen de menor a mayor prioridad. En otras palabras las clases definidas en último lugar serán las que tengan mayor prioridad. Si en un proceso de la instalación sólo puede usarse una clase se usará la más prioritaria.

#### 3.3.1.5. Uso de las clases

Para determinar qué fichero de configuración usar un cliente de instalación buscar en la lista de clases definidas y usa todos los ficheros de configuración que concuerdan con esos nombres de clases.

Podremos ver este concepto de forma más clara con un ejemplo. El directorio */srv/fai/config/package_ config* se encarga definir qué paquetes instalar. Si suponemos que estamos instalando un nodo cuyo nombre de host empieza por xubuntudesktop. Vemos en su definición de clase por nombre de host:

```
FAIBASE DHCPC XUBUNTUDESKTOP
```

Es decir, que los ficheros que se leerán para saber qué paquetes instalar serán en este orden: XUBUNTUDESKTOP, DHCPC, FAIBASE, DEFAULT , el nombre del host y finalmente LAST.

#### 3.3.1.6. Documentación oficial

Para más información sobre el funcionamiento de las clases podemos consultar: */usr/share/doc/fai-doc/classes_ description.txt.*

### 3.3.2. Particionamiento y sistemas de ficheros

Para definir el particionamiento de los discos duros y qué sistemas de ficheros usar se usan los ficheros de configuración encontrados en */srv/fai/config/disk_ config* que son también procesados como clases. A continuación un ejemplo:

```
# generic disk configuration for one small disk
# disk size from 500Mb up to what you can buy today
#
# <type><mountpoint><size in mb>[mount options] [;extra options]
disk_ config disk1
primary / 150-500 rw,errors=remount-ro ; -c -j ext3
logical swap 40-500 rw
logical /var 90-1000 rw ; -m 5 -j ext3
logical /tmp 50-1000 rw ; -m 0 -j ext3
logical /usr 200-4000 rw ; -j ext3
logical /home 50- rw,nosuid ; -m 1 -j ext3
```

La etiqueta disk_config define un nuevo disco. Disk_config se sigue de disk1 que puede significar hda o sda, de disk2 que puede significar hdb o sdb y así sucesivamente.

A continuación viene la definición de particiones con:

- Tipo: Primaria o lógica.
- Punto de montaje: El camino entero empezando con / del sistema de ficheros a montar. Swap define una partición swap. Un guión (-) define que la partición no será montada.

- Tamaño: El tamaño de la partición en megabytes. Este valor se redondea para acomodarse a un número de cilindro. Hay diferentes maneras de definir el tamaño:
  - "200": 200 MB, ni más ni menos.
  - "100-300" : Mínimo de 100 MB y máximo de 300 MB.
  - "10-": Mínimo de 10 MB y máximo el tamaño entero del disco.
  - "-300": Mínimo de 1 MB y máximo de 300 MB.
- Puntos de montajes: Serán copiados al */etc/fstab*. Si el campo está vacío se pondrá al valor por defecto.
- Opciones extra: Hay más opciones extras como swap (partición swap), ext3 (usar ext3 en vez de auto en fstab),

Para poder preservar una partición de ser formateada se deberá definir su tamaño como preserveX dónde X es el número que identifica esa partición (en */dev/hdaX* o */dev/sdaX*).

Por ejemplo:

```
primary /backup preserve2 # 2
```

preservará la partición 2 siempre y cuando esta sea la segunda partición definida en el fichero.

### 3.3.3. Paquetes a instalar

El directorio */srv/fai/config/package_ config* se encarga definir qué paquetes instalar y también se rige por clases.

Un ejemplo de fichero sería:

```
# Definimos que los paquetes de la clase I386 se tienen que
instalar con el comando aptitude
# Concretamente instalaremos: linux-image-generic y memtest86+.
PACKAGES aptitude I386
linux-image-generic
memtest86+

PACKAGES aptitude CHROOT
linux-image-generic-

PACKAGES aptitude AMD64
linux-image-generic
memtest86+

PACKAGES aptitude DHCPC
dhcp3-client
#Definimos que los paquetes de la clase GRUB han de ser
instalados con aptitude.
# Habrá que instalar grub y a la vez desinstalar lilo si
estuviera instalado.
PACKAGES aptitude GRUB
grub lilo-

PACKAGES aptitude LILO
lilo grub-
```

Los comentarios empiezan por almohadilla (#). Cada comando empieza con la palabra PACKAGES seguido de un comando de esta lista: aptitude, apt-get, smart, y2pmsh, yast, yum, urpm, rpm. También se puede añadir una lista de nombres de clases después del comando apt-get (o aptitude) pero sólo se recomienda hacer esto en el fichero DEFAULT para seguir mejor los cambios. Si un paquete se ha de quitar debe ser seguido de un guión (-).

### 3.3.4. Scripts específicos

Los scripts específicos o hooks te dejan especificar funciones o programas que son ejecutados a diferentes pasos del proceso de instalación. Antes de que una tarea se llame FAI busca hooks existentes para esa tarea y los ejecuta. Los hooks también funcionan bajo el paradigma de las clases. Para crear un hook este debe llevar en el nombre como prefijo la tarea y terminar con el nombre de la clase. De esa manera con la tarea especificamos cuando se ha de ejecutar (con el prefijo) y en qué ocasiones debe ejecutarse (cuando el nodo forma parte de la clase). Todos los hooks se guardan en el directorio */fai/hooks*.

## 3.4. Actualización de los nodos

### 3.4.1. Reinstalación como actualización

Una vez instalados y personalizados los nodos se presenta la duda de la actualización de los nodos. Una opción válida podría ser regenerar el directorio

raíz que sirve FAI por nfs con nuestros cambios e iniciar de nuevo la instalación automatizada.

No obstante esta actualización presenta algunos inconvenientes como el tiempo que ocupa. Esto es así porque la instalación de FAI necesita reiniciar la máquina, iniciar la captura del cargador de arranque, kernel e initrd y seguir montando el sistema de instalación e iniciar la instalación.

### 3.4.2. Actualización tipo Debian

Gracias a que Ubuntu está basada en Debian podemos echar mano del sistema de paquetes para automatizar los cambios.

El sistema propuesto para las actualizaciones consiste en un repositorio de Debian que sirve paquetes específicos del cluster. Estos paquetes no sólo actualizan la configuración del entorno (ficheros de configuración de los diferentes demonios) lista para el propósito del cluster sino que recargan los demonios del sistema.

De esta manera el tiempo de actualización será mínimo y las actualizaciones serán fácilmente diseñables.

Para que los nodos puedan actualizarse simultáneamente podemos hacer uso del script rshall proporcionado en la documentación de FAI (Véase 3.6 Scripts Útiles), podemos emplear también el programa clusterssh o bien programar con el programa cron una actualización de todos los paquetes del sistema diaria (u otra frecuencia que se desee).

### 3.4.3. Actualización tipo FAI

Se describe la actualización de los nodos propuesta por la documentación oficial de FAI como una propuesta más a estudiar. Esta tiene la ventaja de usar la potencia de las clases. Sin embargo tiene el inconveniente de que sus scripts de actualización han de ser diseñados de tal manera que si son ejecutados varias veces generen el mismo resultado que si sólo son ejecutados una sola vez.

#### 3.4.3.1. ¿Como funciona una actualización de software?

Las actualizaciones de software usan los mismos ficheros de configuración que una instalación de FAI nueva. Pueden incluso usar los comandos por defecto de FAI. Por tanto se comportan casi de la misma manera que una instalación. Sin embargo algunos puntos son diferentes:

- Por defecto se usa la lista antigua de clases (la creada durante la instalación inicial) por lo que fai-class no se llama para generar una nueva lista de clases. Esto puede ser cambiado llamando a: fai - N softupdate.
- No se particiona ni se formatea ningún sistema de archivos.
- No se realiza un debootstrap del sistema base.
- FAI se salta aquellas tareas que sólo eran útiles durante la instalación como configurar el teclado o arrancar demonios especiales.
- FAI no impide que los paquetes pueden reiniciar demonios.

- FAI no se reinicia al final de una actualización de software.

Quitando esas salvedades tenemos estas tareas comunes con una instalación normal:

1. Definir clases (por defecto la lista antigua) y variables.
2. Actualizar los paquetes instalados.
3. Instalar el software nuevo.
4. Llamar a los scripts de configuración.
5. Guardar los registros.

#### 3.4.3.2. Como ejecutar una actualización de software

Como las actualizaciones de software usan la misma infraestructura que la instalación de FAI puedes empezarla usando el mismo comando fai que se usa para la instalación:

```
/usr/sbin/fai softupdate
```

Esto inicia una actualización de software. Si quieres usar actualizaciones de software en un sistema que no ha sido instalado con FAI la primera vez tendrás que ejecutar fai con el parametro -N: fai -N softupdate.

#### 3.4.3.3. Como realizar actualizaciones de software en masa.

De alguna manera hay que automatizar la ejecución del comando de inicio de la actualización en cada uno de los nodos. Varias soluciones se proponen:

- Cron: Podemos usar las entradas del crontab para iniciar la actualización. Si nuestra instalación es grande podemos pensar en incorporar un mecanismo aleatorio de demora del inicio de la misma.
- Inicio de la actualización de software de forma remota. Se pueden instalar mecanismos de inicio remoto como root en los nodos, preferentemente usando la conexión ssh con una llave de autorización para inicio como root.
  Herrramientas como clusterssh nos permiten ejecutar el comando */usr/sbin/fai softupdate* en ellas, mientras que los resultados pueden ver inmediatamente en los terminales abiertos para cada host.

#### 3.4.3.4. Cómo escribir una configuración preparada para las actualizaciones de software

Cuando uno quiere escribir actualizaciones de software ha de ser más cuidadoso que con los scripts de la instalación. La regla principal es que el resultado de la ejecución de un script una sola vez debe ser el mismo que la ejecución de ese mismo script dos veces.

Así que hay algunas recomendaciones que seguir:

- Nunca hay que añadir tal cual a los ficheros:
  ```
  echo "Mas opciones Configuración" >>/etc/fstab
  ```
  Esto es incorrecto. La manera correcta de resolver el problema consiste en comprobar manualmente si la linea ya existe antes de añadirla o bien usar la instancia *AppendIfNoSuchLine* de cfengine.

- Usa los variables del entorno de FAI para determinar qué haces en tus scripts de configuración:
  **FAI_ROOT**: Apunta al directorio raíz del nodo. En caso de actualizaciones de software es: /.
  **ROOTCMD** contiene un comando para hacer chroot en el nodo. Esta vacío cuando hacemos actualizaciones de software (Ya que / ya es nuestra raíz).
  **FAI_ACTION** contiene la acción ejecutada actualmente:
  **install** cuando se instala
  **softudpate** cuando se realiza una actualización de software.

- Reinicia demonios si necesario: La mayoría de los demonios sólo leen su configuración cuando se inician. Si se modifica la configuración tendrás que hacer que la recarguen usando:
  ```
  $ROOTCMD invoke-rc.d $algundemonio reload
  ```
  O bien reiniciar el demonio con:
  ```
  $ROOTCMD invoke-rc.d $algundemonio restart
  ```

- Otras cosas como programar un reinicio del sistema si un nuevo kernel se ha instalado.

## 3.5. Paquetes modificados

Los paquetes modificados que Desdeslin proporciona son los siguientes:

- initramfs-tools: Es el paquete encargado de crear la imagen de disco en RAM para sistemas normales. Parte de sus scripts son usados por live-initramfs. El paquete ha sido modificado para evitar unos problemas de rutas con librerías. Se ha añadido el binario wc.

- live-initramfs: Es el paquete encargado de crear la imagen de disco en RAM para sistemas de arranque por CDROM, RAM o red. Usa parte de los scripts de initramfs-tools. El paquete ha sido modificado para evitar unos problemas de rutas con librerías.

- fai-client, fai-doc, fai-nfsroot, fai-quickstart,fai-server: El paquete fai se modificó para actualizar los repositorios a la versión 8.10 de Ubuntu así como para añadir algunos paquetes a la instalación estándar del root del nfs servido de FAI. De esa manera se arreglan algunos bugs relacionados con la salida de la nueva distribución.

Los paquetes se proporcionan ya situados en un repositorio creado con el programa reprepro. Así mismo los paquetes están encuadrados dentro de la sección codip2p en vez de universe.

## 3.6. Scripts útiles

La documentación de FAI nos proporciona unos cuántos scripts que nos pueden ser de utilidad a la hora de trabajar con él. Veremos las posibilidades que nos ofrece cada uno de ellos.

- */usr/share/doc/fai-doc/examples/utils/all_hosts*: Determina la lista de todos los hosts que responden a un ping. Para ello debemos tener un NIS y tendremos que editar el fichero all_hosts para reflejar la nomenclatura de nuestros hosts. Además el script *prnetgr* (encontrado en este mismo directorio) debe estar en el path de ejecución.
- */usr/share/doc/fai-doc/examples/utils/create-nfsroot-tar:* Crea un archivo tar de los archivos de fai excluyendo el fichero dhcpd.conf para poder transportar FAI a otras máquinas sin Debian.
- */usr/share/doc/fai-doc/examples/utils/mkdebmirror:* Gracias a debmirror y rsync podemos crear un mirror de Debian parcial.
- */usr/share/doc/fai-doc/examples/utils/prtnetgr:* Imprime todos los miembros de un grupo de red convirtiendo el arbol de red en un listado simple. Este script es usado por *all_hosts*.
- */usr/share/doc/fai-doc/examples/utils/rshall:* Ejecuta un comando en todos los nodos que son salida del script *all_hosts* gracias a rsh.
- */usr/share/doc/fai-doc/examples/utils/tlink:* Crea un enlace simbólico para asignar el arranque del nodo via la tarjeta de red (TFTP y BOOTP) a una imagen de arranque determinada. El listado de los nodos está descrito en el mismo script.
- *perl -e 'for (1..25) {printf "192.168.42. %s atom %02s\n",$_,$_;}'*: Crea entradas para usar en el fichero */etc/hosts* basado en el prefijo atom, desde 1 hasta 25. Además rellena los números inferiores a 10 con un 0. Es decir tendríamos atom05 en lugar de atom5. La dirección de red empieza por 192.168.42.

Además de los scripts por defecto de FAI Desdeslin proporciona los siguientes scripts basados en un fichero de texto con tres columnas separadas por tabuladores que contienen: nombre de host, ip de host y MAC del host tal que:

```
nodo01 192.168.1.5 aa:bb:cc:dd:ee:f1
nodo02 192.168.1.6 aa:bb:cc:dd:ee:f2
nodo03 192.168.1.7 aa:bb:cc:dd:ee:f3
```

- Generador de *dhcpd.conf* (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.1 para su contenido). Genera un dhcpd.conf válido a partir de la tabla de hosts, IPs y MACs.
- Generador de */etc/hosts* (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.2 para su contenido). Genera la relación de hosts para el formato del fichero /etc/hosts a partir de nuestros ficheros.

- Arranca equipos (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.3 para su contenido) . Llama a etherwake para que mande el paquete especial wake-on-lan (arranque de equipos por red) por la interfaz de red que le digamos.

Por último Desdeslin nos proporciona estos otros útiles scripts:

- Generador del fichero de configuración general del demonio DHCP (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.4 para su contenido) . Genera un fichero */etc/default/dhcp3-server* para que el demonio sólo sirva paquetes dhcp por la interfaz de red que le especifiquemos.
- Generador del fichero de configuración de las interfaces de red (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.5 para su contenido) . Genera un fichero /etc/network/interfaces a partir de la configuración de red suministrada por su linea de comandos.
- Generador de las lineas del repositorio codip2p para el fichero de configuración de repositorio (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.6 para su contenido) . Genera las lineas necesarias para la configuración del repositorio de codip2p en el fichero de configuración de repositorio a partir de la ruta en el sistema de ficheros del repositorio.
- Script de instalación de automática de la configuración de laboratorio 1 (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.7 para su contenido) . Hace uso de los ficheros de configuración propuestos para la configuración de laboratorio 1. Así mismo hace uso de otros scripts. Con este script después de una instalación de Ubuntu se puede configurar un servidor para adoptar la configuración de laboratorio 1.
- Script de instalación de automática de la configuración de laboratorio 2 (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.4.8 para su contenido) . Hace uso de los ficheros de configuración propuestos para la configuración de laboratorio 2. Así mismo hace uso de otros scripts. Con este script después de una instalación de Ubuntu se puede configurar un servidor para adoptar la configuración de laboratorio 2.

## 3.7. Ficheros de configuración (Relación de)

- */etc/apt/sources.list*: Configuración de los paquetes de nuestra distribución. Se debe adaptar al medio dónde se encuentran los paquetes. Debe añadirse también la ruta a los paquetes modificados por Desdeslin.
- */etc/apt/preferences*: Configuración del sistema de instalación de paquetes. Las modificaciones incorporadas a este fichero establecen los paquetes modificados por Desdeslin como preferentes a la hora de instalarse.
- */etc/fai/fai.conf*: Configuración principal de FAI. Podemos definir entre otros el acceso al mirror de paquetes.
- */etc/fai/apt/sources.list*:Configuración de los paquetes de nuestra distribución pero desde el punto de vista del nodo.

