LTSP − Linux Terminal Server Project − v3.0

James McQuillan

jam@LTSP.org

Historial de revisiones

Revisión 1.0.4	2002−03−02	Revised by: jam

GNU/Linux es una gran plataforma para el uso de estaciones de trabajo sin disco. El objetivo principal de este documento es mostrarte cómo puedes utilizar LTSP para desplegar clientes sin disco. Sin embargo, este documento también muestra cuestiones de las estaciones sin disco en general.

Tabla de Contenidos

Tabla de Contenidos

Introducción

El LTSP provee una manera simple de utilizar estaciones de trabajo de bajo costo tanto como terminales gráficas o bien como terminales de caracteres sobre un servidor GNU/Linux.

En una configuración de oficina tradicional, hay PCs de bastante poder desparramadas en cada escritorio. Cada una con varios gigabytes de espacio en disco. Los usuarios almacenan su información en sus discos locales y las copias de resguardo se realizan raramente (si es que se realizan).

¿Tiene realmente sentido tener un computador completo en cada escritorio?

Nosotros decimos que no.

Afortunadamente, hay otro camino. Utilizando LTSP, puedes tomar cada PC de bajo poder, removerle el disco duro, el floppy y la lectora de CDs y agregarle una placa de red con chip de inicio. Muchas tarjetas de red poseen sócalos para bootroms (memorias ROM de inicio), que están esperando que se les inserte uno de estos chips.

Durante la fase de inicio, la estación de trabajo sin disco obtiene su dirección IP y un kernel (núcleo) desde el servidor, montando luego su sistema de archivos raíz desde el mismo servidor vía NFS.

La estación de trabajo puede ser configurada de tres maneras:

- **X Window Gráfico**

  Utilizando X Window, la estación de trabajo puede ser usada para acceder cualquier aplicación en el servidor, o en cualquier otro servidor dentro de la red.

- **Sesiones de Telnet basadas en caracteres**

  Cada estación de trabajo puede invocar múltiples sesiones de telnet al servidor. Cada una de estas sesiones aparecerá en una pantalla virtual separada. Presionando desde Alt−F1 hasta Alt−F9 se cambiará entre cada una de dichas sesiones.

- **Prompt de Shell**

  La estación de trabajo puede ser configurada para dejarte justo dentro de una sesión de Bash en la consola. Esto es muy útil para depurar problemas con X Window o con NFS.

Lo que es realmente impresionante de todo esto es que puedes tener montones de estaciones de trabajo, todas servidas desde un único servidor GNU/Linux. ¿Hasta cuántas estaciones te preguntas? Pues bien, esto depende del tamaño del servidor y de las aplicaciones que has de utilizar.

No es inusual tener 40 estaciones de trabajo, todas corriendo Netscape y StarOffice, desde un Dual PIII−650 con 1GB de RAM. Sabemos que esto funciona. ¡De hecho, el promedio de carga muy rara vez sobrepasa 1.0!

---

1. Disclaimer

Ni el autor ni ninguno de los distribuidores, o cualquier otro contribuyente a este documento, es en modo alguno responsable por los daños físicos, morales, o de cualquier otro tipo incurridos por sólo hecho de seguir las sugerencias citadas en este texto.

2. Licencia y Copyright

Capítulo 1. Teoría de Operación

Iniciar una estación de trabajo sin disco involucra muchos pasos. Comprender qué está pasando a través del camino hará mucho mas fácil resolver los problemas que pudiesen surgir.

Este ejemplo está basado en la siguiente configuración:

- Estación de trabajo estándar, basada en x86.
- Tarjeta de red Linksys LNE100TX con una bootrom Etherboot.
- Chipset gráfico basado en Intel i810.
- Servidor corriendo RedHat 7.2.
- DHCP.
- Dirección de la red 192.168.0.0/24.

Asumiendo que el servidor está configurado con el paquete LTSP, esto es lo que sucede:

1. Cuando enciendes la estación de trabajo, la misma pasa a través de su Power On Self Test − POST (autorevisión de encendido).

2. Durante la autorevisión, el BIOS buscará roms de expansión. Cada tarjeta de red posee una bootrom Etherboot, la cual es una rom de expansión. Por lo tanto el BIOS detectará la rom en la placa de red.

3. Una vez que el POST es completado, la ejecución saltará dentro del código Etherboot.

4. El código Etherboot buscará una placa de red. Una vez que la misma es detectada, será inicializada.

5. El código Etherboot realizará luego una petición DHCP mediante un broadcast por la red. La petición incluirá la dirección MAC de la tarjeta de red.

6. El demonio dhcpd del servidor verá la petición de broadcast de la estación de trabajo, y buscará dentro de su propio archivo de configuración la entrada que concuerde con la dirección MAC de la terminal.

7. El demonio dhcpd construirá un paquete de respuesta, conteniendo varias piezas de información. Este paquete será enviado de vuelta a la estación de trabajo. Esta respuesta incluye:

    - Dirección IP de la estación de trabajo.
    - Máscara de red de la red local.
    - Localización del núcleo a descargar.
    - Localización del sistema de archivos raíz a montar.
    - Parámetros opcionales que deben ser pasados al núcleo, a través de la línea de comandos del kernel.

8. El código Etherboot recibirá la respuesta desde el servidor y configurará la interfase TCP/IP en la tarjeta de red con los parámetros que le fueron suministrados.

9. Utilizando TFTP (Trivial File Transfer Protocol), el código Etherboot se contactará con el servidor y comenzará a descargar el núcleo.

10. Una vez que el núcleo fue completamente descargado en la estación de trabajo, el código Etherboot colocará al mismo en la correcta locación de memoria.

11. El control es pasado luego al núcleo, el cual inicializará al sistema completo y a todos los periféricos que reconozca.

12. Aquí es donde la diversión comienza verdaderamente. Hilvanada al final del kernel hay una imagen de un sistema de archivos. Esta imagen es cargada en memoria como un ramdisk (disco de memoria), y es temporariamente montada como el sistema de archivos raíz. Un argumento **root=/dev/ram0** en la línea de comandos del kernel le dice al mismo que debe montar la imagen como su directorio raíz.

13. Normalmente, cuando el núcleo finaliza su inicialización, lanza al programa **init**. Pero, en este caso, hemos instruido al kernel para que cargue un script de shell. Hacemos esto mediando un argumento **init=/linuxrc** en la línea de comandos del núcleo.

14. El script **/linuxrc** comienza rastreando el bus PCI en busca de alguna placa de red. Por cada dispositivo PCI que encuentra, realiza una búsqueda en el archivo /etc/niclist, para ver si hay alguna concordancia. Una vez que una concordancia es encontrada, el nombre del módulo de la interfase de red es retornado para que sea posteriormente cargado en el kernel. Para las placas ISA, el nombre del módulo DEBE estar especificado en la línea de comandos del kernel, junto con cualquier parámetro de IRQ o dirección de memoria que fuera requerido.

15. Una vez que la placa de red ha sido identificada, el script /linuxrc cargará el módulo de kernel que soporta a dicha tarjeta.

16. **dhclient** correrá entonces, para realizar otra petición a el servidor DHCP. Necesitamos hacer esta petición separada debido a que si dependemos de la petición que viene con Etherboot, la misma sería tragada por el kernel, y el mismo ignoraría cualquier servidor NFS que fuera especificado en el root−path (localización del directorio raíz). Esto es importante si deseas que el servidor NFS sea diferente al servidor TFTP.

17. Cuando **dhclient** obtenga una respuesta desde el servidor, correrá el archivo **/etc/dhclient−script**, el cual tomará la información recogida y configurará la interfase eth0.

18. Hasta este punto, el sistema de archivos raíz había sido un disco ram. Ahora, el script /linuxrc montará un nuevo sistema de archivos raíz a través de NFS. El directorio exportado por el servidor es típicamente **/opt/ltsp/i386**. No puede tan sólo montar al nuevo sistema de archivos como /. Debe primero montarlo en /mnt. Luego, ejecutará **pivot_root** (pivotaje del raíz). pivot_root realizará un intercambio del sistema de archivos corriente por uno nuevo. Cuando esto esté completado, el sistema de archivos NFS será montado como /, y el viejo será montado en /oldroot.

19. Una vez completado el montaje y el pivotaje del nuevo sistema de archivos raíz, se acaba el script /linuxrc y un **init** real es invocado.

20. Init leerá el archivo `etc/inittab` y comenzará a configurar el ambiente de la estación de trabajo.

21. Init mantiene la idea de *runlevel* (nivel de corrida). Cada runlevel prepara un conjunto diferente de servicios para la estación de trabajo. La estación LTSP inicia en el runlevel '2'. Esto está predispuesto por la línea *initdefault* del archivo inittab.

22. Uno de los primeros puntos en el archivo inittab es el comando **rc.local** que correrá mientras la estación de trabajo esté en el estado '***sysinit***'.

23. El script rc.local creará un disco ram de 1 MB que contedrá todas las cosas que necesitan ser escritas o modificadas de algún modo.

24. El disco ram será montado en el directorio `/tmp`. Cualquier archivo que necesite ser escrito o modificado existirá realmente en el directorio `/tmp`, con algún link simbólico apuntando hacia él.

25. El sistema de archivos `/proc` es montado.

26. Si la estación de trabajo está configurada para realizar intercambio sobre NFS, el directorio **/var/opt/ltsp/swapfiles** del servidor será montado en /tmp/swapfiles en la estación. Luego, si no hay ningún archivo de intercambio para esta estación de trabajo que haya sido creado con anterioridad, será creado automáticamente. El tamaño de este archivo de intercambio es preconfigurado en el archivo `lts.conf`,

    El archivo de intercambio es habilitado mediante el comando **swapon**.

27. La interfase de red de ***loopback*** es configurada. Este es el dispositivo de red que tiene la dirección IP *127.0.0.1*.

28. Si la opción de Aplicaciones Locales (Local Apps) es habilitada, entonces el directorio **/home** será montado, para que las aplicaciones puedan utilizar los directorios personales de los usuarios.

29. Varios directorios son creados en el sistema de archivos `/tmp` para que contengan algunos de los ficheros transitorios que son necesarios mientras el sistema está corriendo. Algunos son:

    a. `/tmp/compiled`

    b. `/tmp/var`

    c. `/tmp/var/run`

    d. `/tmp/var/log`

    e. `/tmp/var/lock`

    f. `/tmp/var/lock/subsys`

    Todos estos serán creados.

30. El sistema X Window será ahora configurado. En el archivo **lts.conf**, hay un parámetro llamado **XSERVER**. Si este parámetro falta, o está puesto en "**auto**", entonces se intentará una detección automática de la placa de video. Si la misma es PCI, entonces se obtendrá el PCI Vendor y el Device id, y se buscará a los mismos en el archivo **/etc/vidlist**.

    Si la placa es soportada por XFree86 4.X, el script pci_scan devolverá el nombre del módulo controlador. Si por el contrario la placa es sólo soportada por XFree86 3.3.6, pci_scan devolverá el nombre del servidor X a utilizar. El script rc.local puede saber la diferencia debido a que los nombres de los viejos servidores X 3.3.6 comienzan con 'XF86_'.

31. Si XFree86 4.x es utilizado, entonces el script **/etc/rc.setupx** será llamado para construir un archivo XF86Config para X4. Por el contrario, si XFree86 3.3.6 es utilizado, entonces **/etc/rc.setupx3** será llamado para construir el archivo XF86Config.

    El archivo XF86Config será construido en base a las entradas contenidas en **/etc/lts.conf**.

32. Cuando el script rc.setupx finaliza, retornará a rc.local. Luego el script **/tmp/start_ws** será creado. Este script es el responsable de llamar al servidor X.

33. El archivo `/tmp/syslog.conf` será creado. Este archivo contendrá información que le dirá al demonio **syslogd** hacia qué máquina de la red se le debe debe enviar la información de los historiales. Esta máquina es especificada en el archivo `lts.conf`. Hay un link simbólico llamado `/etc/syslog.conf` que apunta hacia `/tmp/syslog.conf`.