- */etc/dhcp3/dhcpd.conf*: Configuración del demonio dhcp. Contiene la relación entre la MAC del equipo y la IP que se le debe asignar a la hora de arrancar por red.
- */etc/inetd.conf*: Configuración del superdemonio inet. Usamos este fichero para iniciar el servicio Tftp.
- */etc/default/dhcp3-server*: Configuración de las opciones del demonio dhcp. Nos permite especificar por qué puerto tiene que escuchar dhcp.
- */etc/init.d/iptables_ codip2p*: Script proporcionado por Desdeslin para configurar al servidor como router en la red del cluster.
- */etc/network/interfaces*: Configuración de las interfaces de redes del equipo.
- *tabla_ base.txt* (Véase Apéndice B Código Fuente de Desdeslin. Apartado B.5.1 para su contenido): Relación de configuración de hostname, ip y MAC de ejemplo para la instalación automática.

# Capítulo 4

# Modificación de los paquetes de FAI

En este capítulo veremos la casuística de Ubuntu 8.10. Este no permite por defecto servir instalaciones de Ubuntu 8.10. Si intentamos modificar la configuración veremos que no es evidente, que tenemos que modificar las paquetes. Aquí veremos qué ficheros se han modificado, cómo regenerar los paquetes y cómo generar un repositorio que incluya esos paquetes para poder ser usados fácilmente desde un fichero sources.list.

## 4.1. Problemática con la versión Ubuntu 8.10 de FAI.

Uno de los requisitos del proyecto es trabajar con la distribución Ubuntu 8.10. El paquete proporcionado por Ubuntu 8.10 del programa FAI está configurado para servir Ubuntu 8.04. Si uno modifica el fichero /etc/fai/make-fai-nfsroot.conf para que la instalación por defecto sirva Ubuntu 8.10 se encuentra con varios problemas.

Uno de ellos es que el paquete aptitude no se encuentra. Otro es la aparición del mensaje *error while loading libpcre.so.3.* Finalmente hay un mensaje que se repite continuamente del programa perl: *en_ US.UTF_ 8 locale not being correctly installed.*

Esto hace imposible el uso de FAI en Ubuntu 8.10 como servidor de instalaciones de Ubuntu 8.10. La solución pasa por modificar los paquetes de FAI y subsanar estos errores.

## 4.2. Modificación de paquetes.

### 4.2.1. Paquete FAI

- Fichero: conf/make-fai-nfsroot.conf (Véase B Código fuente de Desdeslin (Apartado B.1.1) para ver el archivo entero):

- Se modifica la variable FAI_DEBOOTSTRAP para obtener las fuentes de intrepid.

  *FAI_DEBOOTSTRAP=*
  *"intrepid http://archive.ubuntu.com/ubuntu"*

- Se descomenta la linea *NFSROOT_HOOKS=/etc/fai/nfsroot-hooks/* para activar los hooks del sistema de ficheros unionfs.

- Se añaden paquetes a la linea *FAI_DEBOOTSTRAP_OPTS="–exclude=dhcp-client,info"* para evitar problemas con paquetes y mensajes repetitivos molestos.

  *FAI_DEBOOTSTRAP_OPTS="–exclude=dhcp-client,info –include=language-pack-en-base,aptitude,fontconfig,defoma,anthy,belocs-locales-bin"*

- Fichero: conf/sources.list (Véase B Código fuente de Desdeslin (Apartado B.1.2) para ver el archivo entero):
  Se modifican todas las direcciones para que apunten a la distribución intrepid.

  *deb http://archive.ubuntu.com/ubuntu intrepid main restricted universe multiverse*
  *deb http://security.ubuntu.com/ubuntu intrepid-security main restricted universe multiverse*
  *deb http://ppa.launchpad.net/fai/ubuntu intrepid main restricted universe multiverse*

- Fichero: debian/changelog (Véase B Código fuente de Desdeslin (Apartado B.1.3) para ver el archivo entero):
  Se añade información sobre los cambios realizados.

  *fai (3.2.4+svn4838-0ubuntu2) intrepid; urgency=low*
  ** Updated fai for serving Ubuntu intrepid*
  *– Adrian Gibanel <adrian.gibanel.lopez@gmail.com>Tue, 06 Jan 2009 11:05:34 +0200*

- Fichero: debian/fai-server.dirs (Véase B Código fuente de Desdeslin (Apartado B.1.4) para ver el archivo entero):
  Fichero que crea los directorios en el sistema destino. Se añade la carpeta que contiene el script de los hooks de unionfs.

  *etc/fai/nfsroot-hooks*

- Fichero: debian/fai-server.install (Véase B Código fuente de Desdeslin (Apartado B.1.5) para ver el archivo entero):
  Fichero que crea los ficheros en el sistema destino. Se añade el script que contiene los hooks de unionfs.

  *etc/fai/nfsroot-hooks/10-unionfs-workaround*

- Fichero: *examples/simple/hooks/faiend.FAIBASE.source* (Véase B Código fuente de Desdeslin (Apartado B.1.6) para ver el archivo entero):
  Script que crea los directorios */var/lock* y */var/run* en la partición /var del sistema a instalar para evitarnos problemas de montaje en el arranque del nodo.

### 4.2.2. Paquete initramfs-tools

- Fichero: *debian/changelog*. (Véase B Código fuente de Desdeslin (Apartado B.2.1) para ver el archivo entero).

```
initramfs-tools (0.92bubuntu17) intrepid; urgency=low
* Adapt initramfs-tools so that Fully Automatic Installation serves an
Ubuntu 8.10 ok.
– Adrian Gibanel <adrian.gibanel.lopez@gmail.com>Wed, 03 Jan 2009
01:50:37 +0100
```

- Fichero: *init* (Véase B Código fuente de Desdeslin (Apartado B.2.2) para ver el archivo entero).
  Evitamos problemas de montado de los sistemas de ficheros mediante el uso de las dos sintaxis del comando mount.

```
mount -n –move /sys ${rootmnt}/sys || mount -n -o move /sys ${root-
mnt}/sys +mount -n –move /proc ${rootmnt}/proc || mount -n -o move
/proc ${rootmnt}/proc
```

- Fichero: *scripts/local* (Véase B Código fuente de Desdeslin (Apartado B.2.3) para ver el archivo entero).
  Evitamos problemas de montado de los sistemas de ficheros mediante el uso de las dos sintaxis del comando mount.

```
mount –move ${rootmnt} /host || mount -o move ${rootmnt} /host
...
mount –move /host ${rootmnt}/host || mount -o move /host ${root-
mnt}/host
```

### 4.2.3. Paquete live-initramfs

- Fichero: hooks/live (Véase B Código fuente de Desdeslin (Apartado B.3.1) para ver el archivo entero).
  Hacemos que el programa wc se copie porque se usa en los scripts de inicio pero no viene incluido por defecto.

```
copy_ exec /usr/bin/wc /bin
```

- Fichero: *scripts/live* (Véase B Código fuente de Desdeslin (Apartado B.3.2) para ver el archivo entero).
  Evitamos problemas de montado de los sistemas de ficheros mediante el uso de las dos sintaxis del comando mount. Por otro lado regeneramos la configuración de ruta hacia las librerías dinámicas que se pierde sin razón alguna después de un comando mount. Para evitar problemas relacionados con unionfs cambiamos los permisos de algunos ficheros que se modifican en la instalación.

```
mount -n –bind /sys /root/sys || mount -n -o bind /sys /root/sys
mount -n –bind /proc /root/proc || mount -n -o bind /proc /root/proc
mount -n –bind /dev /root/dev || mount -n -o bind /dev /root/dev
mount -r –move "${copyto}" "${rootmnt}" || mount -r -o move
"${copyto}" "${rootmnt}"
mount -r –move ${copyto} ${copyfrom} || mount -r -o move ${copyto}
${copyfrom}
mount –bind ${exposedrootfs} ${rootmnt} || mount -o bind ${expose-
drootfs} ${rootmnt} || \
/root/sbin/ldconfig.real
chmod 600 /root/etc/fstab
chmod 600 /root/etc/live.conf
chmod 600 /root/etc/environment
chmod 600 /root/etc/network/interfaces
mount –move "${d}" "${rootmnt}/live/${d##*/}" || mount -o move
"${d}" "${rootmnt}/live/${d##*/}"
mount –move /cow "${rootmnt}/live/cow" || mount -o move /cow
"${rootmnt}/live/cow"
mount -o bind "${devname}" $mountpoint || mount –bind "${devna-
me}" $mountpoint || continue
```

## 4.3. Construcción de los paquetes.

Una vez modificado el código fuente de los paquetes de Ubuntu tendremos que generar tanto los paquetes binarios como los paquetes de codigo fuente. Para ello primero nos aseguraremos de que cumplimos todos las dependencias de compilación del paquete y ejecutaremos en el directorio raíz del paquete:

```
dpkg-buildpackage -rfakeroot .
```

Esto nos generará uno o varios paquetes binarios con extensión .deb, un paquete con el código fuente con extensión .tar.gz, un fichero que contiene los cambios del paquete con extensión .changes y finalmente un fichero que contiene la firma del paquete con extensión .dsc.

## 4.4. Generación del repositorio

El conjunto de paquetes generados a partir de nuestras modificaciones puede instalarse mediante el comando dpkg. No obstante esta opción de instalación nos quita todas las ventajas del sistema de paquetería de Debian. Por ejemplo no se instalan automáticamente las dependencias.

Para podernos aprovechar de esas ventajas tendremos que conformar un repositorio Debian con los paquetes. Para ello primero obtendremos los ficheros la salida de la construcción de paquetes. Es decir, necesitaremos los paquetes binarios, los paquetes de código fuente, el fichero de cambios y el fichero de firmas.

Por otro lado necesitamos un directorio con espacio libre dónde pondremos la salida de nuestro repositorio. Es desde ahí desde dónde trabajaremos.

Los pasos a seguir para la generación del repositorio serán:

1. Instalar el paquete *reprepro*:
   ```
   apt-get install reprepro
   ```
2. Crear el fichero *conf/distributions* en el repositorio destino con el contenido:
   ```
   Origin: Adrian Gibanel Lopez
   Label: codip2p_label
   Suite: unstable
   Codename: intrepid
   Architectures: i386 all source
   Components: codip2p
   Description: Fai packages for serving Ubuntu 8.10 without any problem.
   ```
3. Para cada uno de los paquetes haremos lo siguiente:
   ```
   reprepro –ignore=missingfile –ignore=wrongdistribution -Vb . include intrepid /ruta/a/paquetes_modificados/fai_3.2.4+svn4837-0ubuntu2_i386.changes
   ```
   De esa manera le decimos a reprepro que nos cree un repositorio para la distribución intrepid ignorando cualquier error de fichero que falte y cualquier error relacionado con una distribución incorrecta.

Finalmente en el directorio del repositorio encontraremos los directorios habituales de un repositorio como: *dists*, *pool* y las particulares de reprepro como *db* o *conf*. Podremos usar el repositorio creado desde una ruta local con una entrada en el fichero *sources.list* parecida a esta:

```
deb file:/home/ubuntu_mirrors/codip2p/ intrepid codip2p
```

# Capítulo 5

# Conclusiones y trabajo futuro

## 5.1. Conclusiones

En este proyecto fin de carrera se ha llevado a cabo un estudio pormenorizado de todas las opciones que el paquete Fully Automatic Installation junto a otros scripts nos pueden ofrecer.

La arquitectura de Fully Automatic Installation ha sido claramente definida así como el funcionamiento del sistema de instalación automática. No hay que olvidar el manual de usuario que no sólo cubre el uso normal del sistema sino que además también describe como se implanta y cómo se puede personalizarlo para adecuarlo a nuestras necesidades.

La imposibilidad de usar los paquetes oficiales de la distribución de Ubuntu nos ha hecho dedicar un capítulo a las modificaciones que se han de hacer a los paquetes para que estos sean funcionales. No sólo se detalla como modificarlos sino cómo generar los paquetes binarios y cómo servirlos en forma de repositorio.

Con todo ello hemos cumplido los objetivos que nos habíamos marcado al inicio del trabajo. Entre ellos destacan:

- **Instalación automatizada**. Proporcionar una instalación automatizada para un cluster.
  Gracias al completo manual de usuario podemos instalar Fully Automatic Installation en un cluster y configurarlo para nuestras necesidades.
- **Actualización automatizada**. El manual nos descarta por su tiempo de ejecución la actualización basada en la reinstalación de todos los nodos desde cero. Así mismo podemos optar por actualizar el sistema mediante paquetes de Debian personalizados que actualizamos o bien por usar los potentes pero complicados scripts de FAI para actualizaciones de software.
- **Multi cluster**: Gracias a la potencia de las clases hemos visto como podemos segmentar un cluster en varios clusters, por ejemplo, a partir del nombre de los hosts.

Al inicio del proyecto contábamos con un cluster de ordenadores que no se podía segmentar. Ahora, gracias a toda la instalación y configuración propuesta por Desdeslin podemos implementar no ślo un cluster muy configurable sino un multi cluster. Además cada particularidad de los nodos puede ser representada y configurada de forma cómoda gracias a las clases.

## 5.2. Trabajo futuro

### 5.2.1. Virtualización de servicios en servidor

Actualmente el modelo que se emplea en el cluster de cálculo en el que es posible que se implemente Desdeslin es el de servidor-cliente. Se instala una distribución de GNU/Linux con los paquetes adecuados en el servidor. En los nodos se instala de forma automática una distribución de GNU/Linux mínima para realizar los cómputos y la monitorización en la misma. Una mejora para este modelo es separar cada una de las funciones del servidor en diferentes máquinas virtuales dedicadas en el servidor: Servidor de instalaciones automatizadas, servidor de repositorio, servidor de cómputo, servidor de monitorización, etcétera. Entre las diferentes ventajas que nos proporciona la virtualización podemos citar la movilidad. Por ejemplo podemos probar una mejora de una de esas máquinas virtuales en otra máquina física y actualizar la máquina virtual cuando este lista sin ningún tipo de problema de hardware.

### 5.2.2. Virtualización de cómputo en nodos. Multi cluster virtual.

Esta mejora vendría a ser la continuación de la mejora: 5.2.1 Virtualización de servicios en servidor. El modelo de cluster antes de Desdeslin hacia muy difícil la concepción de multi-cluster. Es decir, poder dividir los nodos de un cluster existente en varios segmentos y trabajar con esos segmentos como si se tratarán de clusters diferentes pero conservando el mismo servidor. Con Desdeslin esto es posible de una forma fácil siempre y cuando se conserve la distribución de GNU/Linux a servir (Ubuntu 8.10, Ubuntu 8.04, Debian,...).

La mejora propuesta consiste en dejar de lado los segmentos físicos y crear de todo un conjunto de nodos físicos varios conjuntos de nodos virtuales. La técnica para realizar esto implica no sólo virtualizar los sistemas operativos del lado del servidor sino también virtualizar los sistemas operativos del lado del cliente.

Para ver más claro veamos el modelo tradicional en la figura 5.1. En cada nodo del cluster sólo se ejecuta un sistema operativo y todos ellos están conectados al servidor mediante un switch. Ahora veamos la figura 5.2 que representa el esquema físico del multi cluster virtual. Tenemos cuatro pcs: A, B, C, D dónde en cada uno de ellos se alojan dos sistemas operativos virtuales (OS1 y OS2). Finalmente el esquema lógico plasmado en la figura 5.3 nos da idea de las capacidades del mismo. De cara al usuario tenemos dos redes de clusters independientes (que luego pueden estar interrelacionadas si se desea). Una de las redes es la que tiene todo sistemas operativos OS1 mientras que la otra es la compuesta por sistemas operativos OS2. Todo esto se consigue sin invertir en hardware nuevo.

Así pues cada nodo ejecuta dos máquinas virtuales correspondientes a dos nodos de dos clusters diferentes que se comunican con dos máquinas virtuales especificas de cada cluster virtual en el servidor.

De esta manera podemos aprovechar el hipotético cómputo sobrante en los nodos en otro cluster pero sin tener que duplicar el hardware.