34. El demonio **syslogd** es iniciado, usando el archivo de configuración creado en el paso anterior.

35. En control es nuevamente pasado a **init**. Init buscará la entrada ***inittdefault*** para determinar a cuál ***runlevel*** ingresar. Como en lts_core−2.08, el valor por defecto es ***2***.

36. El runlevel ***2*** causará que init corra el script **set_runlevel** el cual leerá el archivo lts.conf para determinar a cuál runlevel la estación de trabajo ingresará de hecho.

    Los runlevels estandar de LTSP son el ***3***, el ***4*** y el ***5***.

    - ***3*** − Este comenzará un shell. Esto es muy útil para depuración de errores.
    - ***4*** − Este correrá una o más sesiones de Telnet en modo de caracteres. Esto es genial si estás reemplazando viejas terminales seriales.
    - ***5*** − Modo GUI (Interfase Gráfica de Usuario). Este iniciará X Window, y enviará una petición XDMCP al servidor, el cual devolverá una caja de diálogo de inicio que te permitirá ingresar al servidor. Necesitarás para esto tener un Display Manager (Administrador de Visualización) escuchando en el servidor, tal como ***XDM***, ***GDM*** o ***KDM***.

Capítulo 2. Instalando LTSP en el servidor

Los paquetes LTSP están disponibles tanto en formato ***RPM*** como en ***TGZ***. Escoge el formato que más te guste, y sigue las instrucciones en la sección apropiada.

Si quieres correr X Window en la estación de trabajo, entonces hay 4 paquetes que necesitarás descargar. Ten en mente que, para los propósitos de este documento, tenemos una estación de trabajo con una tarjeta de red basada en Tulip y una placa de video basada en el chipset Intel i810.

1. Paquete principal LTSP (core package)
2. Paquete de núcleo (kernel package)
3. Paquete principal de X (X core package)
4. Paquete principal de X (X core package)

El paquete de fuentes X no es realmente necesario, pero para una instalación desde cero, es recomendado. Una vez que tengas en claro cómo configurar un servidor y las estaciones de trabajo sin disco, puedes levantar un servidor de fuentes X (XFS).

Luego de que los paquetes hayan sido instalados, el sistema LTSP necesita ser inicializado. Este proceso incluye cambios en los archivos de configuración del sistema para habilitar los servicios necesitados por las estaciones en el servidor.

2.1. Instalando los paquetes RPM

Descarga la última versión de los paquetes ltsp, e instálalos mediante los siguientes comandos:

```
rpm -ivh ltsp_core-3.0.0-1.i386.rpm

rpm -ivh ltsp_kernel-3.0.1-1.i386.rpm

rpm -ivh ltsp_x_core-3.0.1-1.i386.rpm

rpm -ivh ltsp_x_fonts-3.0.0-0.i386.rpm
```

Los comandos arriba mencionados instalarán los paquetes dentro del directorio `/opt/ltsp/i386`.

2.2. Instalando los paquetes TGZ

Descarga la última versión de los paquetes ltsp e instálalos mediante los siguientes comandos:

```
tar xzf ltsp_core-3.0.0-i386.tgz
cd ltsp_core
./install.sh
cd ..

tar xzf ltsp_kernel-3.0.1-i386.tgz
cd ltsp_kernel
./install.sh
cd ..

tar xzf ltsp_x_core-3.0.1-i386.tgz
cd ltsp_x_core
```

```
./install.sh
cd ..

tar xzf ltsp_x_fonts-3.0.0-i386.tgz
cd ltsp_x_fonts
./install.sh
cd ..
```

Los comandos arriba mencionados instalarán los paquetes dentro del directorio `/opt/ltsp/i386`.

2.3. Inicialización del servidor

Una vez que la instalación de los paquetes es completada, necesitas entrar al directorio `/opt/ltsp/templates`. Allí hay varios scripts que configurarán los archivos de sistema de tu servidor. Cada uno de estos scripts es responsable por un archivo de sistema. Mira el interior de estos scripts para asegurarte de que estás de acuerdo con lo que van a hacer, ya que potencialmente pueden hacer tu sistema vulnerable a un ataque de seguridad. Podrías desear realizar los cambios al sistema de forma manual. Si por el contrario quieres hacerlo automáticamente, entonces corre el comando ltsp_initialize:

```
cd /opt/ltsp/templates
./ltsp_initialize
```

El comando de arriba te preguntará acerca de cuáles servicios quieres que configure. El mismo puede inicializar a los siguientes :

- XDM − X Display Manager
- GDM − Gnome Display Manager
- Script de inicio del Display Manager
- bootp
- NFS, archivo /etc/exports
- tcpwrappers (archivos /etc/hosts.allow, /etc/hosts.deny)
- Port mapper
- syslogd
- Script de inicio de TFTP

2.4. Configuración específica de la estación de trabajo

Ahora, es tiempo de contarle al servidor LTSP acerca de tu estación de trabajo. Hay tres archivos que contienen información acerca de la misma.

1. /etc/dhcpd.conf
2. /etc/hosts
3. /opt/ltsp/i386/etc/lts.conf

2.4.1. /etc/dhcpd.conf

La estación de trabajo necesita una dirección IP y otra información. Obtendrá a través de un servidor DHCP lo siguiente:

- Dirección IP
- Nombre de host

- Dirección IP del servidor
- Pasarela por defecto
- Localización del núcleo a cargar
- Servidor y directorio a montar como el sistema de archivos raíz

Para nuestro ambiente LTSP de ejemplo, hecmos elegido DHCP para el manejo de las direcciones IP a las estaciones de trabajo.

Durante el script ltsp_initialize, un archivo dhcpd.conf de muestra es instalado, llamado `/etc/dhcpd.conf.example`. Puedes copiar al mismo en `/etc/dhcpd.conf` para usarlo como base para tu configuración de dhcp. Probablemente necesitarás modificar las partes correspondientes a tu estación de trabajo específica y a tu ambiente en particular.

```
default-lease-time            21600;
max-lease-time                21600;

option subnet-mask            255.255.255.0;
option broadcast-address      192.168.0.255;
option routers                192.168.0.254;
option domain-name-servers    192.168.0.254;
option domain-name            "ltsp.org";
option root-path              "192.168.0.254:/opt/ltsp/i386";

shared-network WORKSTATIONS {
    subnet 192.168.0.0 netmask 255.255.255.0 {
    }
}

group   {
    use-host-decl-names       on;
    option log-servers        192.168.0.254;

    host ws001 {
        hardware ethernet     00:E0:18:E0:04:82;
        fixed-address         192.168.0.1;
        filename              "/lts/vmlinuz.ltsp";
    }
}
```

Figura 2−1. /etc/dhcpd.conf

Como en LTSP versión 2.09pre2, no necesitas especificar un kernel en particular para cargar. El paquete estándar de núcleo soporta todas las placas de red que Linux soporta. Hay dos archivos de kernel incluidos en el paquete LTSP−kernel. Uno de los archivos tiene el Linux Progress Patch (parche que muestra una barra de progreso gráfica) y el otro no. Los nombre de estos núcleos son:

```
vmlinuz-2.4.9-ltsp-5
vmlinuz-2.4.9-ltsp-lpp-5
```

Puede que hayas notado que el kernel sigue estando ubicado en el directorio `/tftpboot/lts`, pero en la entrada "filename" del archivo `/etc/dhcpd.conf` no está el `/tftpboot` del comienzo. Esto es así debido a que en versiones de RedHat superiores a 7.1 el TFTP corre con la opción '−s', lo que causa que el demonio tftpd corra en modo ***seguro***, Esto significa que realiza un **chroot** al directorio `/tftpboot` cuando comienza. Por lo tanto, todos los archivos que están disponibles para el demonio tftpd son relativos al directorio `/tftpboot`.

Otras distribuciones de Linux pueden no tener la opción '−s' activa para tftpd, por lo que puedes necesitar agregar el prefijo `/tftpboot` a la localización del kernel en el archivo /etc/dhcpd.conf.

2.4.2. /etc/hosts

Mapeo de direcciones IP a nombres de host

Las computadoras generalmente se comunican bastante bien solo con direcciones IP. Pero los humanos venimos y necesitamos ponerles nombres a los ordenadores, debido a que no podemos recordar los números. Ahí es cuando el DNS o el archivo `/etc/hosts` entra escena. Este mapeo de direcciones IP a nombres de host no es requerido generalmente, excepto en un ambiente LTSP. Esto es así porque sin este mapeo, el NFS dará errores de permisos cuando la estación de trabajo intente montar el sistema de archivos raíz.

En adición a los problemas de NFS, si la estación de trabajo no se encuentra en el archivo `/etc/hosts`, también podrías tener problemas con los Display Managers ***GDM*** y ***KDM***.

2.4.3. /opt/ltsp/i386/etc/lts.conf

Hay un número de entradas de configuración que son especificados en el archivo lts.conf.

El archivo `lts.conf` tiene una sintaxis simple, que consiste de múltiples secciones. Hay una sección por defecto, llamada **[default]** y hay secciones adicionales para estaciones de trabajo individuales. Las estaciones de trabajo pueden ser identificadas por nombre de host, dirección IP o dirección MAC.

Un archivo típico `lts.conf` luce así:

```
#
# Config file for the Linux Terminal Server Project (www.ltsp.org)
#

[Default]
        SERVER             = 192.168.0.254
        XSERVER            = auto
        X_MOUSE_PROTOCOL   = "PS/2"
        X_MOUSE_DEVICE     = "/dev/psaux"
        X_MOUSE_RESOLUTION = 400
        X_MOUSE_BUTTONS    = 3
        USE_XFS            = N
        LOCAL_APPS         = N
        RUNLEVEL           = 5

[ws001]
        USE_NFS_SWAP       = Y
        SWAPFILE_SIZE      = 48m
        RUNLEVEL           = 5

[ws002]
        XSERVER            = XF86_SVGA
        LOCAL_APPS         = N
        USE_NFS_SWAP       = Y
        SWAPFILE_SIZE      = 64m
        RUNLEVEL           = 3
```

Ejemplo 2−1. lts.conf file

La siguiente es una lista de algunas entradas:

XSERVER

Si tu placa de video es PCI y está soportada por XFree86 4.1, entonces sólo necesitas el paquete lts_x_core. Contiene todos los módulos para X4.

Por el contrario, si tu tarjeta no está soportada por XFree86 4.1, hay varios paquetes XFree86 3.3.6 disponibles para LTSP.

Puedes hacer entradas en el archivo `lts.conf` para cada estación de trabajo individual, o puedes realizar entradas por defecto para que sean compartidas por todas las estaciones de trabajo.

Nuestra terminal tiene un chipset de video Intel i810, y puede ser detectado correctamente, por lo que no necesitamos ninguna entrada XSERVER en lts.conf. Esta entrada XSERVER puede ser especificada, si lo deseas, o puede ser puesta en 'auto', si es que quieres dejar constancia de que la placa de hecho será detectada automáticamente.

RUNLEVEL

Queremos correr la estación de trabajo en modo gráfico, por lo que necesitamos configurar el runlevel en '5'. Esto es efectuado mediante otra entrada en el archivo `lts.conf`.

Capítulo 3. Configurando la estación de trabajo

Una vez que el servidor ha sido configurado, es tiempo de concentrarse en la estación de trabajo.

El proyecto LTSP trata acerca de qué es lo que sucede cuando el kernel está en memoria. Hay varias maneras de hacer que un núcleo se cargue en memoria, incluyendo Etherboot, Netboox, PXE y disco floppy.

Para el objeto de este documento, usaremos un disco floppy con código del proyecto **Etherboot**.

3.1. Creando el disco de inicio

> Etherboot es un paquete de software para la creación de imágenes ROM que puedan descargar código a través de una red Ethernet y luego ejecutarlo en un computador x86. Muchos dispositivos de red poseen un sócalo donde un chip ROM puede ser instalado. Etherboot es código que puede ser colocado en dicha ROM.
>
> −−Ken Yap

Etherboot es también Open Source, protegido bajo la Licencia Pública General GNU, Versión 2 (GPL2).