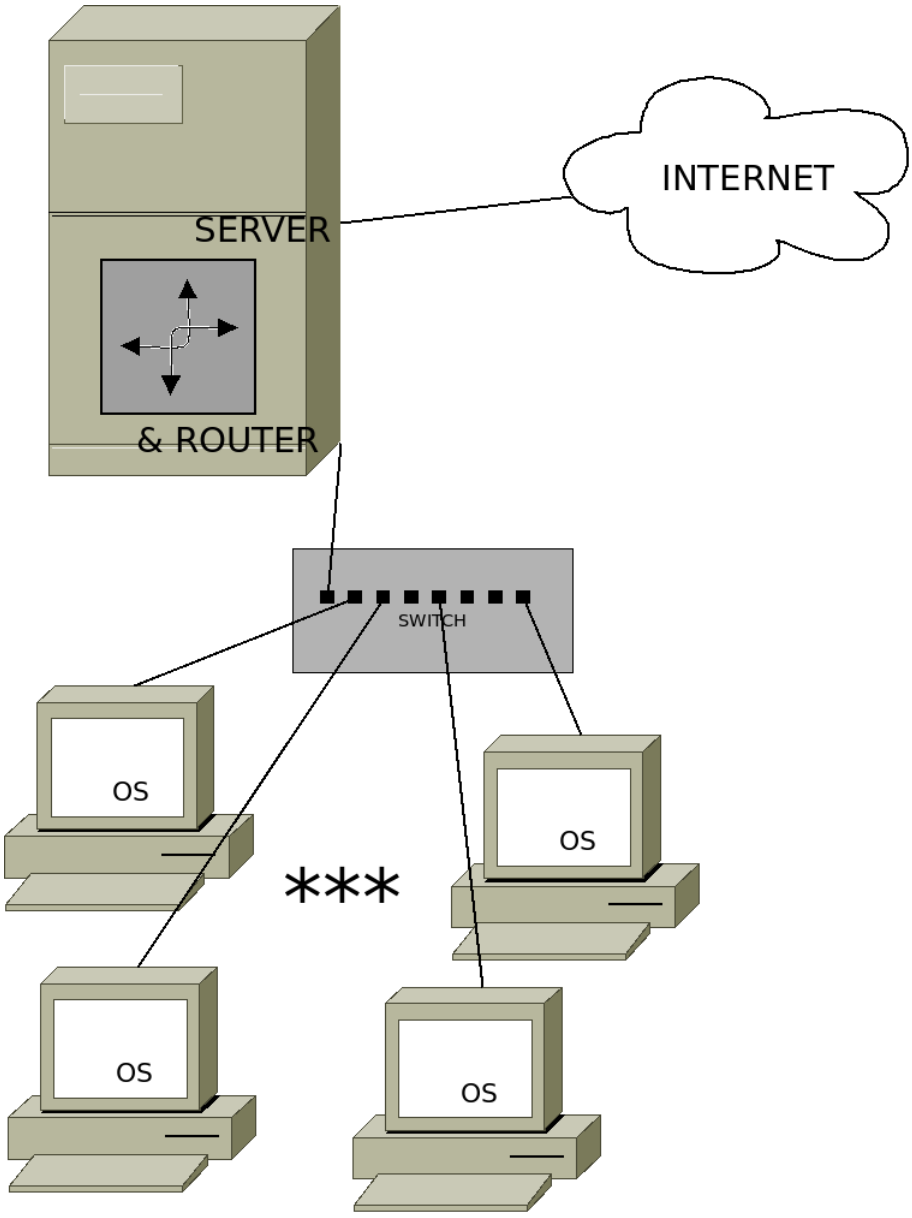



Figura 5.1: Esquema del cluster tradicional

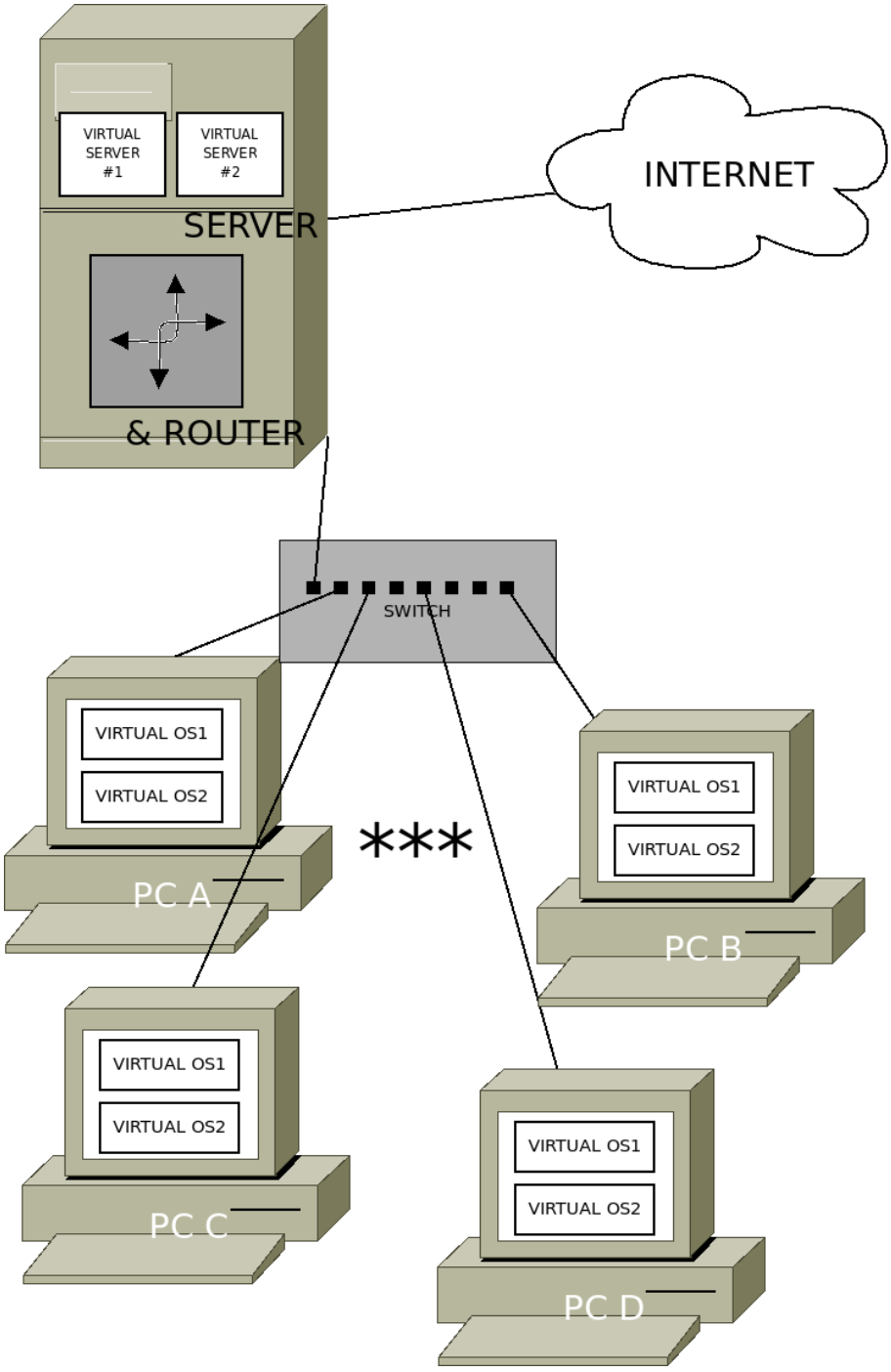



Figura 5.2: Esquema físico del multicluster virtual

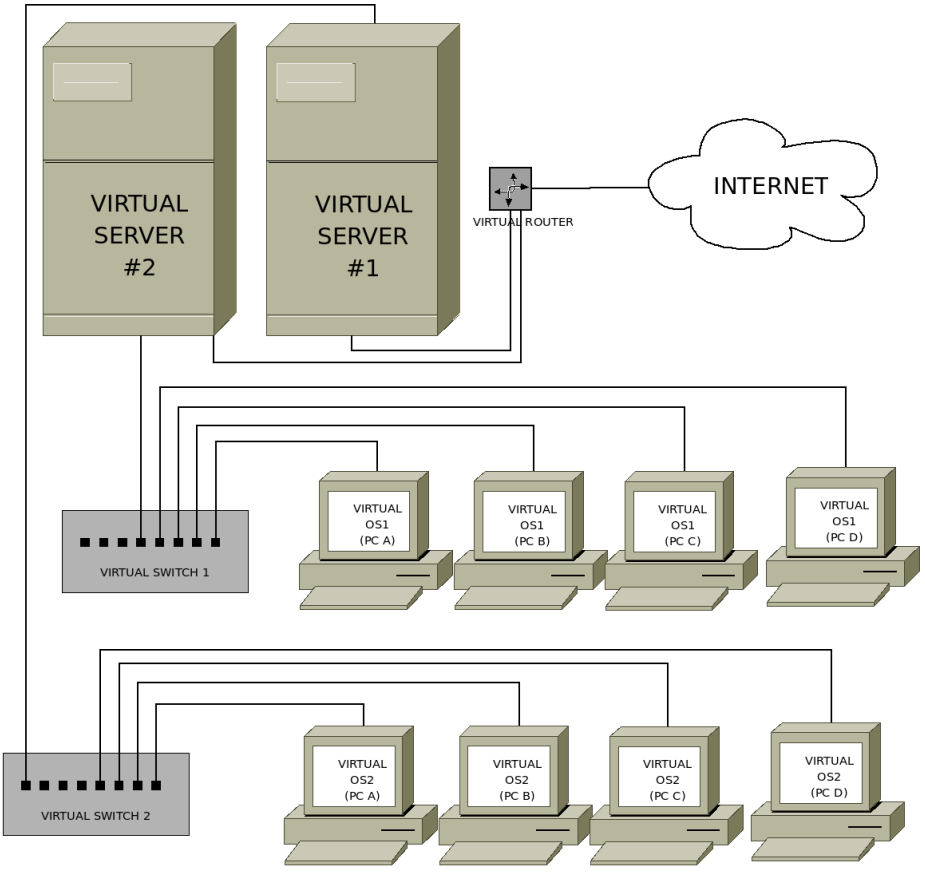



Figura 5.3: Esquema lógico del multicluster virtual

### 5.2.3. Monitorización de nodos de multi cluster virtual.

A partir de la mejora propuesta: 5.2.2 Virtualización de cómputo en nodos tendremos que replantearnos la monitorización. Así pues al ejecutar las herramientas clásicas de monitorización dentro del cluster virtual obtendremos los datos del cluster virtual. Así pues un nodo podrá parecer que realiza más cómputo dentro de un cluster virtual sin razón aparente. La razón oculta podría ser que ese mismo nodo dentro del otro cluster virtual realiza menos cómputo porque así se le ha ordenado. La primera aproximación a la resolución del problema que se me ocurre es que los diferentes sistemas de monitorización del multi cluster virtual tendrán que cruzar sus datos para obtener datos fiables del rendimiento del sistema.

# Apéndice A

# Contenido del CDROM

- *configuracion_previa/laboratorio1* – Contiene todos los archivos necesarios para cumplir los requisitos necesarios para la configuración de laboratorio 1
  - *01_distribucion_ubuntu_8.10*
    - *01_distribucion_ubuntu_8.10.txt* – Explicación de cómo instalar Ubuntu
  - *02_configuracion_de_paquetes_propios*
    - *02_configuracion_de_paquetes_propios.txt* – Explicación de cómo configurar los paquetes propios
    - *etc/apt/sources.list* – Fichero de configuración de repositorios propuesto para el servidor
  - *03_configuracion_preferencia_paquetes*
    - *03_configuracion_preferencia_paquetes.txt* – Explicación de cómo hacer preferentes nuestros paquetes
    - *etc/apt/preferences* – Fichero que establece la preferencia de la sección codip2p sobre las demás
  - *04_instalacion_de_fai_y_otros_paquetes*
    - *04_instalacion_de_fai_y_otros_paquetes.txt* – Explicación de la instalación de FAI y otros paquetes necesarios
  - *05_configuracion_fai_proyecto_codip2p*
    - *05_configuracion_fai_proyecto_codip2p.txt* – Explicación de la adaptación de la configuración de FAI para el proyecto codip2p
    - *etc/fai/apt/sources.list* – Fichero de configuración de repositorios propuesto para los nodos
    - *etc/fai/fai.conf* – Fichero de configuración de FAI propuesto adaptado para el proyecto codip2p
    - *srv/fai/config/scripts/last/50-misc* – Hook que permite el montado del repositorio de paquetes por nfs en los nodos al reinicio de la instalación
  - *06_configuracion_sistema_restante*

        - *06_configuracion_sistema_restante.txt* – Explicación de otras configuraciones necesarias
        - *etc/dhcp3/dhcpd.conf* – Fichero de configuración de la asignación de ips a MACs del demonio DHCP
        - etc/*exports* – Fichero de configuración del demonio nfs
        - *etc/hosts* – Fichero de configuración que establece las relaciones entre nombres de máquinas y sus ips a nivel local del servidor
        - etc/*inetd.conf* – Fichero de configuración del superservidor inetd para arranque automático del demonio tftpd
    - *07_configuracion_red*
        - *07_configuracion_red.txt* – Explicación de la configuración de la red
        - *etc/default/dhcp3-server* – Fichero de configuración general del comportamiento del demonio DHCP
        - *etc/init.d/iptables_codip2p* – Script de configuración de las iptables para que el servidor actúe como router
        - etc/*network/interfaces* – Fichero de configuración de las interfaces de red propuesto
        - etc/*resolv.conf* – Fichero de configuración para la resolución de nombres en la red de la UdL
    - *08_instalacion_nfsroot_fai*
        - *08_instalacion_nfsroot_fai.txt* – Explicación de la creación del nfsroot de fai
    - *resumen_laboratorio1.txt* – Resumen de todos los pasos necesarios para llevar a cabo la configuración de laboratorio 1
    - *script_automatico_laboratorio1.sh* – Script que tras una instalación de Ubuntu permite configurar el servidor como la configuración de laboratorio 1 a partir de los parámetros encontrados en el mismo script y de la tabla proporcionada por el fichero *../../scripts_utiles/tabla_base.txt*

- *configuracion_previa/laboratorio2* – Contiene todos los archivos necesarios para cumplir los requisitos necesarios para la configuración de laboratorio 2
    - *01_distribucion_ubuntu_8.10*
        - *01_distribucion_ubuntu_8.10.txt* – Explicación de cómo instalar Ubuntu
    - *02_configuracion_de_paquetes_propios*
        - *02_configuracion_de_paquetes_propios.txt* – Explicación de cómo configurar los paquetes propios
        - *etc/apt/sources.list* – Fichero de configuración de repositorios propuesto para el servidor
    - *03_configuracion_proxy_paquetes*

  - *03_configuracion_proxy_paquetes.txt* – Explicación de cómo configurar el proxy de paquetes
  - *etc/apt/sources.list* – Fichero de configuración de repositorios propuesto para el servidor tras aplicar la configuración del proxy de paquetes
  - etc/*apt-proxy/apt-proxy-v2.conf* – Fichero de configuración del proxy de paquetes
- *04_configuracion_preferencia_paquetes*
  - *04_configuracion_preferencia_paquetes.txt* – Explicación de cómo hacer preferentes nuestros paquetes
  - *etc/apt/preferences* – Fichero que establece la preferencia de la sección codip2p sobre las demás
- *05_instalacion_de_fai_y_otros_paquetes*
  - *05_instalacion_de_fai_y_otros_paquetes.txt* – Explicación de la instalación de FAI y otros paquetes necesarios
- *06_configuracion_fai_proyecto_codip2p*
  - *06_configuracion_fai_proyecto_codip2p.txt* – Explicación de la adaptación de la configuración de FAI para el proyecto codip2p
  - *etc/fai/apt/sources.list* – Fichero de configuración de repositorios propuesto para los nodos que aprovecha el proxy de paquetes del servidor
  - etc/fai/*fai.conf* – Fichero de configuración de FAI propuesto adaptado para el proyecto codip2p
  - etc/fai/*make-fai-nfsroot.conf* – Fichero de configuración del nfsroot de FAI que aprovecha el proxy de paquetes del servidor
  - *srv/fai/config/scripts/last/50-misc* – Hook que permite el montado del repositorio de paquetes por nfs en los nodos al reinicio de la instalación
- *07_configuracion_sistema_restante*
  - *07_configuracion_sistema_restante.txt* – Explicación de otras configuraciones necesarias
  - *etc/dhcp3/dhcpd.conf* – Fichero de configuración de la asignación de ips a MACs del demonio DHCP
  - etc/*exports* – Fichero de configuración del demonio nfs
  - etc/*hosts* – Fichero de configuración que establece las relaciones entre nombres de máquinas y sus ips a nivel local del servidor
  - etc/*inetd.conf* – Fichero de configuración del superservidor inetd para arranque automático del demonio tftpd
- *08_configuracion_red*
  - *08_configuracion_red.txt* – Explicación de la configuración de la red
  - *etc/default/dhcp3-server* – Fichero de configuración general del comportamiento del demonio DHCP

        - *etc/network/interfaces* – Fichero de configuración de las interfaces de red propuesto
        - etc/*resolv.conf* – Fichero de configuración para la resolución de nombres en la red de la UdL
    - *09_instalacion_nfsroot_fai*
        - *09_instalacion_nfsroot_fai.txt* – Explicación de la creación del nfsroot de FAI
    - *resumen_laboratorio2.txt* – Resumen de todos los pasos necesarios para llevar a cabo la configuración de laboratorio 2
    - *script_automatico_laboratorio2.sh* – Script que tras una instalación de Ubuntu permite configurar el servidor como la configuración de laboratorio 2 a partir de los parámetros encontrados en el mismo script y de la tabla proporcionada por el fichero *../../scripts_utiles/tabla_base.txt*

- *configuracion_previa/virtualbox* – Contiene todos los archivos necesarios para cumplir los requisitos necesarios para la configuración virtualbox
    - *01_distribucion_ubuntu_8.10*
        - *01_distribucion_ubuntu_8.10.txt* – Explicación de cómo instalar Ubuntu
    - *02_configuracion_de_paquetes_propios*
        - *02_configuracion_de_paquetes_propios.txt* – Explicación de cómo configurar los paquetes propios
        - *etc/apt/sources.list* – Fichero de configuración de repositorios propuesto para el servidor
        - *03_configuracion_preferencia_paquetes*
            - *03_configuracion_preferencia_paquetes.txt* – Explicación de cómo hacer preferentes nuestros paquetes
            - *etc/apt/preferences* – Fichero que establece la preferencia de la sección codip2p sobre las demás
        - *04_instalacion_de_fai_y_otros_paquetes*
            - *04_instalacion_de_fai_y_otros_paquetes.txt* – Explicación de la instalación de FAI y otros paquetes necesarios
        - *05_configuracion_fai_proyecto_codip2p*
            - *05_configuracion_fai_proyecto_codip2p.txt* – Explicación de la adaptación de la configuración de FAI para el proyecto codip2p
            - *etc/fai/apt/sources.list* – Fichero de configuración de repositorios propuesto para los nodos
            - etc/fai/*fai.conf* – Fichero de configuración de FAI propuesto adaptado para el proyecto codip2p
            - *etc/fai/make-fai-nfsroot.conf* – Fichero de configuración del nfsroot de FAI configurado para usar el repositorio del servidor

        - ◇ *srv/fai/config/scripts/last/50-misc* – Hook que permite el montado del repositorio de paquetes por nfs en los nodos al reinicio de la instalación
      - ○ *06_configuracion_sistema_restante*
        - ◇ *06_configuracion_sistema_restante.txt* – Explicación de otras configuraciones necesarias
        - ◇ *etc/dhcp3/dhcpd.conf* – Fichero de configuración de la asignación de ips a MACs del demonio DHCP
        - ◇ *etc/exports* – Fichero de configuración del demonio nfs
        - ◇ *etc/hosts* – Fichero de configuración que establece las relaciones entre nombres de máquinas y sus ips a nivel local del servidor
        - ◇ *etc/inetd.conf* – Fichero de configuración del superservidor inetd para arranque automático del demonio tftpd
      - ○ *07_configuracion_red*
        - ◇ *07_configuracion_red.txt* – Explicación de la configuración de la red
        - ◇ *etc/network/interfaces* – Fichero de configuración de las interfaces de red propuesto para realizar un bridge con la red de virtualbox
      - ○ *08_configuracion_virtualbox*
        - ◇ *08_configuracion_virtualbox.txt* – Explicación de las configuraciones adicionales en el programa Virtualbox
        - ◇ *etc/vbox/interfaces* – Fichero de configuración de las interfaces virtuales de Virtualbox
      - ○ *09_instalacion_nfsroot_fai*
        - ◇ *09_instalacion_nfsroot_fai.txt* – Explicación de la creación del nfsroot de FAI
    - *resumen_virtualbox.txt* – Resumen de todos los pasos necesarios para llevar a cabo la configuración virtualbox

- *repositorio*
  - *configuracion*
    - *distributions* – Configuración del repositorio para el programa reprepro
  - *repositorio_paquetes_desdeslin.tar.gz* – Repositorio con binarios y fuentes de los paquetes modificados dentro de un archivo de tipo tar comprimido mediante gzip

- *scripts_utiles*
  - *dhcp3_server_default_generator.sh* – Script generador del fichero de configuración general del demonio DHCP
  - *dhcpd_generator.sh* – *S*cript generador del fichero de configuración de la asignación de ips a MACs del demonio DHCP. Los parámetros los recoge del fichero *tabla_base.txt*

- *etc_hosts_generator.sh* – Script generador del fichero de configuración que establece las relaciones entre nombres de máquinas y sus ips a nivel local del servidor. Los parámetros los recoge del fichero *tabla_base.txt*
- *interfaces_generator.sh* – Script generador del fichero de configuración de las interfaces de red
- *multiple_host_awake.sh* – Script para iniciar por red (wake-on-lan) los nodos del cluster. Los parámetros los recoge del fichero *tabla_base.txt*
- *server_codip2p_repository_sources_list_generator.sh* – Script generador de las lineas sugeridas para el fichero de configuración de repositorios del servidor
- *tabla_base.txt* – Fichero de configuración de algunos de los scripts propuestos. En él se especifican: Nombre de host, ip y MAC de cada uno de los nodos

# Apéndice B

# Código fuente de Desdeslin

## B.1. Ficheros modificados del paquete FAI

### B.1.1. conf/make-fai-nfsroot.conf

```
# these variables are only used by make-fai-nfsroot(8)
# here you can use also variables defined in fai.conf
# directory on the install server where the nfsroot for
FAI is
# created, approx size: 330MB, also defined in bootptab
or dhcp.conf
NFSROOT=/srv/fai/nfsroot
# TFTP root directory
TFTPROOT=/srv/tftp/fai
# Add a line for mirrorhost and installserver when DNS
is not available
# on the clients. This line(s) will be added to
$nfsroot/etc/hosts.
#NFSROOT_ETC_HOSTS="192.168.1.250 yourinstallserver"
FAI_DEBOOTSTRAP="intrepid
http://archive.ubuntu.com/ubuntu"
# the encrypted (with md5 or crypt) root password on all
install clients during
# installation process; used when log in via ssh;
default pw is: fai
FAI_ROOTPW='$1$kBnWc0.E$djxB128U7dMkrltJHPf6d1'
# location of a identity.pub file; this user can log to
the install
# clients in as root without a password; only useful
with FAI_FLAGS="sshd"
#SSH_IDENTITY=/home/admin/.ssh/identity.pub
# directory of hooks to be sourced at the end of
make-fai-nfsroot,
# i.e they have to be shell scripts.
NFSROOT_HOOKS=/etc/fai/nfsroot-hooks/
# - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
- - - - - - -
# following lines should be read only for most of you
FAI_DEBOOTSTRAP_OPTS="--exclude=dhcp-client,info
--include=language-pack-en-base,aptitude,fontconfig,\
defoma,anthy,belocs-locales-bin"
```

### B.1.2. conf/sources.list

```
# These lines should work for installations of ubuntu intrepid
# please adapt the following entries if you are using
another distribution
# make sure that a recent version of live-initramfs is
available
# (e.g. feisty doesn't ship live-initramfs at all,
therefore the ppa is needed!)
# A more comprehensive example is at
/usr/share/doc/fai-doc/examples/etc
deb http://archive.ubuntu.com/ubuntu intrepid main
restricted universe multiverse
deb http://security.ubuntu.com/ubuntu intrepid-security
main restricted universe multiverse
deb http://ppa.launchpad.net/fai/ubuntu intrepid main
restricted universe multiverse
```

### B.1.3. debian/changelog

```
fai (3.2.4+svn4838-0ubuntu2) intrepid; urgency=low
  * Updated fai for serving Ubuntu intrepid
 -- Adrian Gibanel <adrian.gibanel.lopez@gmail.com>Tue,
06 Jan 2009 11:05:34 +0200
fai (3.2.4+svn4837-0ubuntu2) hardy; urgency=low
* upload to hardy
-- Reinhard Tartler <siretart@tauware.de>Tue, 01 Apr
2008 12:41:34 +0200
fai (3.2.4+svn4837-0ubuntu2~ppa3) hardy; urgency=low
* simple config GRUB/10-setup: add to kernel-img.conf
postinst and postrm
hook settings that run update-grub. Without them, grub's
menu.lst is not
updated on linux image installation/removal.
-- Achim Bohnet <allee@kubuntu.org>Mon, 31 Mar 2008
18:12:34 +0200
fai (3.2.4+svn4837-0ubuntu2~ppa2) hardy; urgency=low
* don't install menu.lst and GRUB postinst script at all
in the binary
package by removing them in the clean rule of
debian/rules
* remove console-data support, show how to preseed
console-setup in the
```

```
example class GERMAN. (LP: #207756)
-- Reinhard Tartler <siretart@tauware.de>Thu, 27 Mar
2008 15:52:45 +0100
fai (3.2.4+svn4837-0ubuntu2~ppa1) hardy; urgency=low
[ Christian Meier ]
* bin/fai, lib/get-config-dir-nfs: always use -n option
in romountopt to
prevent accessing /etc/mtab in read-only environment
(closes: LP: #207582)
* fix grub example (closes LP: #187659)
* remove menu.lst of examples
* move the postinst script of menu.lst to
examples/simple/scripts/GRUB/10-setup and modify it to
let update-grub
do the configuration of menu.lst
* workaround for not correctly working unionfs over nfs
* create nfshooks (/etc/fai/10-unionfs-workaround) that
delete certain
files in nfsroot that will be created automatically
through fai scripts.
(LP: #197006)
* add ghostscript to build dependencies
* don't fail build if there are no postinst scripts in
the 'simple-example'
configuration space
[ Reinhard Tartler ]
* add genisofs alternative to mkisofs in fai-server and
fai-quickstart
packages (LP: #156836)
```

### B.1.4. debian/fai-server.dirs

```
var/run/fai/make-fai-nfsroot
var/log/fai
etc/fai/apt
usr/bin
usr/sbin
usr/share/fai/pixmaps
etc/fai/nfsroot-hooks
```

### B.1.5. fai-server.install

```
sbin/fai-start-stop-daemon
usr/bin/faimond-gui
usr/sbin/fai-cd
usr/sbin/fai-chboot
usr/sbin/fai-setup
usr/sbin/faimond
usr/sbin/make-fai-nfsroot
usr/sbin/make-fai-bootfloppy
```

```
usr/bin/fai-mirror
usr/share/fai/pixmaps/*
etc/fai/menu.lst
etc/fai/NFSROOT
etc/fai/apt/sources.list
etc/fai/make-fai-nfsroot.conf
etc/fai/nfsroot-hooks/10-unionfs-workaround
```

### B.1.6. examples/simple/hooks/faiend.FAIBASE.source

```
#! /bin/bash
# Create /var/lock and /var/run in / folder.
#
# TODO: If folders do exist do not create them.
VAR_MOUNT_TEMPORAL_FOLDER="/mnt/tmp"
VAR_LOCK="/var/lock"
VAR_RUN="/var/run"
TARGET_FOLDER="/target"
mkdir ${VAR_MOUNT_TEMPORAL_FOLDER}
mount -o bind ${TARGET_FOLDER}
${VAR_MOUNT_TEMPORAL_FOLDER}
mkdir -p ${VAR_MOUNT_TEMPORAL_FOLDER}${VAR_RUN}
mkdir -p ${VAR_MOUNT_TEMPORAL_FOLDER}${VAR_LOCK}
umount ${VAR_MOUNT_TEMPORAL_FOLDER}
rmdir  ${VAR_MOUNT_TEMPORAL_FOLDER}
sync
echo "/var/lock and /var/run creation - END"
```

## B.2. Ficheros modificados del paquete initramfs-tools

### B.2.1. debian/changelog

```
initramfs-tools (0.92bubuntu17) intrepid; urgency=low
* Adapt initramfs-tools so that Fully Automatic
Installation serves an Ubuntu 8.10 ok.
 -- Adrian Gibanel <adrian.gibanel.lopez@gmail.com>Wed,
03 Jan 2009 01:50:37 +0100
initramfs-tools (0.92bubuntu15) intrepid; urgency=low
[ Dan Munckton ]
* Revert "framebuffer: Let udev create fb devices." udev
isn't started at
this point and therefore can't create framebuffer
devices. This causes
usplash not to run on PS3 (LP: #274860).
-- Colin Watson <cjwatson@ubuntu.com>Wed, 01 Oct 2008
16:02:37 +0100
initramfs-tools (0.92bubuntu14) intrepid; urgency=low
```

```
* scripts/functions: Call ipconfig with a one-minute
timeout rather than
waiting forever (LP: #182940).
-- Colin Watson <cjwatson@ubuntu.com>Wed, 01 Oct 2008
14:51:59 +0100
initramfs-tools (0.92bubuntu13) intrepid; urgency=low
* scripts/local-premount/resume: Fix incorrect resume
message by checking
for swsusp signature before assuming there's a resume
about to happen
-- Matt Zimmerman <mdz@ubuntu.com>Tue, 16 Sep 2008
11:49:55 +0100
initramfs-tools (0.92bubuntu12) intrepid; urgency=low
* scripts/functions: must chvt(1) during failure hooks,
as console input
might be required (eg, boot degraded raid) (LP:
#268873).
-- Dustin Kirkland <kirkland@ubuntu.com>Thu, 11 Sep 2008
16:49:17 -0400
initramfs-tools (0.92bubuntu11) intrepid; urgency=low
* scripts/local-premount/resume: Display a message
indicating that a resume
has begun. This is non-localized text, but is better
than nothing until
we can display a graphical indicator. (LP: #41137)
-- Matt Zimmerman <mdz@ubuntu.com>Thu, 11 Sep 2008
09:26:31 +0100
```

### B.2.2. init

```
#!/bin/sh
echo "Loading, please wait..."
[ -d /dev ] || mkdir -m 0755 /dev
[ -d /root ] || mkdir -m 0700 /root
[ -d /sys ] || mkdir /sys
[ -d /proc ] || mkdir /proc
[ -d /tmp ] || mkdir /tmp
mkdir -p /var/lock
mount -t sysfs -o nodev,noexec,nosuid none /sys
mount -t proc -o nodev,noexec,nosuid none /proc
# Note that this only becomes /dev on the real
filesystem if udev's scripts
# are used; which they will be, but it's worth pointing
out
mount -t tmpfs -o mode=0755 udev /dev
[ -e /dev/console ] || mknod -m 0600 /dev/console c 5 1
[ -e /dev/null ] || mknod /dev/null c 1 3
>/dev/.initramfs-tools
mkdir /dev/.initramfs
# Export the dpkg architecture
```

```
export DPKG_ARCH=
. /conf/arch.conf
# Set modprobe env
export MODPROBE_OPTIONS="-Qb"
# Export relevant variables
export ROOT=
export ROOTDELAY=
export ROOTFLAGS=
export ROOTFSTYPE=
export break=
export init=/sbin/init
export quiet=n
export readonly=y
export rootmnt=/root
export debug=
export panic=
export blacklist=
export resume_offset=
# Bring in the main config
. /conf/initramfs.conf
for conf in conf/conf.d/*; do
[ -f ${conf} ] && . ${conf}
done
. /scripts/functions
# Parse command line options
for x in $(cat /proc/cmdline); do
case $x in
init=*)
init=${x#init=}
;;
root=*)
ROOT=${x#root=}
case $ROOT in
LABEL=*)
ROOT="/dev/disk/by-label/${ROOT#LABEL=}"
;;
UUID=*)
ROOT="/dev/disk/by-uuid/${ROOT#UUID=}"
;;
/dev/nfs)
[ -z "${BOOT}" ] && BOOT=nfs
;;
esac
;;
rootflags=*)
ROOTFLAGS="-o ${x#rootflags=}"
;;
rootfstype=*)
ROOTFSTYPE="${x#rootfstype=}"
;;
```

```
rootdelay=*)
ROOTDELAY="${x#rootdelay=}"
case ${ROOTDELAY} in
*[![:digit:].]*)
ROOTDELAY=
;;
esac
;;
resumedelay=*)
RESUMEDELAY="${x#resumedelay=}"
;;
loop=*)
LOOP="${x#loop=}"
;;
loopflags=*)
LOOPFLAGS="-o ${x#loopflags=}"
;;
loopfstype=*)
LOOPFSTYPE="${x#loopfstype=}"
;;
cryptopts=*)
cryptopts="${x#cryptopts=}"
;;
nfsroot=*)
NFSROOT="${x#nfsroot=}"
;;
netboot=*)
NETBOOT="${x#netboot=}"
;;
ip=*)
IPOPTS="${x#ip=}"
;;
boot=*)
BOOT=${x#boot=}
;;
resume=*)
RESUME="${x#resume=}"
;;
resume_offset=*)
resume_offset="${x#resume_offset=}"
;;
noresume)
noresume=y
;;
panic=*)
panic="${x#panic=}"
case ${panic} in
*[![:digit:].]*)
panic=
;;
```

```
esac
;;
quiet)
quiet=y
;;
ro)
readonly=y
;;
rw)
readonly=n
;;
debug)
debug=y
quiet=n
exec >/tmp/initramfs.debug 2>&1
set -x
;;
debug=*)
debug=y
quiet=n
set -x
;;
break=*)
break=${x#break=}
;;
break)
break=premount
;;
blacklist=*)
blacklist=${x#blacklist=}
;;
esac
done
if [ -z "${noresume}" ]; then
export resume=${RESUME}
else
export noresume
fi
depmod -a
maybe_break top
# export BOOT variable value for compcache,
# so we know if we run from casper
export BOOT
# Don't do log messages here to avoid confusing usplash
run_scripts /scripts/init-top
maybe_break modules
log_begin_msg "Loading essential drivers..."
load_modules
log_end_msg
maybe_break premount
```

```
[ "$quiet" != "y" ] && log_begin_msg "Running
/scripts/init-premount"
run_scripts /scripts/init-premount
[ "$quiet" != "y" ] && log_end_msg
maybe_break mount
log_begin_msg "Mounting root file system..."
. /scripts/${BOOT}
parse_numeric ${ROOT}
mountroot
log_end_msg
maybe_break bottom
[ "$quiet" != "y" ] && log_begin_msg "Running
/scripts/init-bottom"
run_scripts /scripts/init-bottom
[ "$quiet" != "y" ] && log_end_msg
# Move virtual filesystems over to the real filesystem
mount -n --move /sys ${rootmnt}/sys || mount -n -o move
/sys ${rootmnt}/sys
mount -n --move /proc ${rootmnt}/proc || mount -n -o
move /proc ${rootmnt}/proc
# Check init bootarg
if [ -n "${init}" ] && [ ! -x "${rootmnt}${init}" ];
then
echo "Target filesystem doesn't have ${init}."
init=
fi
# Search for valid init
if [ -z "${init}" ] ; then
for init in /sbin/init /etc/init /bin/init /bin/sh; do
if [ ! -x "${rootmnt}${init}" ]; then
continue
fi
break
done
fi
# No init on rootmount
if [ ! -x "${rootmnt}${init}" ]; then
panic "No init found. Try passing init= bootarg."
fi
# Confuses /etc/init.d/rc
if [ -n ${debug} ]; then
unset debug
fi
# Chain to real filesystem
maybe_break init
exec run-init ${rootmnt} ${init} "$@"
<${rootmnt}/dev/console >${rootmnt}/dev/console 2>&1
panic "Could not execute run-init."
```