Puedes descargar el paquete Etherboot y configurarlo para el tipo de bootrom que necesites. Luego, puedes compilar el fuente para producir la imagen que será escrita en el chip EPROM, o bien en un diskette para pruebas.

Una alternativa mucho más simple es ir al sitio web de Marty Connor *www.Rom−O−Matic.net*.

Marty a realizado un excelente trabajo al ponerle un frente basado en web al proceso de configuración y compilación de las imagenes de bootrom con Etherboot. En su sitio, puedes elegir qué tipo de tarjeta de red posees, y qué clase de imagen quieres. Luego, tienes la oportunidad de modificar cualquiera de las opciones de configuración de Etherboot. Por último, presionando el botón 'Get ROM' una imagen bootrom personalizada será generada para tí.

Nuestra estación de trabajo tiene una tarjeta Linksys LNE100TX, versión 4.1. Esta placa posee un chipset ADMTek Centaur−P, por lo que necesitamos seleccionar ***centaur−p*** como nuestro tipo de NIC/ROM.

No necesitamos realizar ningún cambio a la configuración por defecto, por lo que podemos saltearnos el botón 'Configure'.

Para el 'ROM Output Format' (formato de salida de la ROM) seleccionamos 'Floppy Bootable ROM Image' (imagen ROM para discos blandos). Esto causará que contenga una cabecera de 512 bytes que es el cargador de inicio para cargar la imagen etherboot en la memoria RAM, donde será ejecutada.

Presiona el botón 'Get ROM<'. La imagen bootrom será generada mientras esperas. Sólo toma unos pocos segundos, y cuando está listo, tu navegador desplegará una ventana de 'Guardar Como' donde puedes designar dónde quiere salvar la imagen.

Yo usualmente salvo la imagen en el directorio `/tmp`, por lo que, para la imagen Centaur−P, debes especificar algo como: `/tmp/eb−5.0.2−centaur−p.lzdsk`.

Una vez que has salvado la imagen en tu disco duro, necesitas grabarla dentro de un disco flexible. Inserta uno de estos discos en tu disquetera y corre el siguiente comando:

```
cat   /tmp/eb−5.0.2−centaur−p.lzdsk   > /dev/fd0
```

Capítulo 4. Corriendo la estación de trabajo

Asumiendo que el servidor y la estación de trabajo han sido configurados correctamente, debería ser sólo una cuestión de insertar el disco flexible en la disquetera y encender la estación de trabajo.

El código Etherboot será leído desde el disquete y cargado en memoria, la tarjeta de red será encontrada e inicializada, la petición dhcp será enviada por la red y una respuesta será enviada desde el servidor y el kernel será descargado en la estación de trabajo. Una vez que el núcleo ha inicializado el hardware de la estación de trabajo, X Window arrancará y una ventana de diálogo de inicio debería aparecer en la terminal, similar al ejemplo de abajo.

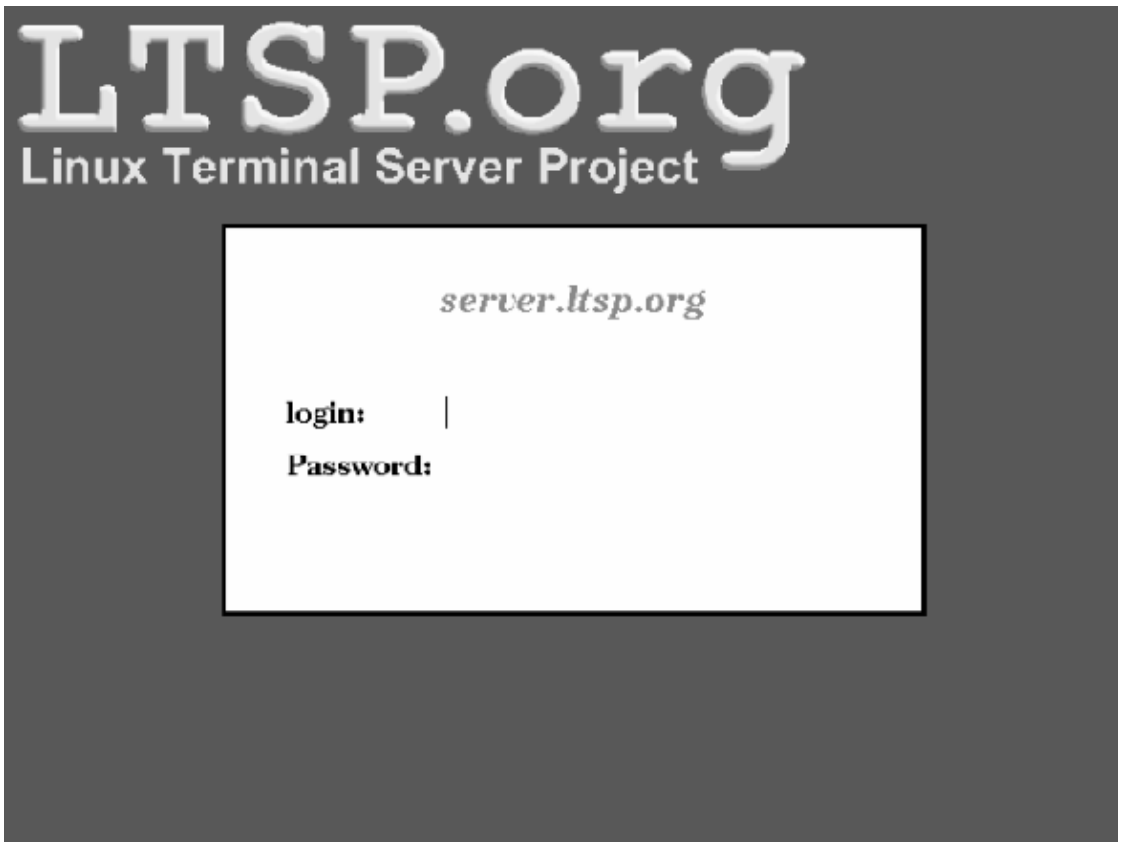

Figura 4−1. Login screen

En este punto, puedes autentificarte. Una cosa importante es tener en cuenta de que hecho estás ingresando en el servidor. Todos los comandos que corras de hecho estarás corriendo en el mismo, mostrando su salida por el monitor de la estación de trabajo. Ese es el poder de X Window.

Puedes correr cualquier programa soportado por el servidor.

Capítulo 5. Imprimiendo

Mas allá del hecho de que una estación de trabajo puede poseer un entorno gráfico completo o ser una terminal de caracteres funcional, también puede actuar como un servidor de impresión, permitiendo que hasta 3 impresoras puedan ser conectadas a sus puertos seriales o paralelos.

Todo esto es transparente para el usuario de la estación de trabajo. Ni siquiera notará la pequeña cantidad de tráfico que es enviada a través de la terminal hacia la impresora.

5.1. Configuración del lado del cliente

LTSP utiliza el programa **lp_server** en la estación de trabajo para redireccionar los trabajos de impresión desde el servidor hacia la impresora conectada a alguno de los puertos de la estación.

Para habilitar la impresora en la estación de trabajo, hay un conjunto de entradas de configuración en el archivo **lts.conf**.

```
[ws001]
    PRINTER_0_DEVICE = /dev/lp0
    PRINTER_0_TYPE   = P
```

La entrada de arriba causará que el programa lp_server corra como demonio, escuchando en el puerto TCP/IP 9100 que llegue alguna corriente de datos de impresión desde el servidor. Estos datos serán luego redireccionados hacia la impresora conectada en el puerto paralelo /dev/lp0.

Hay muchas opciones disponibles. Mira la sección lts.conf más adelante en este documento para obtener mayor información acerca de las entradas de configuración.

5.2. Configuración del lado del servidor

Configurar una impresora del lado del servidor es cuestión de definir una cola de impresión, utilizando la herramienta de configuración apropiada en el servidor.

En RedHat 7.2, hay herramientas tanto gráficas como de texto para configurar impresoras. La herramienta gráfica se llama **printconf−gui**, y la basada en text **printconf−tui**. Versiones mas antiguas de RedHat poseen un programa denominado **printtool**. Printtool también existe en RedHat 7.2, pero llamará a printconf−gui. Otras distribuciones de Linux tienen sus propias herramientas de configuración de impresoras.

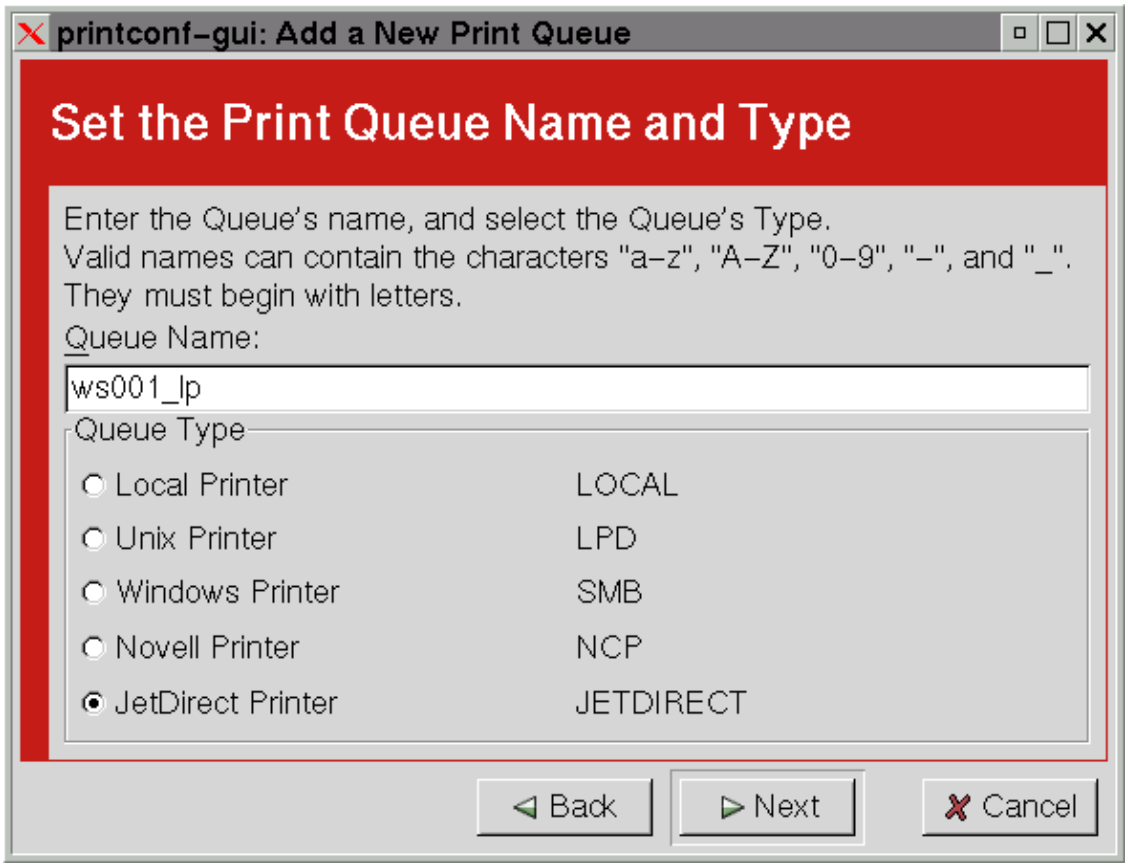

Figura 5−1. Printconf−gui Agregando una nueva impresora

Una vez que has lanzado la herramienta de configuración, debes agregar una nueva impresora. El programa lp_server le permite a la estación de trabajo emular un servidor de impresión HP JetDirect, por lo que sólo necesitas crear una impresora **JetDirect**.

Necesitas luego darle a la cola un nombre. Este nombre puede ser cualquier cosa (trata de que sea un nombre significativo) y debe contener sólo los siguientes caracteres:

- `"a-z"` letras en minúsculas
- `"A-Z"` letras en mayúsculas
- `"0-9"` dígitos numéricos
- `"-"` guión medio
- `"_"` guión bajo

El nombre seleccionado en el ejemplo de arriba es **ws001_lp**. Este nombre hace que sea fácil saber que la impresora está asociada con **ws001**.

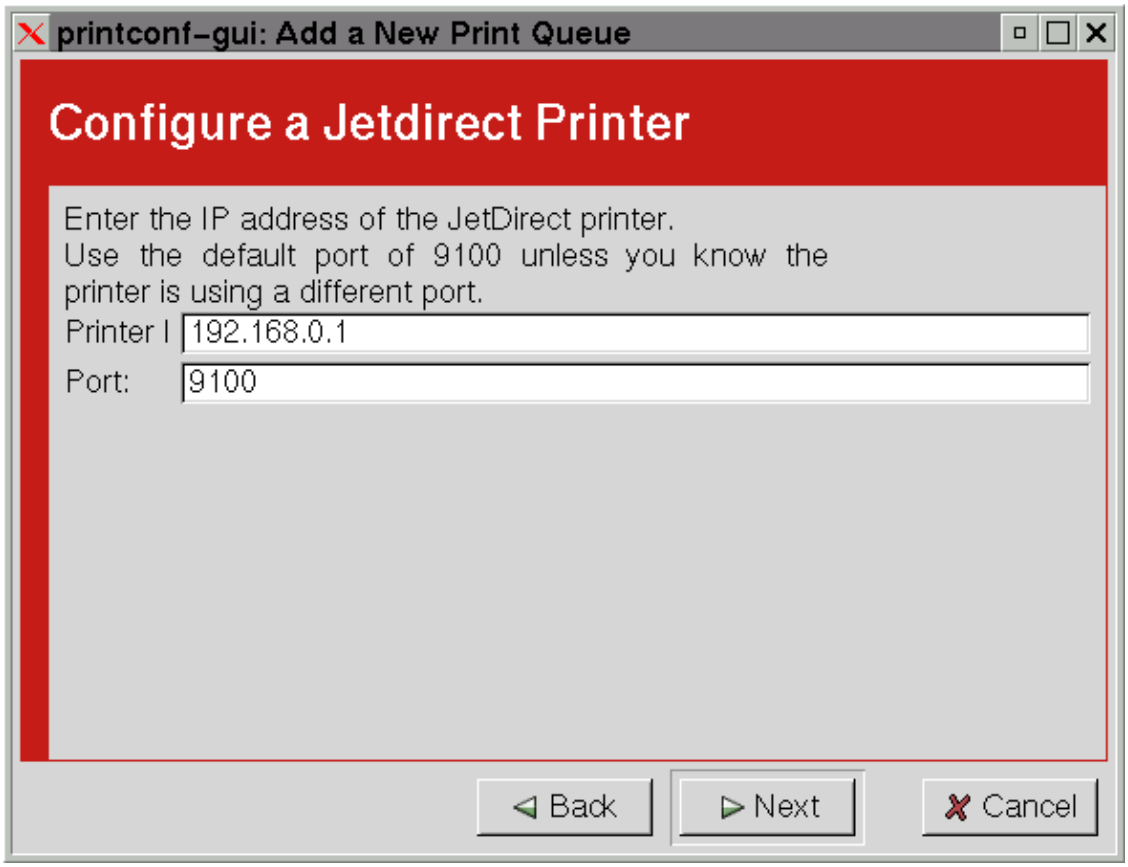

Figura 5−2. Printconf−gui Información en detalle

Hay dos campos necesarios para comunicarse con la impresora:

1. La dirección IP y el nombre de host de la estación de trabajo a la cual la impresora está asociada.
2. El puerto TCP en el cual el demonio **lp_server** está corriendo.

   La primera impresora que conectes a la estación de trabajo estará en el puerto TCP/IP **9100**. La segunda lo hará en el puerto **9101** y la tercera en el **9102**.

Capítulo 6. Solución de problemas

Si, luego de haber seguido los anteriores pasos, tu estación de trabajo no inicia, entonces deberás empezar el proceso de detección de problemas de la instalación.

La primer cosa por hacer es tratar de ver cuán lejos ha llegado la estación de trabajo en su proceso de inicio.

6.1. Solucionando problemas con la imagen Etherboot para disco flexible

Cuando inicias desde un disco flexible, debería ver algo similar a esto:

```
loaded ROM segment 0x0800 length 0x4000 reloc 0x9400
Etherboot 5.0.1 (GPL) Tagged ELF for [LANCE/PCI]
Found AMD Lance/PCI at 0x1000, ROM address 0x0000
Probing...[LANCE/PCI] PCnet/PCI−II 79C970A base 0x1000, addr 00:50:56:81:00:01
Searching for server (DHCP)...
<sleep>
```

El ejemplo de arriba muestra lo que puedes esperar ver en la pantalla cuando inicias desde un disco flexible. Si no ves estos mensajes, indicando que Etherboot a iniciado, entonces puede ser que tengas un disco flexible defectuoso, o que hayas escrito la imagen en el mismo de manera inapropiada.

Si, por el contrario, ves una imagen como la siguiente, entonces es probable que la imagen Etherboot generada no sea la apropiada para tu tarjeta de red.

```
ROM segment 0x0800 length 0x8000 reloc 0x9400
Etherboot 5.0.2 (GPL) Tagged ELF for [Tulip]
Probing...[Tulip]No adapter found
<sleep>
<abort>
```

Si llega al punto en donde detecta a la tarjeta de red y muestra la dirección MAC, entonces el probable que el disco flexible sea bueno.

6.2. Solucionando problemas de DHCP

Una vez que la tarjeta de red es inicializada, enviará una petición de broadcast a través de la red local, buscando algún servidor DHCP.

Si la estación de trabajo obtiene una respuesta válida de parte del servidor DHCP, deberá configurar la placa de red. Puedes saber si ha funcionado correctamente si la información acerca de la dirección IP es mostrada por la pantalla. Aquí hay un ejemplo de cómo debería verse esto:

```
ROM segment 0x0800 length 0x4000 reloc 0x9400
Etherboot 5.0.1 (GPL) Tagged ELF for [LANCE/PCI]
Found AMD Lance/PCI at 0x1000, ROM address 0x0000
Probing...[LANCE/PCI] PCnet/PCI−II 79C970A base 0x1000, addr 00:50:56:81:00:01
Searching for server (DHCP)...
<sleep>
Me: 192.168.0.1, Server: 192.168.0.254, Gateway 192.168.0.254
```

Si ves la línea que comienza con 'Me:', seguida por una dirección IP, entonces sabes que DHCP está funcionando correctamente. Puedes continuar ahora para saber si TFTP está de hecho funcionando también.

Si, en cambio, observas el siguiente mensaje en la estación de trabajo, seguida por montones de <sleep>, entonces algo está mal, aunque es común ver uno o dos de estos mensajes de <sleep>.

```
Searching for server (DHCP)...
```

Tratar de averiguar qué es lo que está mal a veces es dificultoso, pero aquí hay algunas cosas en las cuales fijarse.

6.2.1. Verificar las conexiones

¿Está la estación de trabajo fisicamente conectada a la misma red del servidor?

Con la estación de trabajo encendida, asegúrate de que las luces de link están encendidas para todas las conexiones.

Si estás conectando directamente entre la estación de trabajo y el servidor (sin hub ni switch), asegúrate de estar utilizando un cable cruzado. Si estás de hecho utilizando un hub o switch, entonces asegúrate de utilizar cable recto, normal, tanto entre las estaciones de trabajo y el hub como entre el hub y el servidor.

6.2.2. ¿Está corriendo dhcpd?

Necesitas determinar si **dhcpd** está corriendo en el servidor. Podemos obtener la respuesta de dos formas.

dhcpd normalmente corre en el background (de fondo), escuchando en el puerto udp 67. Trata de correr el comando **netstat** para ver si hay algo escuchando en dicho puerto:

```
netstat −an | grep ":67 "
```

Deberías ver una salida similar a la siguiente:

```
udp     0    0   0.0.0.0:67         0.0.0.0:*
```

La cuarta columna contiene la dirección IP y el puerto, separadas por dos puntos (':'). Una dirección de todos ceros ('0.0.0.0') indica que está escuchando en todas las interfases. Esto es, puede ser que tengas una interfase **eth0** y otra **eth1**, y que **dhcpd** esté escuchando en ambas.

Sólo porque netstat muestre que hay algo escuchando en el puerto udp 67 no significa que de hecho **dhcpd** sea el que está escuchando. Podría ser **bootpd**, aunque sería poco probable, ya que **bootp** ya no se incluye más en la mayoría de las distribuciones de Linux.

Para asegurarte de que **dhcpd** está corriendo, trata con el comando **ps**.

```
ps aux | grep dhcpd
```

Deberías ver algo como lo siguiente:

```
root 23814 0.0 0.3 1676 820 ?      S 15:13 0:00 /usr/sbin/dhcpd
root 23834 0.0 0.2 1552 600 pts/0  S 15:52 0:00 grep dhcp
```

La primera línea muestra que **dhcpd** está corriendo. La segunda es tan sólo nuestro comando **grep**.

Si no ves ninguna línea indicando que dhcpd está corriendo, entonces necesitas verificar que el servidor esté configurado para el runlevel 5 y que **dhcpd** esté listo para iniciar en dicho nivel. En sistemas basados en Red Hat, puedes ejecutar **ntsysv** para asegurarte de que **dhcpd** esté marcado.

Puedes probar de iniciar **dhcpd** con este comando:

```
service dhcpd start
```

Presta atención a la salida. Puede contener errores.

6.2.3. Verifica dos veces la configuración de dhcpd

¿Tiene el archivo `/etc/dhcpd.conf` una entrada para tu estación de trabajo?

Deberías verificar dos veces la entrada 'fixed−address' en el archivo de configuración, para asegurarte de que concuerda exactamente con la tarjeta de red de la estación de trabajo.

6.2.4. ¿Está ipchains o iptables bloqueando el pedido?

6.2.4.1. Verificando ipchains

Corre el siguiente comando para ver qué dice:

```
ipchains −L −v
```

Si ves algo como esto:

```
Chain input (policy ACCEPT: 229714 packets, 115477216 bytes):
Chain forward (policy ACCEPT: 10 packets, 1794 bytes):
Chain output (policy ACCEPT: 188978 packets, 66087385 bytes):
```

Entonces no es ipchains el que está metido en el camino.

6.2.4.2. Verificando iptables.

Corre el siguiente comando para ver qué dice:

```
iptables −L −v
```

Si ves algo como esto:

```
Chain INPUT (policy ACCEPT 18148 packets, 2623K bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 17721 packets, 2732K bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Entonces no es iptables el que está metido en el camino.

6.2.5. ¿Está la estación de trabajo enviando la petición?

Trata de observar `/var/log/messages` mientras la estación de trabajo se inicia. Puedes hacer tal cosa con el siguiente comando:

```
tail −f /var/log/messages
```

Esto 'seguirá' al archivo de historiales mientras se vayan escribiendo los nuevos registros.