### B.2.3. scripts/local

```
# Local filesystem mounting shell script
# Parameter: device node to check
# Echos fstype to stdout
# Return value: indicates if an fs could be recognized
get_fstype ()
{
local FS FSTYPE FSSIZE RET
FS="${1}"
# vol_id has a more complete list of file systems,
# but fstype is more robust
eval $(fstype "${FS}" 2>/dev/null)
if [ -z "${FSTYPE}" ]; then
FSTYPE="unknown"
fi
if [ "$FSTYPE" = "unknown" ] && [ -x /lib/udev/vol_id ];
then
FSTYPE=$(/lib/udev/vol_id -t "${FS}" 2>/dev/null)
fi
RET=$?
if [ -z "${FSTYPE}" ]; then
FSTYPE="unknown"
fi
echo "${FSTYPE}"
return ${RET}
}
root_missing()
{
ROOT="${1}"
[ ! -e "${ROOT}" ] || ! $(get_fstype "${ROOT}"
>/dev/null) || ! /sbin/udevadm settle
}
# Parameter: Where to mount the filesystem
mountroot ()
{
[ "$quiet" != "y" ] && log_begin_msg "Running
/scripts/local-top"
run_scripts /scripts/local-top
[ "$quiet" != "y" ] && log_end_msg
# If the root device hasn't shown up yet, give it a
little while
# to deal with removable devices
while root_missing "${ROOT}"; do
log_begin_msg "Waiting for root file system..."
# Default delay is 30s
if [ -z "${ROOTDELAY}" ]; then
slumber=30
else
slumber=${ROOTDELAY}
```

```
fi
if [ -x /sbin/usplash_write ]; then
/sbin/usplash_write "TIMEOUT ${slumber}" || true
fi
slumber=$(( ${slumber} * 10 ))
while root_missing "${ROOT}"; do
/bin/sleep 0.1
slumber=$(( ${slumber} - 1 ))
[ ${slumber} -gt 0 ] || break
done
if [ ${slumber} -gt 0 ]; then
log_end_msg 0
else
log_end_msg 1 || true
fi
if [ -x /sbin/usplash_write ]; then
/sbin/usplash_write "TIMEOUT 15" || true
fi
# Run failure hooks, hoping one of them can fix up the
system
# and we can restart the wait loop. If they all fail,
abort
# and move on to the panic handler and shell.
if root_missing "${ROOT}" && ! try_failure_hooks; then
break
fi
done
# We've given up, but we'll let the user fix matters if
they can
while root_missing "${ROOT}"; do
# give hint about renamed root
case "${ROOT}" in
/dev/hd*)
suffix="${ROOT#/dev/hd}"
major="${suffix%[[:digit:]]}"
major="${major%[[:digit:]]}"
if [ -d "/sys/block/sd${major}" ]; then
echo "WARNING bootdevice may be renamed. Try
root=/dev/sd${suffix}"
fi
;;
/dev/sd*)
suffix="${ROOT#/dev/sd}"
major="${suffix%[[:digit:]]}"
major="${major%[[:digit:]]}"
if [ -d "/sys/block/hd${major}" ]; then
echo "WARNING bootdevice may be renamed. Try
root=/dev/hd${suffix}"
fi
;;
```

```
esac
echo "Gave up waiting for root device. Common problems:"
echo " - Boot args (cat /proc/cmdline)"
echo " - Check rootdelay= (did the system wait long
enough?)"
echo " - Check root= (did the system wait for the right
device?)"
echo " - Missing modules (cat /proc/modules; ls /dev)"
panic "ALERT! ${ROOT} does not exist. Dropping to a
shell!"
done
# Get the root filesystem type if not set
if [ -z "${ROOTFSTYPE}" ]; then
FSTYPE=$(get_fstype "${ROOT}")
else
FSTYPE=${ROOTFSTYPE}
fi
[ "$quiet" != "y" ] && log_begin_msg "Running
/scripts/local-premount"
run_scripts /scripts/local-premount
[ "$quiet" != "y" ] && log_end_msg
if [ ${readonly} = y ] && \
[ -z "$LOOP" ]; then
roflag=-r
else
roflag=-w
fi
# FIXME This has no error checking
modprobe ${FSTYPE}
# FIXME This has no error checking
# Mount root
mount ${roflag} -t ${FSTYPE} ${ROOTFLAGS} ${ROOT}
${rootmnt}
mountroot_status="$?"
if [ "$LOOP" ]; then
if [ "$mountroot_status" != 0 ]; then
if [ ${FSTYPE} = ntfs ] || [ ${FSTYPE} = vfat ]; then
panic "
Could not mount the partition ${ROOT}.
This could also happen if the file system is not clean
because of an operating
system crash, an interrupted boot process, an improper
shutdown, or unplugging
of a removable device without first unmounting or
ejecting it. To fix this,
simply reboot into Windows, let it fully start, log in,
run 'chkdsk /r', then
gracefully shut down and reboot back into Windows. After
this you should be
able to reboot again and resume the installation.
```

```
(filesystem = ${FSTYPE}, error code = $mountroot_status)
"
fi
fi
mkdir -p /host
mount --move ${rootmnt} /host || mount -o move
${rootmnt} /host
while [ ! -e "/host/${LOOP#/}" ]; do
panic "ALERT! /host/${LOOP#/} does not exist. Dropping
to a shell!"
done
# Get the loop filesystem type if not set
if [ -z "${LOOPFSTYPE}" ]; then
eval $(fstype <"/host/${LOOP#/}")
else
FSTYPE="${LOOPFSTYPE}"
fi
if [ "$FSTYPE" = "unknown" ] && [ -x /lib/udev/vol_id ];
then
FSTYPE=$(/lib/udev/vol_id -t "/host/${LOOP#/}")
[ -z "$FSTYPE" ] && FSTYPE="unknown"
fi
if [ ${readonly} = y ]; then
roflag=-r
else
roflag=-w
fi
# FIXME This has no error checking
modprobe loop
modprobe ${FSTYPE}
# FIXME This has no error checking
mount ${roflag} -o loop -t ${FSTYPE} ${LOOPFLAGS}
"/host/${LOOP#/}" ${rootmnt}
if [ -d ${rootmnt}/host ]; then
mount --move /host ${rootmnt}/host || mount -o move
/host ${rootmnt}/host
fi
fi
[ "$quiet" != "y" ] && log_begin_msg "Running
/scripts/local-bottom"
run_scripts /scripts/local-bottom
[ "$quiet" != "y" ] && log_end_msg
}
```

## B.3. Ficheros modificados del paquete live-initramfs

### B.3.1. hooks/live

```
#!/bin/sh
```

```
# initramfs hook for live-initramfs (Debian Live)
set -e
# initramfs-tools header
PREREQ=""
prereqs()
{
echo "${PREREQ}"
}
case "${1}" in
prereqs)
prereqs
exit 0
;;
esac
. /usr/share/initramfs-tools/hook-functions
# live-initramfs hook
# Handling live-initramfs
# Configuration
if [ -r /etc/live.conf ]
then
. /etc/live.conf
mkdir -p "${DESTDIR}"/etc
cp /etc/live.conf "${DESTDIR}"/etc
fi
# Directories
mkdir -p "${DESTDIR}"/lib/live-initramfs
# Executables
copy_exec /usr/share/live-initramfs/live-reconfigure
/bin
copy_exec /usr/share/live-initramfs/live-preseed /bin
# Scripts
cp /usr/share/initramfs-tools/scripts/live-functions
"${DESTDIR}"/scripts
cp /usr/share/initramfs-tools/scripts/live-helpers
"${DESTDIR}"/scripts
# klibc dependencies
for FILE in /lib/libacl* /lib/libblkid* /lib/libuuid*
/lib/libdevmapper* /lib/libattr*
do
if [ ! -e "${DESTDIR}"/"${FILE}" ]
then
cp -a "${FILE}" "${DESTDIR}"/"${FILE}"
fi
done
# Handling other stuff
# Configuration: keymap (usefull when using encryption)
if [ -x /bin/loadkeys ] && [ -r
/etc/console/boottime.kmap.gz ]
then
copy_exec /bin/loadkeys /bin
```

```
mkdir -p "${DESTDIR}"/etc
cp /etc/console/boottime.kmap.gz "${DESTDIR}"/etc
fi
# Configuration: Unique ID
if [ -n "${LIVE_GENERATE_UUID}" ]
then
mkdir -p "${DESTDIR}"/conf
uuidgen -r >"${DESTDIR}"/conf/uuid.conf
fi
# Filesystem: cifs
if [ -x /sbin/mount.cifs ]
then
copy_exec /sbin/mount.cifs /sbin
manual_add_modules cifs
fi
# Filesystem: ext3
manual_add_modules ext3
# Filesystem: jffs2
manual_add_modules jffs2
# Filesystem: squashfs
copy_exec /sbin/losetup /sbin
manual_add_modules loop
manual_add_modules squashfs
manual_add_modules sqlzma
manual_add_modules unlzma
# Filesystem: unionfs/aufs
manual_add_modules unionfs
manual_add_modules aufs
# Filesystem: vfat
manual_add_modules nls_cp437
manual_add_modules nls_iso8859-1
manual_add_modules nls_utf8
manual_add_modules vfat
# Hardware: cdrom
manual_add_modules ide-cd
manual_add_modules ide-generic
manual_add_modules ohci1394
manual_add_modules sbp2
manual_add_modules sr_mod
# Hardware: usb
manual_add_modules sd_mod
# Hardware: network
auto_add_modules net
# Program: eject
if [ -x /usr/bin/eject ]
then
copy_exec /usr/bin/eject /bin
fi
# Program: md5sum
copy_exec /usr/bin/md5sum /bin
```

```
# Program: udev
copy_exec /sbin/udevtrigger /sbin
copy_exec /sbin/udevsettle /sbin
copy_exec /usr/bin/udevinfo /bin
# Program: wget
if [ -x /usr/bin/wget ]
then
copy_exec /usr/bin/wget /bin
fi
# Program: wc
copy_exec /usr/bin/wc /bin
```

### B.3.2. scripts/live

```
#!/bin/sh
# set -e
export PATH="/root/usr/bin:/root/usr/sbin:/root/bin:\
/root/sbin:/usr/bin:/usr/sbin:/bin:/sbin"
echo "/root/lib" >>/etc/ld.so.conf
echo "/root/usr/lib" >>/etc/ld.so.conf
mountpoint="/live/image"
LIVE_MEDIA_PATH="live"
root_persistence="live-rw"
home_persistence="home-rw"
root_snapshot_label="live-sn"
home_snapshot_label="home-sn"
USERNAME="user"
USERFULLNAME="Live user"
HOSTNAME="host"
mkdir -p "${mountpoint}"
[ -f /etc/live.conf ] && . /etc/live.conf
export USERNAME USERFULLNAME HOSTNAME
. /scripts/live-helpers
if [ ! -f /live.vars ]
then
touch /live.vars
fi
Arguments ()
{
PRESEEDS=""
for ARGUMENT in $(cat /proc/cmdline)
do
case "${ARGUMENT}" in
access=*)
ACCESS="${ARGUMENT#access=}"
export ACCESS
;;
console=*)
DEFCONSOLE="${ARGUMENT#*=}"
export DEFCONSOLE
```

```
;;
debug)
DEBUG="Yes"
export DEBUG
set -x
;;
fetch=*)
FETCH="${ARGUMENT#fetch=}"
export FETCH
;;
hook=*)
HOOK="${ARGUMENT#hook=}"
export HOOK
;;
hostname=*)
HOSTNAME="${ARGUMENT#hostname=}"
LIVECONF="changed"
export HOSTNAME LIVECONF
;;
username=*)
USERNAME="${ARGUMENT#username=}"
LIVECONF="changed"
export USERNAME LIVECONF
;;
userfullname=*)
USERFULLNAME="${ARGUMENT#userfullname=}"
LIVECONF="changed"
export USERFULLNAME LIVECONF
;;
ignore_uuid)
IGNORE_UUID="Yes"
export IGNORE_UUID
;;
ip=*)
STATICIP="${ARGUMENT#ip=}"
if [ -z "${STATICIP}" ]
then
STATICIP="frommedia"
fi
export STATICIP
;;
keyb=*|kbd-chooser/method=*)
KBD="${ARGUMENT#*=}"
export KBD
;;
klayout=*|console-setup/layoutcode=*)
KLAYOUT="${ARGUMENT#*=}"
export KLAYOUT
;;
kvariant=*|console-setup/variantcode=*)
```

```
KVARIANT="${ARGUMENT#*=}"
export KVARIANT
;;
kmodel=*|console-setup/modelcode=*)
KMODEL="${ARGUMENT#*=}"
export KMODEL
;;
koptions=*)
KOPTIONS="${ARGUMENT#koptions=}"
export KOPTIONS
;;
live-getty)
LIVE_GETTY="1"
export LIVE_GETTY
;;
live-media=*|bootfrom=*)
LIVE_MEDIA="${ARGUMENT#*=}"
export LIVE_MEDIA
;;
live-media-encryption=*|encryption=*)
LIVE_MEDIA_ENCRYPTION="${ARGUMENT#*=}"
export LIVE_MEDIA_ENCRYPTION
;;
live-media-offset=*)
LIVE_MEDIA_OFFSET="${ARGUMENT#live-media-offset=}"
export LIVE_MEDIA_OFFSET
;;
live-media-path=*)
LIVE_MEDIA_PATH="${ARGUMENT#live-media-path=}"
export LIVE_MEDIA_PATH
;;
live-media-timeout=*)
LIVE_MEDIA_TIMEOUT="${ARGUMENT#live-media-timeout=}"
export LIVE_MEDIA_TIMEOUT
;;
language=*|debian-installer/language=*)
language=${x#debian-installer/language=}
locale="$(lang2locale "$language")"
set_locale="true"
;;
locale=*|debian-installer/locale=*)
LOCALE="${ARGUMENT#*=}"
export LOCALE
;;
module=*)
MODULE="${ARGUMENT#module=}"
export MODULE
;;
netboot=*)
NETBOOT="${ARGUMENT#netboot=}"
```

```
export NETBOOT
;;
nfsopts=*)
NFSOPTS="${ARGUMENT#nfsopts=}"
export NFSOPTS
;;
nfscow=*)
NFS_COW="${ARGUMENT#nfscow=}"
export NFS_COW
;;
noaccessibility)
NOACCESSIBILITY="Yes"
export NOACCESSIBILITY
;;
noapparmor)
NOAPPARMOR="Yes"
export NOAPPARMOR
;;
noaptcdrom)
NOAPTCDROM="Yes"
export NOAPTCDROM
;;
noautologin)
NOAUTOLOGIN="Yes"
export NOAUTOLOGIN
;;
noxautologin)
NOXAUTOLOGIN="Yes"
export NOXAUTOLOGIN
;;
noconsolekeyboard)
NOCONSOLEKEYBOARD="Yes"
export NOCONSOLEKEYBOARD
;;
nofastboot)
NOFASTBOOT="Yes"
export NOFASTBOOT
;;
nofstab)
NOFSTAB="Yes"
export NOFSTAB
;;
nognomepanel)
NOGNOMEPANEL="Yes"
export NOGNOMEPANEL
;;
nohosts)
NOHOSTS="Yes"
export NOHOSTS
;;
```

```
nokpersonalizer)
NOKPERSONALIZER="Yes"
export NOKPERSONALIZER
;;
nolanguageselector)
NOLANGUAGESELECTOR="Yes"
export NOLANGUAGESELECTOR
;;
nolocales)
NOLOCALES="Yes"
export NOLOCALES
;;
nonetworking)
NONETWORKING="Yes"
export NONETWORKING
;;
nopowermanagement)
NOPOWERMANAGEMENT="Yes"
export NOPOWERMANAGEMENT
;;
noprogramcrashes)
NOPROGRAMCRASHES="Yes"
export NOPROGRAMCRASHES
;;
norestrictedmanager)
NORESTRICTEDMANAGER="Yes"
export NORESTRICTEDMANAGER
;;
nosudo)
NOSUDO="Yes"
export NOSUDO
;;
noswap)
NOSWAP="Yes"
export NOSWAP
;;
noupdatenotifier)
NOUPDATENOTIFIER="Yes"
export NOUPDATENOTIFIER
;;
nouser)
NOUSER="Yes"
export NOUSER
;;
noxautoconfig)
NOXAUTOCONFIG="Yes"
export NOXAUTOCONFIG
;;
noxscreensaver)
NOXSCREENSAVER="Yes"
```

```
export NOXSCREENSAVER
;;
persistent)
PERSISTENT="Yes"
export PERSISTENT
;;
nopersistent)
NOPERSISTENT="Yes"
export NOPERSISTENT
;;
preseed/file=*|file=*)
LOCATION="${ARGUMENT#*=}"
export LOCATION
;;
nopreseed)
NOPRESEED="Yes"
export NOPRESEED
;;
url=*)
location="${ARGUMENT#url=}"
mount -n --bind /sys /root/sys || mount -n -o bind /sys
/root/sys
 mount -n --bind /proc /root/proc || mount -n -o bind
/proc /root/proc
 mount -n --bind /dev /root/dev || mount -n -o bind /dev
/root/dev
mkdir -p /root/var/run/network
chroot /root dhclient eth0
chroot /root wget -P /tmp "${location}"
chroot /root ifconfig eth0 down
umount /root/sys
umount /root/proc
umount /root/dev
LOCATION="/tmp/$(basename "${location}")"
;;
*/*=*)
question="${ARGUMENT%%=*}"
value="${ARGUMENT#*=}"
PRESEEDS="${PRESEEDS}\"${question}=${value}\" "
export PRESEEDS
;;
showmounts)
SHOWMOUNTS="Yes"
export SHOWMOUNTS
;;
textonly)
TEXTONLY="Yes"
export TEXTONLY
;;
timezone=*)
```

```
TIMEZONE="${ARGUMENT#timezone=}"
export TIMEZONE
;;
notimezone)
NOTIMEZONE="Yes"
export NOTIMEZONE
;;
todisk=*)
TODISK="${ARGUMENT#todisk=}"
export TODISK
;;
toram)
TORAM="Yes"
export TORAM
;;
toram=*)
TORAM="Yes"
MODULETORAM="${ARGUMENT#toram=}"
export TORAM MODULETORAM
;;
exposedroot)
EXPOSED_ROOT="Yes"
export EXPOSED_ROOT
;;
plainroot)
PLAIN_ROOT="Yes"
export PLAIN_ROOT
;;
root=*)
ROOT="${ARGUMENT#root=}"
export ROOT
;;
union=*)
UNIONTYPE="${ARGUMENT#union=}"
export UNIONTYPE
;;
utc=*)
UTC="${ARGUMENT#utc=}"
export UTC
;;
xdebconf)
XDEBCONF="Yes"
export XDEBCONF
;;
xdriver=*)
XDRIVER="${ARGUMENT#xdriver=}"
export XDRIVER
;;
xvideomode=*)
XVIDEOMODE="${ARGUMENT#xvideomode=}"
```

```
export XVIDEOMODE
;;
esac
done
# sort of compatibility with netboot.h from linux docs
if [ -z "${NETBOOT}" ]
then
if [ "${ROOT}" = "/dev/nfs" ]
then
NETBOOT="nfs"
export NETBOOT
elif [ "${ROOT}" = "/dev/cifs" ]
then
NETBOOT="cifs"
export NETBOOT
fi
fi
if [ -z "${MODULE}" ]
then
MODULE="filesystem"
export MODULE
fi
if [ -z "${UNIONTYPE}" ]
then
UNIONTYPE="unionfs"
export UNIONTYPE
fi
}
is_live_path ()
{
DIRECTORY="${1}"
if [ -d "${DIRECTORY}"/"${LIVE_MEDIA_PATH}" ]
then
for FILESYSTEM in squashfs ext2 ext3 xfs dir jffs2
do
if [ "$(echo
${DIRECTORY}/${LIVE_MEDIA_PATH}/*.${FILESYSTEM})" !=
"${DIRECTORY}/${LIVE_MEDIA_PATH}/*.${FILESYSTEM}" ]
then
return 0
fi
done
fi
return 1
}
matches_uuid ()
{
if [ "${IGNORE_UUID}" ] || [ ! -e /conf/uuid.conf ]
then
return 0
```

```
fi
path="${1}"
uuid="$(cat /conf/uuid.conf)"
for try_uuid_file in "${mountpoint}/.disk/casper-uuid"*
do
[ -e "${try_uuid_file}" ] || continue
try_uuid="$(cat "${try_uuid_file}")"
if [ "${uuid}" = "${try_uuid}" ]
then
return 0
fi
done
return 1
}
get_backing_device ()
{
case "${1}" in
*.squashfs|*.ext2|*.ext3|*.jffs2)
echo $(setup_loop "${1}" "loop" "/sys/block/loop*" '0'
"${LIVE_MEDIA_ENCRYPTION}" "${2}")
;;
*.dir)
echo "directory"
;;
*)
panic "Unrecognized live filesystem: ${1}"
;;
esac
}
match_files_in_dir ()
{
# Does any files match pattern ${1} ?
local pattern="${1}"
if [ "$(echo ${pattern})" != "${pattern}" ]
then
return 0
fi
return 1
}
mount_images_in_directory ()
{
directory="${1}"
rootmnt="${2}"
mac="${3}"
if match_files_in_dir
"${directory}/${LIVE_MEDIA_PATH}/*.squashfs" ||
match_files_in_dir
"${directory}/${LIVE_MEDIA_PATH}/*.ext2" ||
match_files_in_dir
"${directory}/${LIVE_MEDIA_PATH}/*.ext3" ||
```

```
match_files_in_dir
"${directory}/${LIVE_MEDIA_PATH}/*.jffs2" ||
match_files_in_dir
"${directory}/${LIVE_MEDIA_PATH}/*.dir"
then
[ -n "${mac}" ] &&
adddirectory="${directory}/${LIVE_MEDIA_PATH}/${mac}"
setup_unionfs "${directory}/${LIVE_MEDIA_PATH}"
"${rootmnt}" "${adddirectory}"
else
:
fi
}
is_nice_device ()
{
sysfs_path="${1#/sys}"
if /lib/udev/path_id "${sysfs_path}" | grep -E -q
"ID_PATH=(usb|pci-[^-]*-[ide|scsi|usb])"
then
return 0
fi
return 1
}
copy_live_to ()
{
copyfrom="${1}"
copytodev="${2}"
copyto="${copyfrom}_swap"
if [ -z "${MODULETORAM}" ]
then
size=$(fs_size "" ${copyfrom} "used")
else
MODULETORAMFILE="${copyfrom}/${LIVE_MEDIA_PATH}/${MODULETORAM}"
if [ -f "${MODULETORAMFILE}" ]
then
size=$( expr $(ls -la ${MODULETORAMFILE} | awk '{print
$5}') / 1024 + 5000 )
else
log_warning_msg "Error: toram-module ${MODULETORAM}
(${MODULETORAMFILE}) could not be read."
return 1
fi
fi
if [ "${copytodev}" = "ram" ]
then
# copying to ram:
freespace=$( expr $(awk '/MemFree/{print $2}'
/proc/meminfo) + $( cat /proc/meminfo | grep Cached |
head -n 1 | awk '/Cached/{print $2}' - ) )
mount_options="-o size=${size}k"
```

```
free_string="memory"
fstype="tmpfs"
dev="/dev/shm"
else
# it should be a writable block device
if [ -b "${copytodev}" ]
then
dev="${copytodev}"
free_string="space"
fstype=$(get_fstype "${dev}")
freespace=$(fs_size "${dev}")
else
[ "$quiet" != "y" ] && log_warning_msg "${copytodev} is
not a block device."
return 1
fi
fi
if [ "${freespace}" -lt "${size}" ]
then
[ "${quiet}" != "y" ] && log_warning_msg "Not enough
free ${free_string} (${freespace}k free, ${size}k
needed) to copy live media in ${copytodev}."
return 1
fi
# begin copying (or uncompressing)
mkdir "${copyto}"
echo "mount -t ${fstype} ${mount_options} ${dev}
${copyto}"
mount -t "${fstype}" ${mount_options} "${dev}"
"${copyto}"
if [ "${extension}" = "tgz" ]
then
cd "${copyto}"
tar zxf "${copyfrom}/${LIVE_MEDIA_PATH}/$(basename
${FETCH})"
rm -f "${copyfrom}/${LIVE_MEDIA_PATH}/$(basename
${FETCH})"
mount -r --move "${copyto}" "${rootmnt}" || mount -r -o
move "${copyto}" "${rootmnt}"
cd "${OLDPWD}"
else
if [ -n "${MODULETORAMFILE}" ]
then
cp ${MODULETORAMFILE} ${copyto} # copy only the
filesystem module
else
cp -a ${copyfrom}/* ${copyto} # "cp -a" from busybox
also copies hidden files
fi
livefs_root
```

```
umount ${copyfrom}
mount -r --move ${copyto} ${copyfrom} || mount -r -o
move ${copyto} ${copyfrom}
fi
rmdir ${copyto}
return 0
}
do_netmount ()
{
rc=1
modprobe -q af_packet # For DHCP
udevtrigger
udevsettle
ipconfig ${DEVICE} | tee /netboot.config
# source relevant ipconfig output
OLDHOSTNAME=${HOSTNAME}
. /tmp/net-${DEVICE}.conf
[ -z ${HOSTNAME} ] && HOSTNAME=${OLDHOSTNAME}
export HOSTNAME
if [ "${NFSROOT}" = "auto" ]
then
NFSROOT=${ROOTSERVER}:${ROOTPATH}
fi
if [ -n "${FETCH}" ] && do_httpmount
then
rc=0
return ${rc}
fi
if [ "${NFSROOT#*:}" = "${NFSROOT}" ] && [ "$NETBOOT" !=
"cifs" ]
then
NFSROOT=${ROOTSERVER}:${NFSROOT}
fi
[ "${quiet}" != "y" ] && log_begin_msg "Trying netboot
from ${NFSROOT}"
if [ "${NETBOOT}" != "nfs" ] && do_cifsmount
then
rc=0
elif do_nfsmount
then
NETBOOT="nfs"
export NETBOOT
rc=0
fi
[ "${quiet}" != "y" ] && log_end_msg
return ${rc}
}
do_httpmount ()
{
rc=1
```

```
extension=$(echo "${FETCH}" | sed
's/\(.*\)\.\(.*\)/\2/')
case "${extension}" in
squashfs|tgz|tar)
[ "${quiet}" != "y" ] && log_begin_msg "Trying wget
${FETCH} -O ${mountpoint}/$(basename ${FETCH})"
mkdir -p "${mountpoint}/${LIVE_MEDIA_PATH}"
wget "${FETCH}" -O
"${mountpoint}/${LIVE_MEDIA_PATH}/$(basename ${FETCH})"
[ ${?} -eq 0 ] && rc=0
[ "${extension}" = "tgz" ] && live_dest="ram"
;;
*)
[ "${quiet}" != "y" ] && log_begin_msg "Unrecognized
archive extension for ${FETCH}"
esac
return ${rc}
}
do_nfsmount ()
{
rc=1
modprobe -q nfs
if [ -z "${NFSOPTS}" ]
then
NFSOPTS=""
fi
[ "${quiet}" != "y" ] && log_begin_msg "Trying nfsmount
-o nolock -o ro ${NFSOPTS} ${NFSROOT} ${mountpoint}"
# FIXME: This for loop is an ugly HACK round an nfs bug
for i in 0 1 2 3 4 5 6 7 8 9 a b c d e f 10 11 12 13
do
nfsmount -o nolock -o ro ${NFSOPTS} "${NFSROOT}"
"${mountpoint}" && rc=0 && break
sleep 1
done
return ${rc}
}
do_cifsmount ()
{
rc=1
if [ -x "/sbin/mount.cifs" ]
then
if [ -z "${NFSOPTS}" ]
then
CIFSOPTS="-ouser=root,password="
else
CIFSOPTS="${NFSOPTS}"
fi
[ "${quiet}" != "y" ] && log_begin_msg "Trying
mount.cifs ${NFSROOT} ${mountpoint} ${CIFSOPTS}"
```

```
modprobe -q cifs
if mount.cifs "${NFSROOT}" "${mountpoint}" "${CIFSOPTS}"
then
rc=0
fi
fi
return ${rc}
}
do_snap_copy ()
{
fromdev="${1}"
todir="${2}"
snap_type="${3}"
size=$(fs_size "${fromdev}" "" "used")
if [ -b "${fromdev}" ]
then
# look for free mem
if [ -n "${HOMEMOUNTED}" -a "${snap_type}" = "HOME" ]
then
todev=$(cat /proc/mounts | grep -s " $(base_path
${todir}) " | awk '{print $1}' )
freespace=$(df -k | grep -s ${todev} | awk '{print $4}')
else
freespace=$( expr $(awk '/MemFree/{print $2}'
/proc/meminfo) + $( cat /proc/meminfo | grep Cached |
head -n 1 | awk '/Cached/{print $2}' - ))
fi
tomount="/mnt/tmpsnap"
if [ ! -d "${tomount}" ]
then
mkdir -p "${tomount}"
fi
fstype=$(get_fstype "${fromdev}")
if [ -n "${fstype}" ]
then
# Copying stuff...
mount -t "${fstype}" -o ro,noatime "${fromdev}"
"${tomount}"
cp -a "${tomount}"/* ${todir}
umount "${tomount}"
else
log_warning_msg "Unrecognized fstype: ${fstype} on
${fromdev}:${snap_type}"
fi
rmdir "${tomount}"
if echo ${fromdev} | grep -qs loop
then
losetup -d "${fromdev}"
fi
return 0
```

```
else
return 1
[ "${quiet}" != "y" ] && log_warning_msg "Unable to find
the snapshot ${snap_type} medium"
fi
}
try_snap ()
{
# Look for ${snap_label}.* in block devices and copy the
contents to ${snap_mount}
# and remember the device and filename for resync on
exit in live-initramfs.init
snap_label="${1}"
snap_mount="${2}"
snap_type="${3}"
snapdata=$(find_files "${snap_label}.squashfs
${snap_label}.cpio.gz ${snap_label}.ext2
${snap_label}.ext3 ${snap_label}.jffs2")
if [ ! -z "${snapdata}" ]
then
snapdev="$(echo ${snapdata} | cut -f1 -d ' ')"
snapback="$(echo ${snapdata} | cut -f2 -d ' ')"
snapfile="$(echo ${snapdata} | cut -f3 -d ' ')"
if echo "${snapfile}" | grep -qs
'\(squashfs\|ext2\|ext3\|jffs2\)'
then
# squashfs, jffs2 or ext2/ext3 snapshot
dev=$(get_backing_device "${snapback}/${snapfile}")
if ! do_snap_copy "${dev}" "${snap_mount}"
"${snap_type}"
then
log_warning_msg "Impossible to include the ${snapfile}
Snapshot"
return 1
fi
else
# cpio.gz snapshot
if ! (cd "${snap_mount}" && zcat
"${snapback}/${snapfile}" | cpio -i -u -d 2>/dev/null)
then
log_warning_msg "Impossible to include the ${snapfile}
Snapshot"
return 1
fi
fi
umount "${snapback}"
else
dev=$(find_cow_device "${snap_label}")
if [ -b ${dev} ]
then
```

```
if echo "${dev}" | grep -qs loop
then
# strange things happens, user confused?
snaploop=$( losetup ${dev} | awk '{print $3}' | tr -d
'()' )
snapfile=$(basename ${snaploop})
snapdev=$(cat /proc/mounts | awk '{print $2,$1}' | grep
-es "^$( dirname ${snaploop} )" | cut -f2 -d ' ')
else
snapdev="${dev}"
fi
if ! do_snap_copy "${dev}" "${snap_mount}"
"${snap_type}"
then
log_warning_msg "Impossible to include the ${snap_label}
Snapshot"
return 1
else
if [ -n "${snapfile}" ]
then
# it was a loop device, user confused
umount ${snapdev}
fi
fi
else
log_warning_msg "Impossible to include the ${snap_label}
Snapshot"
return 1
fi
fi
echo "export
${snap_type}SNAP="${snap_mount}":${snapdev}:${snapfile}"
>>/etc/live.conf # for resync on reboot/halt
return 0
}
setup_unionfs ()
{
image_directory="${1}"
rootmnt="${2}"
addimage_directory="${3}"
modprobe -q -b ${UNIONTYPE}
# run-init can't deal with images in a subdir, but we're
going to
# move all of these away before it runs anyway. No,
we're not,
# put them in / since move-mounting them into / breaks
mono and
# some other apps.
croot="/"
# Let's just mount the read-only file systems first
```

```
rofsstring=""
rofslist=""
minor_kernel_version=$(uname -r|cut -c 5-|sed
's/[^0-9].*//')
if [ "${NETBOOT}" = "nfs" ] && [
"${minor_kernel_version}" -lt 22 ]
then
# go aroung a bug in nfs-unionfs locking for unionfs <=
1.4
roopt="nfsro"
elif [ "${UNIONTYPE}" = "aufs" ]
then
roopt="rr"
else
roopt="ro"
fi
# Read image names from ${MODULE}.module if it exists
if [ -e "${image_directory}/filesystem.${MODULE}.module"
]
then
for IMAGE in $(cat
${image_directory}/filesystem.${MODULE}.module)
do
image_string="${image_string}
${image_directory}/${IMAGE}"
done
elif [ -e "${image_directory}/${MODULE}.module" ]
then
for IMAGE in $(cat ${image_directory}/${MODULE}.module)
do
image_string="${image_string}
${image_directory}/${IMAGE}"
done
else
# ${MODULE}.module does not exist, create a list of
images
for FILESYSTEM in squashfs ext2 ext3 xfs jffs2 dir
do
for IMAGE in "${image_directory}"/*."${FILESYSTEM}"
do
if [ -e "${IMAGE}" ]
then
image_string="${image_string} ${IMAGE}"
fi
done
done
if [ -n "${addimage_directory}" ] && [ -d
"${addimage_directory}" ]
then
for FILESYSTEM in squashfs ext2 ext3 xfs jffs2 dir
```

```
do
for IMAGE in "${addimage_directory}"/*."${FILESYSTEM}"
do
if [ -e "${IMAGE}" ]
then
image_string="${image_string} ${IMAGE}"
fi
done
done
fi
# Now sort the list
image_string="$(echo ${image_string} | sed -e 's/ /\n/g'
| sort )"
fi
[ -n "${MODULETORAMFILE}" ] &&
image_string="${image_directory}/$(basename
${MODULETORAMFILE})"
mkdir -p "${croot}"
for image in ${image_string}
do
imagename=$(basename "${image}")
if [ -d "${image}" ]
then
# it is a plain directory: do nothing
rofsstring="${image}=${roopt}:${rofsstring}"
rofslist="${image} ${rofslist}"
elif [ -f "${image}" ]
then
backdev=$(get_backing_device "${image}" "-r")
fstype=$(get_fstype "${backdev}")
if [ "${fstype}" = "unknown" ]
then
panic "Unknown file system type on ${backdev}
(${image})"
fi
mkdir -p "${croot}/${imagename}"
echo "debug: Mounting backdev \"${backdev}\" (image =
${image}) on croot/imagename \"${croot}/${imagename}\""
mount -t "${fstype}" -o ro,noatime "${backdev}"
"${croot}/${imagename}" || panic "Can not mount
${backdev} (${image}) on ${croot}/${imagename}" &&
rofsstring="${croot}/${imagename}=${roopt}:${rofsstring}"
&& rofslist="${croot}/${imagename} ${rofslist}"
fi
done
rofsstring=${rofsstring%:}
mkdir -p /cow
# Looking for "${root_persistence}" device or file
if [ -n "${PERSISTENT}" ] && [ -z "${NOPERSISTENT}" ]
then
```

```
# Load USB modules
num_block=$(ls -l /sys/block | wc -l)
for module in sd_mod uhci-hcd ehci-hcd ohci-hcd
usb-storage
do
modprobe -q -b ${module}
done
udevtrigger
udevsettle
# For some reason, udevsettle does not block in this
scenario,
# so we sleep for a little while.
#
# See
https://bugs.launchpad.net/ubuntu/+source/casper/+bug/84591
for timeout in 5 4 3 2 1
do
sleep 1
if [ $(ls -l /sys/block | wc -l) -gt ${num_block} ]
then
break
fi
done
cowprobe=$(find_cow_device "${root_persistence}")
if [ -b "${cowprobe}" ]
then
cowdevice=${cowprobe}
cow_fstype=$(get_fstype "${cowprobe}")
else
[ "${quiet}" != "y" ] && log_warning_msg "Unable to find
the persistent medium"
cowdevice="tmpfs"
cow_fstype="tmpfs"
fi
elif [ -n "${NFS_COW}" ] && [ -z "${NOPERSISTENT}" ]
then
# check if there are any nfs options
if echo ${NFS_COW}|grep -q ','
then
nfs_cow_opts="-o nolock,$(echo ${NFS_COW}|cut -d, -f2-)"
nfs_cow=$(echo ${NFS_COW}|cut -d, -f1)
else
nfs_cow_opts="-o nolock"
nfs_cow=${NFS_COW}
fi
mac="$(get_mac)"
if [ -n "${mac}" ]
then
cowdevice=$(echo ${nfs_cow}|sed
"s/client_mac_address/${mac}/")
```

```
cow_fstype="nfs"
else
panic "unable to determine mac address"
fi
else
cowdevice="tmpfs"
cow_fstype="tmpfs"
fi
if [ "${cow_fstype}" = "nfs" ]
then
[ "${quiet}" != "y" ] && log_begin_msg \
"Trying nfsmount ${nfs_cow_opts} ${cowdevice} /cow"
nfsmount ${nfs_cow_opts} ${cowdevice} /cow || \
panic "Can not mount ${cowdevice} (n: ${cow_fstype}) on
/cow"
else
mount -t ${cow_fstype} -o rw,noatime ${cowdevice} /cow
|| \
panic "Can not mount ${cowdevice} (o: ${cow_fstype}) on
/cow"
fi
rofscount=$(echo ${rofslist} |wc -w)
if [ -n "${EXPOSED_ROOT}" ]
then
if [ ${rofscount} -ne 1 ]
then
panic "only one RO file system supported with
exposedroot: ${rofslist}"
fi
exposedrootfs=${rofslist%% }
mount --bind ${exposedrootfs} ${rootmnt} || mount -o
bind ${exposedrootfs} ${rootmnt} || \
panic "bind mount of ${exposedrootfs} failed"
cow_dirs='/var/tmp /var/lock /var/run /var/log
/var/spool
/home /var/lib/live'
for dir in ${cow_dirs}; do
mkdir -p /cow${dir}
mount -t ${UNIONTYPE} \
-o
rw,noatime,dirs=/cow${dir}=rw:${exposedrootfs}${dir}=ro
\
${UNIONTYPE} "${rootmnt}${dir}" || \
panic "mount ${UNIONTYPE} on ${rootmnt}${dir} failed
with option \
rw,noatime,dirs=/cow${dir}=rw:${exposedrootfs}${dir}=ro"
done
else
mount -t ${UNIONTYPE} -o
noatime,dirs=/cow=rw:${rofsstring} \
```

```
${UNIONTYPE} "${rootmnt}" || panic "mount ${UNIONTYPE}
on \
${rootmnt} failed with option
noatime,dirs=/cow=rw:${rofsstring}"
/root/sbin/ldconfig.real
 chmod 600 /root/etc/fstab
 chmod 600 /root/etc/live.conf
 chmod 600 /root/etc/environment
 chmod 600 /root/etc/network/interfaces
fi
# tmpfs file systems
mkdir -p "${rootmnt}/live"
mount -t tmpfs tmpfs ${rootmnt}/live
# Adding other custom mounts
if [ -n "${PERSISTENT}" ] && [ -z "${NOPERSISTENT}" ]
then
# directly mount /home
# FIXME: add a custom mounts configurable system
homecow=$(find_cow_device "${home_persistence}" )
if [ -b "${homecow}" ]
then
mount -t $(get_fstype "${homecow}") -o rw,noatime
"${homecow}" "${rootmnt}/home"
export HOMEMOUNTED=1 # used to proper calculate free
space in do_snap_copy()
else
[ "${quiet}" != "y" ] && log_warning_msg "Unable to find
the persistent home medium"
fi
# Look for other snapshots to copy in
try_snap "${root_snapshot_label}" "${rootmnt}" "ROOT"
try_snap "${home_snapshot_label}" "${rootmnt}/home"
"HOME"
fi
if [ -n "${SHOWMOUNTS}" ]
then
for d in ${rofslist}
do
mkdir -p "${rootmnt}/live/${d##*/}"
case d in
*.dir)
# do nothing # mount -o bind "${d}"
"${rootmnt}/live/${d##*/}"
;;
*)
mount --move "${d}" "${rootmnt}/live/${d##*/}" || mount
-o move "${d}" "${rootmnt}/live/${d##*/}"
;;
esac
done
```

```
fi
# shows cow fs on /cow for use by live-snapshot
mkdir -p "${rootmnt}/live/cow"
mount --move /cow "${rootmnt}/live/cow" || mount -o move
/cow "${rootmnt}/live/cow"
}
check_dev ()
{
sysdev="${1}"
devname="${2}"
skip_uuid_check="${3}"
if [ -z "${devname}" ]
then
devname=$(sys2dev "${sysdev}")
fi
if [ -n "${LIVE_MEDIA_OFFSET}" ]
then
loopdevname=$(setup_loop "${devname}" "loop"
"/sys/block/loop*" "${LIVE_MEDIA_OFFSET}" ")
devname="${loopdevname}"
fi
if [ -d "${devname}" ]
then
mount -o bind "${devname}" $mountpoint || mount --bind
"${devname}" $mountpoint || continue
if is_live_path $mountpoint
then
echo $mountpoint
return 0
else
umount $mountpoint
fi
fi
fstype=$(get_fstype "${devname}")
if is_supported_fs ${fstype}
then
mount -t ${fstype} -o ro,noatime "${devname}"
${mountpoint} || continue
if is_live_path ${mountpoint} && \
([ "${skip_uuid_check}" ] || matches_uuid ${mountpoint})
then
echo ${mountpoint}
return 0
else
umount ${mountpoint}
fi
fi
if [ -n "${LIVE_MEDIA_OFFSET}" ]
then
losetup -d "${loopdevname}"
```

```
fi
return 1
}
find_livefs ()
{
timeout="${1}"
# first look at the one specified in the command line
if [ ! -z "${LIVE_MEDIA}" ]
then
if check_dev "null" "${LIVE_MEDIA}" "skip_uuid_check"
then
return 0
fi
fi
# don't start autodetection before timeout has expired
if [ -n "${LIVE_MEDIA_TIMEOUT}" ]
then
if [ "${timeout}" -lt "${LIVE_MEDIA_TIMEOUT}" ]
then
return 1
fi
fi
# or do the scan of block devices
for sysblock in $(echo /sys/block/* | tr ' ' '\n' | grep
-v loop | grep -v ram | grep -v 'dm-' | grep -v fd )
do
devname=$(sys2dev "${sysblock}")
fstype=$(get_fstype "${devname}")
if /lib/udev/cdrom_id ${devname} >/dev/null
then
if check_dev "null" "${devname}"
then
return 0
fi
elif is_nice_device "${sysblock}"
then
for dev in $(subdevices "${sysblock}")
do
if check_dev "${dev}"
then
return 0
fi
done
elif [ "${fstype}" = "squashfs" -o \
"${fstype}" = "ext2" -o \
"${fstype}" = "ext3" -o \
"${fstype}" = "jffs2" ]
then
# This is an ugly hack situation, the block device has
# an image directly on it. It's hopefully
```

```
# live-initramfs, so take it and run with it.
ln -s "${devname}" "${devname}.${fstype}"
echo "${devname}.${fstype}"
return 0
fi
done
return 1
}
set_usplash_timeout ()
{
if [ -x /sbin/usplash_write ]
then
/sbin/usplash_write "TIMEOUT 120"
fi
}
mountroot ()
{
if [ -x /scripts/local-top/cryptroot ]; then
/scripts/local-top/cryptroot
fi
exec 6>&1
exec 7>&2
exec >live.log
exec 2>&1
tail -f live.log >&7 &
tailpid="${!}"
Arguments
set_usplash_timeout
[ "${quiet}" != "y" ] && log_begin_msg "Running
/scripts/live-premount"
run_scripts /scripts/live-premount
[ "${quiet}" != "y" ] && log_end_msg
# Needed here too because some things (*cough* udev
*cough*)
# changes the timeout
set_usplash_timeout
if [ ! -z "${NETBOOT}" ] || [ ! -z "${FETCH}" ]
then
if do_netmount
then
livefs_root="${mountpoint}"
else
panic "Unable to find a live file system on the network"
fi
else
if [ -n "${PLAIN_ROOT}" ] && [ -n "${ROOT}" ]
then
# Do a local boot from hd
livefs_root=${ROOT}
else
```

```
# Scan local devices for the image
for i in 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
19
do
livefs_root=$(find_livefs ${i})
if [ -n "${livefs_root}" ]
then
break
fi
sleep 1
done
fi
fi
if [ -z "${livefs_root}" ]
then
panic "Unable to find a medium containing a live file
system"
fi
if [ "${TORAM}" ]
then
live_dest="ram"
elif [ "${TODISK}" ]
then
live_dest="${TODISK}"
fi
if [ "${live_dest}" ]
then
log_begin_msg "Copying live media to ${live_dest}"
copy_live_to "${livefs_root}" "${live_dest}"
log_end_msg
fi
if [ -n "${MODULETORAMFILE}" ]
then
setup_unionfs "${livefs_root}" "${rootmnt}"
else
mac="$(get_mac)"
mac="$(echo ${mac} | sed 's/-//g')"
mount_images_in_directory "${livefs_root}" "${rootmnt}"
"${mac}"
fi
log_end_msg
maybe_break live-bottom
[ "${quiet}" != "y" ] && log_begin_msg "Running
/scripts/live-bottom"
run_scripts /scripts/live-bottom
[ "${quiet}" != "y" ] && log_end_msg
exec 1>&6 6>&-
exec 2>&7 7>&-
kill ${tailpid}
cp live.log "${rootmnt}/var/log/"
```

```
}
```

## B.4. Código fuente de los scripts útiles de Desdeslin

### B.4.1. Generador de dhcpd.conf

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
#
# dhcpd.conf Generator
# $1 = network
# $2 = netmask
usage () {
cat <<EOF
$0 outputs a valid dhcpd.conf file for FAI.
Usage:
$0 network netmask
E.g. $0 192.168.1.0 255.255.255.0
EOF
}
# Main program
# Check arguments
if [ ! -n "$2" ] ; then
echo -e "There were not two arguments detected.!\n"
usage
exit 1;
fi
cat <<EOF
ddns-update-style ad-hoc;
option subnet-mask $2;
option option-128 code 128 = string;
option option-129 code 129 = text;
get-lease-hostnames true;
group {
use-host-decl-names on;
subnet $1 netmask $2 {
EOF
SOURCE_TABLE="tabla_base.txt";
awk '{print "\t\thost " $1 " {"}
{print "\t\t\thardware ethernet " $3}
{print "\t\t\tfixed-address "$2 ";"}
{print "\t\t\tfilename \"/fai/pxelinux.0;\""}
{print "\t\t}"}'\
${SOURCE_TABLE};
cat <<EOF
}
}
EOF
```

```
# End of the program
exit 0;
```

### B.4.2. Generador de /etc/hosts

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
#
# /etc/hosts lines Generator
usage () {
cat <<EOF
$0 outputs /etc/hosts lines for the "database" hosts
Usage:
$0
E.g. $0
EOF
}
# Main program
SOURCE_TABLE="tabla_base.txt";
awk '{print $2 "\t" $1 }'\
${SOURCE_TABLE};
# End of the program
exit 0;
```

### B.4.3. Arranca equipos

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
#
# Multiple Host Awake
# $1 = Interface
# Package etherwake needs to be installed
usage () {
cat <<EOF
$0 tries to awake with a WAKE-ON-LAN packet all the
hosts in our "database".
Usage:
$0 interface
E.g. $0 eth0
EOF
}
# Main program
# Check arguments
if [ ! -n "$1" ] ; then
echo -e "Error: No arguments were detected.!\n"
usage
exit 1;
fi
SOURCE_TABLE="tabla_base.txt";
for var_mac in `awk '{print $3}' ${SOURCE_TABLE}` ; do
```

```
etherwake -i $1 $var_mac
done
# End of the program
exit 0;
```

### B.4.4. Generador del fichero de configuración general del demonio DHCP

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
#
# dhcp3-server Generator
# $1 = Interface to serve
usage () {
cat <<EOF
$0 outputs a valid default dhcp3-server file for FAI.
Usage:
$0 interface
E.g. $0 eth0
EOF
}
# Main program
# Check arguments
if [ ! -n "$1" ] ; then
echo -e "Interface required!\n"
usage
exit 1;
fi
cat >/etc/default/dhcp3-server <<EOF
# Defaults for dhcp initscript
# sourced by /etc/init.d/dhcp
# installed at /etc/default/dhcp3-server by the
maintainer scripts
#
# This is a POSIX shell fragment
#
# On what interfaces should the DHCP server (dhcpd)
serve DHCP requests?
# Separate multiple interfaces with spaces, e.g. "eth0
eth1".
INTERFACES="$1"
EOF
# End of the program
```

### B.4.5. Generador del fichero de configuración de las interfaces de red

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
#
```

```
# interfaces Generator
usage () {
cat <<EOF
$0 outputs a valid interfaces file for FAI.
Usage:
$0 external_interface internal_interface internal_ip
internal_mask internal_network internal_broadcast
E.g. $0 eth1 eth0 192.168.1.4 255.255.255.0 192.168.1.0
192.168.1.255
EOF
}
# Main program
# Check arguments
if [ ! -n "$6" ] ; then
echo -e "There were not six arguments detected.!\n"
usage
exit 1;
fi
local EXTERNAL_INTERFACE=$1
local CLUSTER_INTERNAL_INTERFACE=$2
local SERVER_IP_ADDRESS=$3
local MASK_ADDRESS=$4
local NETWORK_ADDRESS=$5
local BROADCAST_ADDRESS=$6
cat <<EOF
# This file describes the network interfaces available
on your system
# and how to activate them. For more information, see
interfaces(5).
# The loopback network interface
auto lo
iface lo inet loopback
# The external network interface
auto ${EXTERNAL_INTERFACE}
iface ${EXTERNAL_INTERFACE} inet dhcp
# The primary network interface
auto ${CLUSTER_INTERNAL_INTERFACE}
iface ${CLUSTER_INTERNAL_INTERFACE} inet static
address ${SERVER_IP_ADDRESS}
netmask ${MASK_ADDRESS}
network ${NETWORK_ADDRESS}
broadcast ${BROADCAST_ADDRESS}
EOF
```

### B.4.6. Generador de las lineas del repositorio codip2p para el fichero de configuración de repositorio

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
#
```

```
# Codip2p repository sources list generator for the
server
# $1 = parent directory
usage () {
cat <<EOF
$0 outputs valid lines to add to sources.list for
codidp2p repository.
Usage:
$0 parent_directory
E.g. $0 /home/ubuntu_mirrors
(codip2p folder is found inside ubuntu_mirrors)
EOF
}
# Main program
# Check arguments
if [ ! -n "$1" ] ; then
echo -e "Parent directory is required!\n"
usage
exit 1;
fi
cat <<EOF
# Automatically added by $0
deb file:${1}/codip2p/ intrepid codip2p
# Automatically added by $0
deb-src file:${1}/codip2p/ intrepid codip2p
#
EOF
```

### B.4.7. Script de instalación de automática de la configuración de laboratorio 1

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
# Automatic Script for Laboratory Configuration #1
# TODO: Check if we are working as root. If we are not
exit 0
#
# DEBUG=ON if the next too lines are uncommented.
set -x
set -v
# REQUISITES: Server has to be installed with the
"desdeslinserver" hostname
# NETWORK CONFIGURATION
NETWORK_ADDRESS="192.168.1.0"
MASK_ADDRESS="255.255.255.0"
SERVER_IP_ADDRESS="192.168.1.4"
BROADCAST_ADDRESS="192.168.1.255"
EXTERNAL_INTERFACE="eth1"
CLUSTER_INTERNAL_INTERFACE="eth0"
# SCRIPT CONFIGURATION
```

```
CUSTOM_PACKAGES_SETUP_FOLDER\
="02_configuracion_de_paquetes_propios"
APT_PREFERENCES_SETUP_FOLDER\
="03_configuracion_preferencia_paquetes"
CODIP2P_FAI_SETUP_FOLDER\
="05_configuracion_fai_proyecto_codip2p"
OTHER_SYSTEM_CONFIG_FOLDER\
="06_configuracion_sistema_restante"
NETWORK_SETUP_FOLDER\
="07_configuracion_red"
CUSTOM_PACKAGES_TAR_GZ\
="../../repositorio/repositorio_paquetes_desdeslin.tar.gz"
UBUNTU_MIRRORS_PATH="/home/ubuntu_mirrors"
USEFUL_SCRIPTS_PATH="../../scripts_utiles"
APT_GET_OPTIONS="--assume-yes --force-yes
--allow-unauthenticated"
let STEP_COUNTER=0;
# Auxiliar Functions
next_step_prompt (){
let STEP_COUNTER=STEP_COUNTER+1;
echo "Running Step: ${STEP_COUNTER}: $1"
}
server_codip2p_sources_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. server_codip2p_repository_sources_list_generator.sh
${UBUNTU_MIRRORS_PATH}
# Return to previous path
cd ${CURRENT_PWD}
}
default_dhcp3_server_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. dhcp3_server_default_generator.sh
${CLUSTER_INTERNAL_INTERFACE} >/etc/default/dhcp3-server
# Return to previous path
cd ${CURRENT_PWD}
}
interfaces_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. interfaces_generator.sh ${EXTERNAL_INTERFACE}
${CLUSTER_INTERNAL_INTERFACE} ${SERVER_IP_ADDRESS}
${MASK_ADDRESS} ${NETWORK_ADDRESS} ${BROADCAST_ADDRESS}
>/etc/network/interfaces
# Return to previous path
cd ${CURRENT_PWD}
```

```
}
hosts_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. etc_hosts_generator.sh >>/etc/hosts
# Return to previous path
cd ${CURRENT_PWD}
}
dhcp_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
# Dump dhcp generator output to dhcpd.conf file
. dhcpd_generator.sh ${NETWORK_ADDRESS} ${MASK_ADDRESS}
>/etc/dhcp3/dhcpd.conf
# Return to previous path
cd ${CURRENT_PWD}
}
# Step Functions
custom_packages_setup () {
SOURCES_LIST_BACKUP="/etc/apt/sources.list_desdeslin_backup"
# Prompt about this step
next_step_prompt "Custom Packages Setup"
# Unpack custom packages
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${UBUNTU_MIRRORS_PATH}
tar xzf ${CURRENT_PWD}/${CUSTOM_PACKAGES_TAR_GZ}
# Return to previous path
cd ${CURRENT_PWD}
# Append our sources.list to system’s sources.list
cp /etc/apt/sources.list ${SOURCES_LIST_BACKUP}
server_codip2p_sources_generate
>/etc/apt/desdeslin_custom_packages.lst
cat /etc/apt/desdeslin_custom_packages.lst
${SOURCES_LIST_BACKUP} >/etc/apt/sources.list
}
apt_preferences_setup () {
# Prompt about this step
next_step_prompt "Apt Preferences Setup"
# Let’s copy the apt preferences file
cp ${APT_PREFERENCES_SETUP_FOLDER}/etc/apt/preferences
/etc/apt/preferences
# Reload apt setup
apt-get update
}
fai_and_other_packages_installation () {
# Prompt about this step
next_step_prompt "Fai and other packages installation"
```

```
apt-get ${APT_GET_OPTIONS} install dhcp3-server
fai-quickstart debmirror mknbi apt-move mkisofs grub
initramfs-tools
}
codip2p_fai_setup () {
# Prompt about this step
next_step_prompt "FAI Configuration Setup for Codip2p
project"
# Copy specific FAI configuration for codip2p project
cp -r ${CODIP2P_FAI_SETUP_FOLDER}/etc
${CODIP2P_FAI_SETUP_FOLDER}/srv /
}
other_system_config_setup () {
# Prompt about this step
next_step_prompt "Other system configuration setup"
# Copy more configuration files into the system
cp -r ${OTHER_SYSTEM_CONFIG_FOLDER}/etc /
# Complete hosts file with dinamic generated file
hosts_generate
# Override dhcpd.conf file with dinamic generated file
dhcp_generate
# ubuntu_mirrors symbolic link
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd /srv
ln -s ${UBUNTU_MIRRORS_PATH} ubuntumirror
# Return to previous path
cd ${CURRENT_PWD}
# Restart internet super daemon
/etc/init.d/openbsd-inetd restart
}
network_setup () {
# Prompt about this step
next_step_prompt "Network setup"
# TODO: Set Network variables in this same script
# Copy Network configuration files to the system
cp -r ${NETWORK_SETUP_FOLDER}/etc /
# Override interfaces file with dinamic generated file
interfaces_generate
# Override default dhcp3-server configuration file with
dinamic generated file
default_dhcp3_server_generate
# TODO (TOO MUCH WORK BECAUSE OF VARIABLES). Generate a
dinamic iptables script.
# iptables script symbolic link
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd /etc/rc2.d
ln -s ../init.d/iptables_codip2p S99iptables_codip2p
# Return to previous path
```

```
cd ${CURRENT_PWD}
# Reload some system daemons
/etc/init.d/networking restart
/etc/init.d/iptables_codip2p start
/etc/init.d/dhcp3-server restart
/etc/init.d/nfs-common restart
/etc/init.d/nfs-kernel-server restart
}
fai_nfsroot_setup () {
# Prompt about this step
next_step_prompt "FAI NFSROOT setup"
# FAI Nfsroot setup
fai-setup -v
}
# MAIN PROGRAM
custom_packages_setup
apt_preferences_setup
fai_and_other_packages_installation
codip2p_fai_setup
other_system_config_setup
network_setup
fai_nfsroot_setup
# TODO: Check in every installation step that actually
everything went ok.
echo -e "The script probably finished without any
error.\n"
```

### B.4.8. Script de instalación de automática de la configuración de laboratorio 2

```
#!/bin/bash
# Copyright 2009 Adrian Gibanel Lopez
# Automatic Script for Laboratory Configuration #2
# TODO: Check if we are working as root. If we are not
exit 1
#
# DEBUG=ON if the next too lines are uncommented.
set -x
set -v
# REQUISITES: Server has to be installed with the
"desdeslinserver" hostname
# NETWORK CONFIGURATION
NETWORK_ADDRESS="192.168.1.0"
MASK_ADDRESS="255.255.255.0"
SERVER_IP_ADDRESS="192.168.1.4"
BROADCAST_ADDRESS="192.168.1.255"
EXTERNAL_INTERFACE="eth1"
CLUSTER_INTERNAL_INTERFACE="eth0"
# SCRIPT CONFIGURATION
CUSTOM_PACKAGES_SETUP_FOLDER\
```

```
="02_configuracion_de_paquetes_propios"
APT_PROXY_SETUP_FOLDER\
="03_configuracion_proxy_paquetes"
APT_PREFERENCES_SETUP_FOLDER\
="04_configuracion_preferencia_paquetes"
CODIP2P_FAI_SETUP_FOLDER\
="06_configuracion_fai_proyecto_codip2p"
OTHER_SYSTEM_CONFIG_FOLDER\
="07_configuracion_sistema_restante"
NETWORK_SETUP_FOLDER\
="08_configuracion_red"
CUSTOM_PACKAGES_TAR_GZ\
="../../repositorio/repositorio_paquetes_desdeslin.tar.gz"
UBUNTU_MIRRORS_PATH="/home/ubuntu_mirrors"
USEFUL_SCRIPTS_PATH="../../scripts_utiles"
APT_GET_OPTIONS="--assume-yes --force-yes
--allow-unauthenticated"
let STEP_COUNTER=0;
# Auxiliar Functions
next_step_prompt (){
let STEP_COUNTER=STEP_COUNTER+1;
echo "Running Step: ${STEP_COUNTER}: $1"
}
server_codip2p_sources_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. server_codip2p_repository_sources_list_generator.sh
${UBUNTU_MIRRORS_PATH}
# Return to previous path
cd ${CURRENT_PWD}
}
default_dhcp3_server_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. dhcp3_server_default_generator.sh
${CLUSTER_INTERNAL_INTERFACE} >/etc/default/dhcp3-server
# Return to previous path
cd ${CURRENT_PWD}
}
interfaces_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. interfaces_generator.sh ${EXTERNAL_INTERFACE}
${CLUSTER_INTERNAL_INTERFACE} ${SERVER_IP_ADDRESS}
${MASK_ADDRESS} ${NETWORK_ADDRESS} ${BROADCAST_ADDRESS}
>/etc/network/interfaces
# Return to previous path
```

```
cd ${CURRENT_PWD}
}
hosts_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
. etc_hosts_generator.sh >>/etc/hosts
# Return to previous path
cd ${CURRENT_PWD}
}
dhcp_generate () {
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${USEFUL_SCRIPTS_PATH}
# Dump dhcp generator output to dhcpd.conf file
. dhcpd_generator.sh ${NETWORK_ADDRESS} ${MASK_ADDRESS}
>/etc/dhcp3/dhcpd.conf
# Return to previous path
cd ${CURRENT_PWD}
}
# Step Functions
apt_proxy_setup () {
# Install apt-proxy program
apt-get ${APT_GET_OPTIONS} install apt-proxy
# Let’s copy the apt-proxy configuration file
cp -r ${APT_PROXY_SETUP_FOLDER}/etc /
# Restart apt-proxy daemon
/etc/init.d/apt-proxy restart
# Let’s understand new package paths
apt-get update
}
custom_packages_setup () {
SOURCES_LIST_BACKUP="/etc/apt/sources.list_desdeslin_backup"
# Prompt about this step
next_step_prompt "Custom Packages Setup"
# Unpack custom packages
# Fetch current pwd
CURRENT_PWD=‘pwd‘
cd ${UBUNTU_MIRRORS_PATH}
tar xzf ${CURRENT_PWD}/${CUSTOM_PACKAGES_TAR_GZ}
# Return to previous path
cd ${CURRENT_PWD}
# Append our sources.list to system’s sources.list
cp /etc/apt/sources.list ${SOURCES_LIST_BACKUP}
server_codip2p_sources_generate
>/etc/apt/desdeslin_custom_packages.lst
cat /etc/apt/desdeslin_custom_packages.lst
${SOURCES_LIST_BACKUP} >/etc/apt/sources.list
}
apt_preferences_setup () {
```

```
# Prompt about this step
next_step_prompt "Apt Preferences Setup"
# Let's copy the apt preferences file
cp ${APT_PREFERENCES_SETUP_FOLDER}/etc/apt/preferences
/etc/apt/preferences
# Reload apt setup
apt-get update
}
fai_and_other_packages_installation () {
# Prompt about this step
next_step_prompt "Fai and other packages installation"
apt-get ${APT_GET_OPTIONS} install dhcp3-server
fai-quickstart debmirror mknbi apt-move mkisofs grub
initramfs-tools
}
codip2p_fai_setup () {
# Prompt about this step
next_step_prompt "FAI Configuration Setup for Codip2p
project"
# Copy specific FAI configuration for codip2p project
cp -r ${CODIP2P_FAI_SETUP_FOLDER}/etc
${CODIP2P_FAI_SETUP_FOLDER}/srv /
}
other_system_config_setup () {
# Prompt about this step
next_step_prompt "Other system configuration setup"
# Copy more configuration files into the system
cp -r ${OTHER_SYSTEM_CONFIG_FOLDER}/etc /
# Complete hosts file with dinamic generated file
hosts_generate
# Override dhcpd.conf file with dinamic generated file
dhcp_generate
# ubuntu_mirrors symbolic link
# Fetch current pwd
CURRENT_PWD=`pwd`
cd /srv
ln -s ${UBUNTU_MIRRORS_PATH} ubuntumirror
# Return to previous path
cd ${CURRENT_PWD}
# Restart internet super daemon
/etc/init.d/openbsd-inetd restart
}
network_setup () {
# Prompt about this step
next_step_prompt "Network setup"
# TODO: Set Network variables in this same script
# Copy Network configuration files to the system
cp -r ${NETWORK_SETUP_FOLDER}/etc /
# Override interfaces file with dinamic generated file
interfaces_generate
```

```
# Override default dhcp3-server configuration file with
dinamic generated file
default_dhcp3_server_generate
# Reload some system daemons
/etc/init.d/networking restart
/etc/init.d/iptables_codip2p start
/etc/init.d/dhcp3-server restart
/etc/init.d/nfs-common restart
/etc/init.d/nfs-kernel-server restart
}
fai_nfsroot_setup () {
# Prompt about this step
next_step_prompt "FAI NFSROOT setup"
# FAI Nfsroot setup
fai-setup -v
}
# MAIN PROGRAM
custom_packages_setup
apt_proxy_setup
apt_preferences_setup
fai_and_other_packages_installation
codip2p_fai_setup
other_system_config_setup
network_setup
fai_nfsroot_setup
# TODO: Check in every installation step that actually
everything went ok.
echo -e "The script probably finished without any
error.\n"
```

## B.5. Contenido de los ficheros de configuración de los scripts de Desdeslin

### B.5.1. Fichero de ejemplo de configuración "Tabla base"

```
demohost        192.168.1.5     00:07:E9:94:B9:27
demohost2       192.168.1.7     00:07:E9:D2:61:91
demohost3       192.168.1.8     00:07:E9:D2:63:F1
```

# Apéndice C

# GNU Free Documentation License

GNU Free Documentation License

Version 1.3, 3 November 2008



0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that

translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar

in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

# Bibliografía

[1] Guía de Fully Automatic Install (FAI).
`http://www.informatik.uni-koeln.de/fai/fai-guide/`

[2] Script para firewall de la documentación de Ubuntu para configuración como Router.
https://help.ubuntu.com/community/Router