```
server dhcpd: DHCPDISCOVER from 00:50:56:81:00:01 via eth0
server dhcpd: no free leases on subnet WORKSTATIONS
server dhcpd: DHCPDISCOVER from 00:50:56:81:00:01 via eth0
server dhcpd: no free leases on subnet WORKSTATIONS
```

Si ves mensajes como los de arriba, donde dice 'no free leases', entonces **dhcpd** está corriendo, pero no sabe nada acerca de la estación de trabajo que hace el pedido de su dirección IP.

6.3. Solucionando problemas de TFTP

Etherboot utiliza TFTP para obtener un kernel Linux desde el servidor. Este es un protocolo en extremo simple, pero a veces hay problemas cuando se trata de hacerlo funcionar.

Si observas algo similar a esto:

```
Loading 192.168.0.254:/lts/vmlinuz.tulip |
```

con el último caracter en la línea (el '|'), rápidamente cambiando a '|', '\', '−' y '/', para formar lo que parece una barra rotativa, entonces el núcleo está siendo descargado. Esto normalmente indica que TFTP está funcionando correctamente.

Si, en cambio, no puedes ver la barra rotativa, entonces hay un problema. Las posibles causas incluyen:

6.3.1. tftpd no está corriendo

En RedHat 7.1, tftp es iniciado por **xinetd**. Hay un script de inicio llamado `/etc/xinetd.d/tftp` que contiene la información necesaria para que corra **tftpd**.

6.3.2. Es núcleo no está donde tftpd espera encontrarlo.

El kernel nececita estar en un lugar accesible para el demonio tftpd. Si la opción '−s' está especificada cuando **tftpd** inicia, entonces cualquier cosa que pida la estación de trabajo debe ser relativa a `/tftpboot`. Por lo tanto, si la entrada **filename** en `/etc/dhcpd.conf` es `/lts/vmlinuz−2.4.9−ltsp−5`, entonces el kernel de hecho necesita estar en `/tftpboot/lts/vmlinuz−2.4.9−ltsp−5`.

6.4. Solucionando problemas del sistema de archivos raíz NFS

Hay varias cosas que pueden impedir que un sistema de archivos raíz pueda ser montado, incluyendo a las siguientes:

6.4.1. No init found (no se encontró Init)

Si obtienes el siguiente error:

```
Kernel panic: No init found.  Try passing init= option to kernel.
```

Entonces es muy probable que o bien estés montando el directorio incorrecto como sistema de archivos raíz, o bien el directorio `/opt/ltsp/i386` esté vacío.

6.4.2. El servidor retona el error −13

Si obtienes el siguiente error:

```
Root−NFS: Server returned error −13 while mounting /opt/ltsp/i386
```

Esto indica que el directorio `/opt/ltsp/i386` no está listado en el archivo `/etc/exports`.

Mira en `/var/log/messages` si hay algunas pistas. Una entrada como ésta:

```
Jul 20 00:28:39 jamlap rpc.mountd: refused mount request from ws004
                  for /opt/ltsp/i386 (/): no export entry
```

Confirma nuestra sospecha de que la entrada en `/etc/exports` es incorrecta.

6.4.3. Problemas con el demonio NFS (portamp, nfsd y mountd)

NFS puede ser un servicio difícil y complejo para la resolución de problemas, pero el hecho de entender qué debería estar configurado y qué herramientas hay disponibles para diágnósticos sin duda ayudarán a hacer más fácil la tarea.

Hay tres demonios que necesitan estar corriendo en el servidor para que NFS funcione correctamente. Estos servicios son **portmap**, **nfsd** y **mountd**.

6.4.3.1. El portmapper (portmap)

Si obtienes los siguientes mensajes:

```
Looking up port of RPC 100003/2 on 192.168.0.254
portmap: server 192.168.0.254 not responding, timed out
Root−NFS: Unable to get nfsd port number from server, using default
Looking up port of RPC 100005/2 on 192.168.0.254
portmap: server 192.168.0.254 not responding, timed out
Root−NFS: Unable to get mountd port number from server, using default
mount: server 192.168.0.254 not responding, timed out
Root−NFS: Server returned error −5 while mounting /opt/ltsp/i386
VFS: unable to mount root fs via NFS, trying floppy.
VFS: Cannot open root device "nfs" or 02:00
Please append a correct "root=" boot option
Kernel panic: VFS: Unable to mount root fs on 02:00
```

Entonces es muy probable que el demonio **portmap** no esté corriendo. Pueder confirmar si el portmapper está o no funcionando mediante el comando **ps**:

```
ps −e | grep portmap
```

Si el portmapper está andando, debería ver una salida como ésta:

```
30455 ?        00:00:00 portmap
```

Otra forma de provar es mediante el uso de **netstat**. El portmapper utiliza los puertos TCP y UDP 110. Prueba corriendo esto:

```
netstat −an | grep ":111 "
```

Deberías observar la siguiente salida:

```
tcp   0   0 0.0.0.0:111       0.0.0.0:*          LISTEN
udp   0   0 0.0.0.0:111       0.0.0.0:*
```

Si no ves algo similar, entonces el portmapper no está corriendo. Puedes iniciarlo mediante:

```
/etc/rc.d/init.d/portmap   start
```

Luego, puedes asegurarte de que el portmapper está configurado para que inicie automáticamente cuando el servidor reinicie. Corre **ntsysv** para asegurarte de que está seleccionado.@

6.4.3.2. Demonios NFS y MOUND (nfsd y mountd)

NFS tiene dos demonios que necesitan estar corriendo: **nfsd** y **mountd**. Ambos son iniciados por el script `/etc/rc.d/init.d/nfs`.

Puedes ejecutar el comando **ps** para asegurarte de que estén funcionando:

```
ps −e | grep nfs
ps −e | grep mountd
```

Si observas que alguno de los dos demonios no está corriendo, entonces necesitarás hacerlo tu mismo.

Normalmente, deberías poder correr el script de inicio con el argumento **restart** para que ambos demonios se reinicien, pero por alguna razón, el script `/etc/rc.d/init.d/nfs` no reinicia **nfsd** de ese modo. Sólo reinicia **mountd** (¿bug?). Por lo tanto, debes ejecutar la siguiente secuencia de comandos:

```
/etc/rc.d/init.d/nfs  stop
/etc/rc.d/init.d/nfs  start
```

Puede ser que obtengas errores durando el comando **stop**, pero esto es normal. Es el **start** el que debería estar todo **OK**.

Si los demonios están corriendo, pero NFS sigue sin funcionar, entonces pueder verificar que los mismos se hayan registrado a sí mismo con el portmapper mediante el comando **rpcinfo**.

```
rpcinfo −p localhost
```

Debería mostrarte resultados similares a los siguientes:

```
program vers proto   port
 100000    2   tcp    111  portmapper
 100000    2   udp    111  portmapper
 100011    1   udp    856  rquotad
 100011    2   udp    856  rquotad
 100005    1   udp   1104  mountd
```

```
100005    1   tcp   2531  mountd
100005    2   udp   1104  mountd
100005    2   tcp   2531  mountd
100003    2   udp   2049  nfs
```

Esto indica que **nfs** (nfsd) y **mountd** están corriendo y han sido registrados con el portmapper.

6.5. Solucionando problemas con el servidor X

¡Muchacho! Probablemente la parte más dificultosa de la configuración de una estación de trabajo LTSP sea que el servidor X esté configurado correctamente. Si estás usando una placa de video medianamente nueva, y está soportada por XFree86, y si también posees un monitor medianamente nuevo que pueda manejar un amplio rango de frecuencias y resoluciones, entonces todo es muy fácil y derecho. Usualmente, en ese caso, si no funciona, es muy probable que sea el servidor X equivocado para dicha placa.

Cuando un servidor X no funciona con tu placa, es bastante obvio. O bien el mismo no inicia, o bien lo que se ve en pantalla es incorrecto.

Cuando la estación de trabajo está lista para iniciar el servidor X, llama al script `/tmp/start_ws`, el cual inicia al servidor X en la estación de trabajo local con la opción **−query** apuntanto al servidor, donde un Display Manager, como **XDM**, **GDM** o **KDM** está escuchando.

Debido a que el servidor X es iniciado por el script `start_ws`, el cual es asi mismo arrancado por el programa **init**, cuando falla, **init** tratará de iniciarlo nuevamente, y seguirá intentándolo 10 veces más antes de rendirse, debido a que piensa que está repitiendo demasiando rápido. Cuando finalmente se rinde, un mensaje de error de parte del servidor X será mostrado por pantalla.

Esperar que el servidor X falle 10 veces puede ser irritante, por lo que una manera simple de evitar tales fallos repetidos puede ser iniciar la terminal en el runlevel 3, para que X NO sea iniciado automáticamente. En cambio, cuando inicies la estación de trabajo, verás un prompt de **bash**. Desde el mismo, pueder iniciar X manualmente mediante el siguiente comando:

```
sh  /tmp/start_ws
```

El servidor X intentará iniciar, pero cuando falle, volverá al prompt de bash, para que puedas ver la razón de tal falla.

6.6. Solucionando problemas del Display Manager

El Display Manager (Administrador de Sesiones Gráficas) es el demonio que corre en el servidor, esperando que un servidor X haga contacto con él. Una vez que el contacto ha sido realizado, mostrará una caja de diálogo de inicio por pantalla, ofreciéndole al usuario la chance de ingresar al servidor.

Los tres Display Managers mas comunes son:

- XDM − Ha estado por ahí desde siempre. Viene incluido con el sistema estandar de X Window.
- GDM − El 'Gnome Display Manager' es parte del paquete Gnome.
- KDM − El 'KDE Display Manager' es parte de sistema de escritorio KDE.

Las distribuciones de GNU/Linux mas recientes incluyen a los tres arriba mencionados.

6.6.1. Pantalla gris con un gran cursor en X

Esto indica que el servidor X está corriendo, pero que no ha podido realizar contacto con el Display Manager. Algunas de las posibles razones son:

1. El display manager no está corriendo.

   En versiones recientes de Red Hat (7.0 o superior), el display manager es iniciado por **init**. En el archivo `/etc/inittab`, hay una línea que luce así:

   ```
   x:5:respawn:/etc/X11/prefdm −nodaemon
   ```

   El script **prefdm** determinará cuál es el display manager a correr.

   El display manager por defecto depende de qué paquetes tengas instalados. Si Gnome está instalado, entonces GDM será el display manager por defecto. Por el contrario si Gnome no está instalado, entonces el script prefdm verificará si KDE lo está. Si así es, entonces KDM será el display manager. Si KDE tampoco está instalado, entonces XDM será el display manager por omisión.

   Utilizando el comando **netstat** deberías poder ver si hay algún Display Manager corriendo. En el servidor, corre lo siguiente:

   ```
   netstat −ap | grep xdmcp
   ```

   Esto debería mostrarte que hay algún proceso escuchando en el puerto de xdmcp (177).

   ```
   udp      0   0 *:xdmcp             *:*                 1493/gdm
   ```

   Esto muestra claramente que **gdm** está corriendo con el PID 1493 y está escuchando en el puerto de xdmcp.

   Si ves una línea como la de arriba, indicando que definitivamente hay un display manager escuchando, entonces necesitas asegurarte de que la estación de trabajo está enviando la petición XDMCP al servidor correcto.

   En el archivo `lts.conf` hay una entrada en la cual puedes especificar la dirección IP del servidor en el que está corriendo el display manager. Esta entrada es opcional, pero de estar presente, luciría así:

   ```
   XDM_SERVER  =  192.168.0.254
   ```

   Por supuesto, la dirección IP del ejemplo puede ser diferente a la tuya.

   Si la entrada 'XDM_SERVER' no está presente, entonces se utilizará el valor de la entrada 'SERVER', de estar presente también. Si no lo está, se usará **192.168.0.254** por omisión.

   Cualquiera sea el modo en el cual lo especifiques, solamente debes asegurarte de que la dirección IP de hecho sea la dirección correcta del servidor en el que corre el display manager.

2. El display manager puede estar configurado para ignorar peticiones de máquinas remotas.

   Si has determinado que el display manager está corriendo, entonces es posible que haya sido configurado para ignorar peticiones XDMCP desde máquinas remotas. Necesitarás verificar los

archivos de configuración del display manager en particular que estés utilizando para ver si está configurado correctamente.

- **XDM**

  La configuración por defecto de RedHat deshabilita a las estaciones de trabajo para obtener su pantalla de inicio a través de XDM. El script **ltsp_initialize** se hará cargo de habilitar esto por ti, pero si aun así no funciona, entonces debería verificar el archivo `/etc/X11/xdm/xdm−config`. Busca una entrada como esta:

  ```
  DisplayManager.requestPort:     0
  ```

  Esta entrada DEBE estar comentada (con un '#'; delante) para que XDM escuche en el puerto 177 las peticiones remotas.

  Otro archivo de configuración importante para XDM es el `/etc/X11/xdm/Xaccess`, el cual DEBE tener una línea que comience con un '*' y nada más. Normalmente en RedHat hay una línea así, pero está comentada. El script **ltsp_initialize** se encargará de corregir esto por tí, pero si aun así no funciona, deberás verificarlo tu mismo. La línea debería verse así:

  ```
  *        #any host can get a login window
  ```

- **KDM**

  Las versiones más recientes de KDM poseen un archivo llamado **kdmrc**. Diferentes distribuciones de GNU/Linux guardan dicho archivo en lugares diferentes. En RedHat 7.2 está ubicado en **/etc/kde/kdm/kdmrc**. En otras distribuciones, deberías correr el comando **locate** para averiguar dónde se encuentra.

  La entrada que controla si las estaciones de trabajo pueden obtener una pantalla de inicio está en la sección que posee la etiqueta **[xdmcp]**. Asegúrate de que la entrada **Enable** esté puesta en **true**.

  Versiones mas viejas de KDM utilizan los archivos de configuración del XDM, ubicados en /etc/X11/xdm.

- **GDM**

  GDM utiliza un conjunto diferente de archivos de configuración. Están ubicados en el directorio `/etc/X11/gdm`.

  El archivo principal es el `gdm.conf`. Busca la sección **[xdmcp]**. Deberías ver una entrada 'Enable', la cual debe tener el valor '1' o 'true', dependiendo de la versión de GDM. He aquí un ejemplo:

  ```
  [xdmcp]
  Enable=true
  HonorIndirect=0
  MaxPending=4
  MaxPendingIndirect=4
  MaxSessions=16
  MaxWait=30
  ```

```
MaxWaitIndirect=30
Port=177
```

Nota el 'Enable=true'. Versiones mas viejas de GDM utilizan '0' y '1' para deshabilitar y habilitar XDMCP. Las mas nuevas en cambio usan 'false' y 'true'.

3. Si el Display Manager está definitivamente corriendo, y está escuchando las peticiones de estaciones de trabajo remotas, entonces puede ser un simple problema de mapeo de direcciones IP a nombres. La estación de trabajo necesita estar en el archivo `/etc/hosts` o bien figurar en algún DNS.

Capítulo 7. Kernels

Hay unas pocas deciciones que necesitan ser hechas sobre el kernel que va a correr en la estación de trabajo. Necesitas decidirte si quieres correr uno de los núcleos estándar disponibles para descarga, o uno propio. Y también necesitas decidir si quieres correr la pantalla gráfica, con la barra de progreso, disponible gracias al **Linux Progress Patch (LPP)**.

7.1. Kernels estándar suministrados por LTSP

El paquete de kernel que está disponible con LTSP incluye dos núcleos. Uno posee el Linux Progress Patch y el otro no.

Ambos ya tienen el parche para Swap sobre NFS aplicado.

7.2. Construye tu propio núcleo

Hay dos maneras de configurar un kernel para LTSP. El método por defecto es usar algo llamado 'Disco RAM Inicial', o **initrd**. La imagen initrd es un pequeño sistema de archivos agregado al núcleo. Este sistema de archivos es cargado en memoria, y una vez que el kernel ha iniciado, montará al disco ram como su sistema de archivos raíz. Hay un par de ventajas provenientes de utilizar una imagen initrd. Primero, podemos compilar los controladores de las placas de red como módulos y cargar el módulo correcto durante el inicio. Esto permite que un solo núcleo soporte virtualmente a todas las placas de red. La otra ventaja es que podemos correr al cliente DHCP en modo usuario, mas que en modo kernel. Correrlo en modo usuario provee un mejor control sobre las opciones requeridas y recibidas desde el servidor. También, hace al núcleo un poco mas pequeño. La otra manera de configurar el kernel es sin el initrd. Construir un kernel sin disco ram requiere que el controlador específico para la tarjeta de red esté estáticamente ligado (linkeado) dentro del núcleo, y también requiere que las opciones (TOKEN=textbf)IP−Autoconfig y (TOKEN=textbf)Root Filesystem on NFS estén activadas. La ventaja de no utilizar un initrd es que el kernel es un poco mas pequeño, por lo que iniciará mas rápido. De todos modos, una vez que la estación de trabajo esté corriendo, virtualmente no hay diferencias de rendimiento.

El núcleo estándar LTSP incluye in Disco Ram Inicial (initrd) que se hace cargo de detectar la placa de red y de realizar un requerimiento DHCP en modo usuario. El principal objetivo fue hacer la imagen lo mas pequeña posible. Por lo tanto, hemos escogido la librería libc de uClinux como reemplazo, y a BusyBox para las herramientas necesarias durante el inicio.

Si quieres construir tus propios núcleos, deberías descargar el paquete ltsp_initrd_kit. Contiene la jerarquía del sistema de archivos raíz y un script para construir la imagen.

7.2.1. Obteniendo las fuentes del kernel

Cuando se construye un núcleo personalizado, usualmente es una buena idea empezar con la fuentes frescas de **ftp.kernel.org**. La razón de esto es que algunas distribuciones, como RedHat, aplican varios parches en sus fuentes de núcleo, dejándote con un conjunto de código fuente que no concuerda con el kernel oficial.

Descarga el paquete de fuentes de kernel de tu agrado, y guárdalo en el directorio `/usr/src`. Los núcleos están ubicados en el directorio `/pub/linux/kernel` del servidor ftp de ftp.kernel.org. Necesitarás uno de la serie 2.4.x, debido a que el soporte para **devfs** es necesario.

También, si quieres incluir soporte para intercambio (swap) sobre NFS o el Linux Progress Patch (LPP), deberás asegurarte de que dichos parches y los fuentes del kernel sean de la misma versión. Al tiempo de escribir este documento, la versión 2.4.9 del kernel de Linux era la última que soportaba estas características.

Para nuestro ejemplo, usaremos el núcleo 2.4.9. La ubicación exacta es: `ftp://ftp.kernel.org/pub/linux/kernel/v2.4/linux−2.4.9.tar.bz2`.

Desempaqueta las fuentes en el directorio `/usr/src`. Necesitas ser cuidadoso, ya que cuando lo haces, habrá un directorio llamado `linux`. Si ya posees un directorio llamado `linux` correspondiente a otro conjunto de fuentes, vas a hacer lío. Por eso, verifica si ya hay un directorio con anterioridad y, de ser así, renómbralo.

El paquete de fuentes que descargamos han sido comprimidas con la utilidad **bzip2**. Por lo tanto, necesitamos descomprimirlo antes de utilizar **tar**. Puedes utilizar el siguiente comando para este propósito:

```
bunzip2 <linux−2.4.9.tar.bz2 | tar xf −
```

Cuando finalice, tendrás un directorio llamado `linux` conteniendo un completo árbol de fuentes. En este punto, yo usualmente renombro al directorio en algo más significativo.

```
mv linux linux−2.4.9
```

Una vez que el directorio ha sido renombrado, cambio a él:

```
cd linux−2.4.9
```

Usualmente me gusta modificar el `Makefile` antes de comenzar a configurar el kernel. Cerca del tope del archivo hay una variable llamada **EXTRAVERSION**, la cual lleno con 'ltsp−1' con el objeto de que el kernel se llame '2.4.9−ltsp−1', lo que hace mas fácil identificarlo luego. El tope de Makefile debería verse así:

```
VERSION = 2
PATCHLEVEL = 4
SUBLEVEL = 9
EXTRAVERSION = −ltsp−1

KERNELRELEASE=$(VERSION).$(PATCHLEVEL).$(SUBLEVEL)$(EXTRAVERSION)
```

7.2.2. Parches al núcleo

Luego de desempaquetar al kernel, puede haber varios parches que quieras aplicar. Por ejemplo, el parche para Intercambio (swap) sobre NFS o el Linux Progress Patch (LPP). Estos parches DEBEN ser aplicados antes de configurar el núcleo.

7.2.2.1. Intercambio (swap) sobre NFS.

El parche para swap sobre NFS le permitirá a la estación de trabajo utilizar un archivo de intercambio ubicado en un servidor NFS. Aunque usualmente se recomienda poseer toda la memoria necesaria en la estación de trabajo para no requerir esto, a veces puede ser dificultoso agregar mas RAM, especialmente en ordenadores viejos. Por lo tanto, la habilidad de intercambiar sobre NFS permitirá que un ordenador inusable se transforme en uno útil.

Si el directorio actual es /usr/src/linux−2.4.9, y el parche está en /usr/src, entonces puedes hacer lo siguiente para probarlo:

```
patch −p1 −−dry−run <../linux−2.4.9−nfs−swap.diff
```

Esto probará al parche, para asegurarse de que será aplicado limpiamente. Si finaliza sin errores, entonces puedes aplicarlo sin la opción **−−dry−run**.

```
patch −p1 <../linux−2.4.9−nfs−swap.diff
```

7.2.2.2. Linux Progress Patch (LPP)

El Linux Progress Patch (LPP) te permitirá configurar un logo gráfico para que se muestre al momento de iniciar la estación de trabajo. Los mensajes normales del kernel al iniciarse son redireccionados a otra pantalla tty, e instrucciones especiales son agregadas a los scripts de inicio para que la barra de progreso refleje cómo va marchando la cosa.

Al igual que el parche de swap sobre NFS, puedes probar el parche LPP mediante el siguiente comando:

```
patch −p1 −−dry−run <../lpp−2.4.9
```

Si la prueba resulta exitosamente, entonces puedes aplicar el parche con:

```
patch −p1 <../lpp−2.4.9
```

7.2.3. Configurando las opciones del núcleo

Ahora puedes correr el programa de configuración del kernel de tu elección. Las opciones disponibles son:

- make xconfig

  Invocará una versión de la utilidad de configuración para X Window.

- make menuconfig

  Invocará una utilidad de configuración simple, basada en una opción a la vez en modo texto.

- make config

  Invocará una utilidad de configuración simple, basada en una opción a la vez en modo texto.

7.2.3.1. Configuración del kernel para utilizar una imagen initrd

Configurar el kernel para utilizar una imagen initrd requiere de las siguientes opciones:

- File systems −> /dev filesystem support

  El soporte para el sistema de archivos /dev debe estar habilitado. Esto es seleccionado en la sección 'File Systems'. NO ESPECIFICAR 'Automatically mount at boot'. El montaje debe ser realizado por el script /linuxrc.

- Block devices −> RAM disk support

Las estaciones de trabajo LTSP requieren que el kernel tengan soporte para disco RAM, lo cual es seleccionado en la sección 'Block Devices'.

- Block devices −> Initial RAM disk (initrd) support

  Esto también debe estar habilitado.

- Processor type and features −> Processor family

  Necesitas asegurarte de que el kernel que vayas a construir corra en el CPU de la estación de trabajo. Esto se hace en la sección 'Processor type and features'. Deberías también desactivar el soporte SMP, a menos que tengas múltiples CPUs.

- File systems −> Network file systems −> NFS Client support

  La estación de trabajo deberá montar su sistema de archivos raíz vía NFS, por lo que el soporte para cliente NFS es requerido.

Eso debería ser suficiente. Puedes también desactivar varias opciones innecesarias, para reducir el tamaño del kernel final.

7.2.3.2. Configuración del kernel para no utilizar una imagen initrd

Configurar el kernel para no utilizar una imagen initrd difiere del que sí lo hace en lo siguiente:

- Block devices −> RAM disk support

  Las estaciones de trabajo LTSP requieren que el kernel soporte un disco RAM.

- Block devices −> Initial RAM disk (initrd) support

  Esto necesita ser desactivado.

- Networking options −> IP:kernel level autoconfiguration

  Esto necesita estar habilitado. Le indicará al kernel que debe configurar su interfase de red eth0 automáticamente, basándose en valores previos pasados por la línea de comandos del núcleo.

  No es necesario especificar las opciones DHCP, BOOTP o RARP debido a que la bootrom Etherboot ya hace una petición DHCP o BOOTP, y hará que los parámetros IP estés disponibles en la línea de comandos del kernel. Esto evita que se realice una segunda petición.

- Network device support −> Ethernet (10 or 100Mbit)

  Cuando no utilizas un initrd, debes elegir una placa de red específica que concuerde con la tuya. Esto DEBE estar ligado (linkeado) estáticamente dentro del núcleo, debido a que la interfase ethernet es necesaria antes de que se monte el sistema de archivos raíz. Esta es la mayor diferencia entre la forma en que se trabaja con y sin initrd.

- File systems −> /dev filesystem support

Como en LTSP versión 2.09pre2, **devfs** es necesario. Esto es así, se use initrd o no.

- File systems −> Automatically mount at boot

  Cuando NO se utiliza initrd, el sistema de archivos /dev debe ser montado por el kernel, durante el inicio. Por lo tanto, tienes que decir 'Y' aquí.

- File systems −> Network file systems −> NFS Client support

  La estación de trabajo montará su sistema de archivos raíz via NFS, por lo que el soporte para cliente NFS es requerido.

7.2.4. Construyendo el kernel

Para hacer las cosas mas fáciles, una copia del archivo `.config` está incluida en el paquete ltsp_initrd_kit. Puedes copiarlo al directorio `/usr/src/linux−2.4.9`.

Una vez que has finalizado seleccionando y deseleccionando las opciones del núcleo, necesitas construirlo. Los siguientes comando necesitan ser ejecutados en orden de lograrlo:

```
make dep
make clean
make bzImage
make modules
make modules_install
```

Puedes juntar todo así:

```
make dep && make clean && make bzImage && make modules && make modules_install
```

El doble ampersand (&) significa que si el primer comando finaliza exitosamente, entonces el segundo será ejecutado. Si el segundo lo hace también exitosamente, se ejecutará el tercero. Y así.

Cuando la compilación ha finalizado, el nuevo kernel estará en `/usr/src/linux−2.4.9/arch/i386/boot/bzImage`.

7.2.5. Etiquetando al kernel para Etherboot

El kernel Linux debe ser preparado para que Etherboot pueda manejarlo. Esto es llamado 'etiquetar' el kernel. Este proceso le agregará algún código adicional que es ejecutado antes de que el control sea pasado al núcleo. La herramienta para etiquetar el kernel se llama '**mknbi−linux**'.

El ltsp_initrd_kit incluye un script de shell llamado **buildk** que incluye todos los comandos que necesites para preparar la imagen del núcleo para que inicie por red.

Capítulo 8. Entradas lts.conf

Cuando diseñamos LTSP, una de las cuestiones que sabíamos era que podíamos tener que lidiar con hardware variado para las estaciones de trabajo. Ciertamente, cualquier combinación disponible hoy de CPU, placa de red y de video, podría no estar disponible en 3 meses. cuando querramos agregar mas estaciones de trabajo a la red.

Por eso, hemos desarrollado una manera de especificar la configuración de cada terminal. El archivo de configuración se llama `lts.conf` y reside en el directorio `/opt/ltsp/i386/etc`.

El formato del archivo lts.conf permite opciones por defecto y opciones específicas para cada estación de trabajo. Si todas tus terminales son idénticas, entonces deberías especificar todas las opciones bajo la sección [Default].

8.1. Archivo lts.conf de ejemplo

Aquí hay un ejemplo de un archivo lts.conf:

```
[Default]
        SERVER             = 192.168.0.254
        X_MOUSE_PROTOCOL   = "PS/2"
        X_MOUSE_DEVICE     = "/dev/psaux"
        X_MOUSE_RESOLUTION = 400
        X_MOUSE_BUTTONS    = 3
        USE_XFS            = N
        RUNLEVEL           = 5

[ws001]
        XSERVER            = auto
        X_MOUSE_PROTOCOL   = "Microsoft"
        X_MOUSE_DEVICE     = "/dev/ttyS1"
        X_MOUSE_RESOLUTION = 50
        X_MOUSE_BUTTONS    = 3
        X_MOUSE_BAUD       = 1200

[ws002]
        XSERVER            = XF86_Mach64

[ws003]
        RUNLEVEL           = 3
```

8.2. Opciones de lts.conf disponibles

8.2.1. Opciones generales

Comentarios
: Los comentarios comienzan con numeral ('#') y continúan hasta el final de la línea.

LTSP_BASEDIR
: Esto indica dónde el sistema de archivos raíz de LTSP está ubicado. Por defecto es en `/opt/ltsp/`.

SERVER

Este es el servidor utilizado por XDM_SERVER, TELNET_HOST, XFS_SERVER y SYSLOG_HOST, si alguno de los mencionados no está especificado explícitamente. Si tienes una máquina que está actuando como el servidor de todo, entonces puedes especificar la dirección IP solamente aquí y omitir el resto. Si ningún valor es elegido, se toma por defecto **192.168.0.254**.

SYSLOG_HOST

Si quieres enviar los mensajes de sistema a una máquina distinta al servidor por defecto, entonces puedes especificar dicha máquina aquí. Si ningún parámetro es especificado se utilizará la entrada de SERVER.

NFS_SERVER

Especifica la dirección IP del servidor desde donde el sistema de archivos `/home` es montado. Por defecto se utiliza la entrada **SERVER**.

USE_NFS_SWAP

Pon **Y** aquí si quieres habilitar intercambio sobre NFS. Por defecto es **N**.

SWAPFILE_SIZE

Aquí es donde puedes controlar el tamaño del archivo de intercambio. El tamaño por defecto es **64m**.

SWAP_SERVER

El archivo de intercambio puede existir en cualquier servidor de la red. Puedes especificar la dirección IP de dicho servidor. Por defecto se toma el valor de NFS_SERVER.

NFS_SWAPDIR

Para especificar el directorio en el server que es exportado via NFS. Por defecto es `/var/opt/ltsp/swapfiles`. Asegúrate de que dicho directorio esté en `/etc/exports`.

TELNET_HOST

Si la estación de trabajo es configurada para tener una interface basada en caracteres, entonces el valor de este parámetro será usado como el host donde se haga la sesión de telnet. Si ningún valor es especificado se utilizará la entrada de **SERVER**.

DNS_SERVER

Usada para crear el archivo resolv.conf.

SEARCH_DOMAIN

Usada para crear el archivo resolv.conf.

MODULE_01** thru **MODULE_10

Hasta 10 módulos de kernel pueden ser cargados mediante estas entradas. Puedes especificar la línea de comando que usarías cuando usas indmod. Por ejemplo:

```
MODULE_01   = uart401.o
MODULE_02   = sb.o io=0x220 irq=5 dma=1
MODULE_03   = opl3.o
```

Si el valor de este parámetro es un camino absoluto, entonces el comando **insmod** será utilizado. De otro modo, se utilizará **modprobe**.

RAMDISK_SIZE

Cuando la estación de trabajo se inicia, crea un disco RAM y monta en él el directorio /tmp. Puedes controlar el tamaño de este sistema de archivos con este parámetro. Debes especificar las unidades en kilobytes (1024 bytes). Por ejemplo para crear un disco RAM de 2 MB debes especificar **RAMDISK_SIZE=2048**.

Si cambias el tamaño del disco RAM aquí, también debes hacerlo dentro del kernel. Esto lo puedes hacer compilándolo o bien, si estás usando Etherboot o Netboot, poniéndolo como argumento cuando etiquetas el kernel con mknbi−linux.

El valor por defecto es 1024 (1 MB).

RCFILE_01** thru **RCFILE_10
Scripts RC adicionales pueden ser ejecutados por rc.local. Tan sólo coloca tu script en el directorio `etc/rc.d` y especifica el nombre en alguna de estas entradas.

SOUND
Si el paquete LTSP Sound está instalado, entonces necesitas poner **Y** en esta entrada y ejecutar el script **rc.sound** para configurar la placa de sonido y el demonio. Por defecto es **N**.

RUNLEVEL
El runlevel determina el modo en el cual la estación de trabajo estará corriendo. Los siguientes runlevels son los soportados:

3 : Iniciará un shell. Útil para depuración de la estación de trabajo.

4 : Esto correrá una o más sesiones de telnet en el TELNET_HOST. Esto es genial para reemplazar viejas terminales seriales.

5 : Modo GUI. Arrancará X Window, y enviará una petición XDMCP al servidor, quien devolverá una pantalla de login a la terminal. Se necesita tener un Display Manager corriendo en el servidor, como XDM, GDM o KDM.

TELNET_SESSIONS
Indica cuántas sesiones de telnet puedes correr. Cada sesión estará en una diferente pantalla virtual, a las que se puede acceder con las teclas ALT−F1 a ALT−F9. El valor por defecto es 2.

8.2.2. Opciones de X Window

XDM_SERVER
Si quieres apuntar XDM hacia una máquina que no sea tu servidor por defecto, entonces debes especificarlo aquí. Si este parámetro NO es especificado, entonces se usará la entrada en SERVER.

XSERVER
Esta entrada define cuál servidor X la estación de trabajo correrá. Para placas PCI y AGP, este parámetro no debería ser necesario. El script rc.local debería poder autodetectar la tarjeta. Puedes también poner **auto** aquí a fin de indicar que se tratará de autodetectar el video.

Para placas ISA, o para especificar un servidor X en particular, puedes poner el nombre del driver o servidor X aquí.

Si el valor comienza con 'XF86_', entonces XFree 3.3.6 será utilizado. De otro modo, XFree 4.1.x lo será. El valor por defecto es **auto**.

X_MODE_0** through **X_MODE_2

Hasta 3 Modelines o resoluciones pueden ser configuradas para la terminal. Esta entrada puede tomar dos tipos diferentes de valores. Puede ser tanto una resolución como un modeline completo:

```
X_MODE_0 = 800x600

   o bien

X_MODE_0 = 800x600 60.75 800 864 928 1088 600 616 621 657 −HSync −VSync
```

Si ninguna de las entradas X_MODE_x es especificada, entonces se utilizarán los modelines por defecto, y las resoluciones serán de 1024x768, 800x600 y 640x480.

Si una o mas entradas X_MODE_x es especificada, entonces no se tendrán en consideración los modelines por defecto.

X_MOUSE_PROTOCOL

Cualquier valor que funcione con el Protocolo de Puntero XFree86 puede ser puesto aquí. Los valores típicos incluyen 'Microsoft' y 'PS/2'. El valor por defecto es '**PS/2**'.

X_MOUSE_DEVICE

Este es el dispositivo al cual el mouse está conectado. Si es un mouse serial, debería ser un puerto serie, como **/dev/ttyS0** o **/dev/ttyS1**. Si es un mouse PS/2, este valor debería ser **/dev/psaux**. El valor por defecto es **/dev/psaux**.

X_MOUSE_RESOLUTION

Este es el valor 'Resolution' que encuentras en cualquier archivo **XF86Config**. Un valor típico para un mouse serial es **50** y para uno PS/2 es **400**. El valor por defecto es **400**.

X_BUTTONS

Le dice la sistema cuántos botones tiene el mouse. Usualmente son **2** o **3**. El valor por defecto es **3**.

X_MOUSE_EMULATE3BTN

Esto lo dice al servidor X que debe emular el tercer botón del mouse cuando se presionen simultáneamente los botones izquierdo y derecho. El valor por defecto es **N**.

X_MOUSE_BAUD

Para ratones seriales, define la tasa de baudios. El valor por defecto es **1200**.

X_COLOR_DEPTH

Este es el número de bits a utilizar para la profundidad del color. Los valores posibles son **8**, **15**, **16**, **24** y **32**. 8 bits te dará 256 colores, 16 te dará 65536, 24 te dará 16 millones y 32 te dará 4,2 billones de colores. No todos los servidores X soportan estos valores. El valor por defecto es **16**

USE_XFS

Tienes la opción de correr un Servidor de Fuentes X (XFS, X Font Server) o bien leer las fuentes vía NFS. El servidor de fuentes debería proveer un camino simple para mantener todas las fuentes en un solo lugar, pero hay algunos problemas cuando el número de estaciones de trabajo supera las 40. Los

2 valores en esta opción son **Y** o **N**. El valor por defecto es **N**. Si sí quieres usar un Servidor de Fuentes, entonces debes usar la entrada **XFS_SERVER** para especificar su dirección IP.

XFS_SERVER
Si estás usando un Servidor de Fuentes, debes especificar su dirección IP aquí. Si no está especificado, entonces se usará por defecto la entrada especificada en **SERVER**.

X_HORZSYNC
Setea la sincronización horizontal del monitor. El valor por defecto es **"31−62"**.

X_VERTREFRESH
Setea el refresco vertical del monitor. Por defecto es **"55−90"**.

XF86CONFIG_FILE
Si quieres crear tu propio archivo de configuración XF86Config, lo puedes hacer. Sólo tienes que colocarlo en el directorio **/opt/ltsp/i386/etc**. Luego, cualquiera sea el nombre que le pongas, debes entrar dicho nombre en este parámetro. Por ejemplo:

```
XF86CONFIG_FILE = XF86Config.ws004
```

8.2.3. Opciones para pantallas táctiles

USE_TOUCH
Si estás conectando una pantalla táctil a la estación de trabajo, puedes habilitarla mediante esta entrada, colocando **Y**. Una vez hecho esto, deberás configurar algunos aspectos más detallados a continuación. El valor por defecto es **N**.

X_TOUCH_DEVICE
Una pantalla táctil funciona como un mouse y usualmente se conecta a un ordenador mediante el puerto serie. Puedes especificar cuál puerto serie en esta entrada. Por ejemplo, podrías poner algo tipo **/dev/ttyS0**. No hay valor por defecto aquí.

X_TOUCH_MINX
Valor de calibración para una pantalla táctil Elo Touch. El valor por defecto es **433**.

X_TOUCH_MAXX
Valor de calibración para una pantalla táctil Elo Touch. El valor por defecto es **3588**.

X_TOUCH_MINY
Valor de calibración para una pantalla táctil Elo Touch. El valor por defecto es **569**.

X_TOUCH_MAXY
Valor de calibración para una pantalla táctil Elo Touch. El valor por defecto es **3526**.

X_TOUCH_UNDELAY
Valor de calibración para una pantalla táctil Elo Touch. El valor por defecto es **10**.

X_TOUCH_RPTDELAY
Valor de calibración para una pantalla táctil Elo Touch. El valor por defecto es **10**.

8.2.4. Opciones para Aplicaciones Locales

LOCAL_APPS
Si quieres usar la opción de correr las aplicaciones localmente en la estación de trabajo, entonces debes poner **Y** aquí. Varios pasos adicionales deben llevarse a cabo para que las aplicaciones locales funcionen. Mira la sección 'Aplicaciones Locales' para mayor información. El valor por defecto es **N**.

NIS_DOMAIN
Si habilitaste Aplicaciones Locales, necesitarás un servidor NIS en la red. La entrada NIS_DOMAIN indica dónde está dicho servidor. Necesita concordar con el nombre de dominio NIS configurado en el servidor NIS. Esto NO es lo mismo que un dominio de Internet. El valor por defecto es **ltsp**.

NIS_SERVER
Coloca aquí la dirección IP del servidor NIS si es que no quieres que se envíe una petición de broadcast preguntando por el mismo.

8.2.5. Opciones de teclado

Todos los archivos de soporte de teclado ahora son copiados dentro de la jerarquía /opt/ltsp/i386, por lo que configurar un teclado internacional ahora es cuestión de configurar XFree86. Hay varios parámetros que hacen esto posible.

Los valores descritos aquí provienen de la documentación de XFree86. Cualquier valor válido para XFree86 es válido aquí.

Nos gustaría agregar documentación que muestre qué valores son necesarios para cada tipo de teclado internacional. Si trabajas con esto y puedes configurar tu teclado internacional, entonces escríbele al grupo de LTSP y te estaremos agradecidos.

XkbTypes
El valor por defecto aquí es la palabra '**default**'.

XkbCompat
El valor por defecto aquí es la palabra '**default**'.

XkbSymbols
El valor por defecto aquí es la palabra '**default**'.

XkbModel
El valor por defecto es '**pc101**'.

XkbLayout
El valor por defecto es '**us**'.

8.2.6. Opciones para la configuración de la impresora

Hasta tres impresoras pueden ser conectadas a una estación de trabajo sin disco. Una combinación de impresoras paralelas o seriales puede ser configurada mediante las siguientes entradas en el archivo **lts.conf**

PRINTER_0_DEVICE
El nombre del puerto de la primera impresora. Nombres como **/dev/lp0**, **/dev/ttyS0**, o **/dev/ttyS1** están permitidos.

PRINTER_0_TYPE
El tipo de impresora conectada. Los valores posibles son **P** para paralela o **S** para serial.

PRINTER_0_PORT
El puerto TCP/IP a usar. Por defecto, es '**9100**'.

PRINTER_0_SPEED
Si la impresora es serial, este esta es la variable que seleccionará la tasa de baudios. Por defecto se usará '**9600**'.

PRINTER_0_FLOWCTRL
Para impresoras seriales, el control de flujo puede ser especificado. Se puede poner tanto **S** para contrlar por Software (XON/XOFF) como **H** para hacerlo por hardware (CTS/RTS). Si ningún valor es especificado, se usará '**S**'.

PRINTER_0_PARITY
Para impresoras seriales, la Paridad puede ser especificada. Las opciones son: **E−Even**, **O− Odd** o **N−None**. El valor por defecto es '**N**'.

PRINTER_0_DATABITS
Para impresoras seriales, el número de data bits puede ser especificado. Las opciones son **5**, **6**, **7** y **8**. El valor por defecto es '**8**'.

PRINTER_1_DEVICE
Puerto de la segunda impresora

PRINTER_1_TYPE
Tipo de la segunda impresora

PRINTER_1_PORT
Tipo de la segunda impresora

PRINTER_1_SPEED
Tasa de baudios de la segunda impresora (serial)

PRINTER_1_FLOWCTRL
Control de flujo de la segunda impresora (serial)

PRINTER_1_PARITY
Paridad de la segunda impresora (serial)

PRINTER_1_DATABITS
Data bits de la segunda impresora (serial)

PRINTER_2_DEVICE
Puerto de la tercera impresora

PRINTER_2_TYPE
Tipo de la tercera impresora

PRINTER_2_PORT
Puerto TCP/IP de la tercera impresora

PRINTER_2_SPEED
Tasa de baudios de la tercera impresora (serial)

PRINTER_2_FLOWCTRL
Control de flujo de la tercera impresora (serial)

PRINTER_2_PARITY
Paridad de la tercera impresora (serial)

PRINTER_2_DATABITS
Data bits de la tercera impresora (serial)

Capítulo 9. Aplicaciones Locales

En un ambiente LTSP, tienes la opción de correr las aplicaciones localmente, en las estaciones de trabajo, o remotamente en el servidor.

Por lejos, la manera mas fácil de configurar un ambiente LTSP es corriendo las aplicaciones en el servidor. Esto es, la aplicación del cliente corre en el servidor, usando su CPU y memoria, mientras muestra su salida por el monitor de la estación de trabajo.

Esta es una capacidad fundamental de X Window. La estación de trabajo funciona como una terminal X.

En orden de que un usuario pueda correr una aplicación en su estación de trabajo, esta última necesita saber alguna información acerca de este usuario, como la siguiente:

- Id del usuario
- Grupo primario al cual pertenece
- Directorio Home del mismo

LTSP confía en el Network Information Service − NIS (antes llamado ***Páginas Amarillas***) para hacer que el usuario y el grupo esté disponible para las estaciones de trabajo.

9.1. Beneficios de correr las aplicaciones localmente

Hay algunos beneficios al correr las aplicaciones en la estación de trabajo.

- Reduce la carga total en el servidor. En redes largas con aplicaciones de uso intensivo de memoria como Netscape, correr las aplicaciones en las estaciones de trabajo puede proveer un mejor rendimiento, siempre y cuando la estación de trabajo sea capaz de manejar estas aplicaciones.
- Las aplicaciones que se cierren solas (runaway) por errores no afectarán a los otros usuarios.
- El soporte de sonido es mucho mas fácil de configurar cuando la aplicación que toca el sonido está corriendo en la estación de trabajo.

9.2. Asuntos a considerar con el soporte para aplicaciones locales

Configurar la habilidad para aplicaciones locales requiere de mucho más:

- Mas demanda a la estación de trabajo. Se requiere mas RAM y un CPU mucho mas poderoso. 64 MB en la estación es un buen punto de partida.
- NIS − Para correr las aplicaciones en la estación de trabajo, primero debes ingresar en ella. Esto requiere de alguna forma de autenticación de contraseña. NIS ha sido elegido como el método para autenticar a los usuarios a través de la red. Este documento explicará brevemente cómo configurar NIS en el servidor.
- Directorios adicionales necesitan ser exportados vía NFS.
- Inicio de aplicaciones más lento debido a que necesitan ser leídas vía NFS, causando un incremento de la actividad de la red. También, a raíz de que cada copia del programa está corriendo en cada CPU, no obtendrás la ventaja de Linux/Unix de compartir segmentos de código entre múltiples instancias del mismo programa, lo que reduciría el tiempo que toma iniciar un programa por sucesivas veces.

9.3. Configuración del servidor para Aplicaciones Locales

9.3.1. Entradas en lts.conf

Una pequeña cantidad de entradas debe ser configurada en `lts.conf`:

LOCAL_APPS
Esta opción debe ser **Y**. Esto causará lo siguiente:

1. El directorio `/home` del servidor será montado vía NFS.
2. El archivo `/var/yp/nicknames` será creado en la estación de trabajo.
3. El **portmapper** será iniciado en la terminal.
4. **xinetd** también será iniciado.
5. El archivo `/etc/yp.conf` será creado.
6. El comando **domainname** correrá con el valor de la entrada **NIS_DOMAIN** del archivo lts.conf.
7. El **ypbind** correrá en la estación de trabajo.

NIS_DOMAIN
Con NIS, todos los nodos de la red que quieran asociarse con un servidor NIS específico deben pertenecep al mismo dominio NIS (que no está en modo alguno relacionado con el dominio DNS). Debes usar esta entrada para especificar el nombre del dominio NIS al que la estación de trabajo pertenece.

NIS_SERVER
NIS tratará de bindearse (emparejarse) con un servidor NIS específico, o tirará una petición de broadcast a través de la red. Si quieres especificar un servidor, entonces pon la dirección IP del mismo en esta entrada.

9.3.2. Network Information Service − NIS

Nis es un servicio del tipo cliente/servidor. En el servidor, hay un demonio corriendo, que aceptará peticiones de los clientes (estaciones de trabajo). El demonio en el servidor es llamado **ypserv**.

En la estación de trabajo, hay un proceso llamado **ypbind**. Cuando la estación de trabajo necesita información acerca de un usuario, como verificar un password o encontrar su directorio principal, utilizará a ypbind para establecer una conexión con ypserv, en el servidor.

Si ya estás corriendo NIS en tu red, entonces no hay necesidad de configurar al servidor LTSP para que corra ypserv. Simplemente puedes ajustas las entradas NIS_DOMAINNAME y NIS_SERVER en lts.conf.

Si NO estás corriendo NIS en tu red, entonces necesitarás configurar el servidor para poder correr **ypserv**.

Para una completa información acerca de cómo configurar un servidor NIS, hay un HOWTO en el Linux Documentation Project llamado ***The Linux NIS(YP)/NYS/NIS+ HOWTO***. Mira la lista de recursos al final de este documento.

9.4. Configuración de la Aplicación

Para configurar una aplicación para que corra en la estación de trabajo, necesitas colocar todos los componentes de la misma en un lugar donde la terminal la vea.

Con versiones mas viejas de LTSP (2.08 y anteriores), montones de directorios eran exportados en el servidor y montados por la estación de trabajo. Directorios como `/bin`, `/usr/bin`, `/lib` y `/usr` eran expuestos a las terminales.

El problema con ese esquema esta que sólo funcionaba si la estación de trabajo y el servidor tenían la misma arquitectura. De hecho, aún pequeñas diferencias, como un servidor Pentium II (i686) y una estación de trabajo Pentium Classic (i586) podía ser un problema, debido a que el servidor generalmente tenía las librerías para i686 que no las de i386, i486 e i586.

Por eso, la manera mas limpia de manejar esto es teniendo un completo árbol con todos los binarios y las librerías que la estación de trabajo necesita, independientemente del servidor.

Configurar una aplicación para ejecución local requiere poner todas las piezas requeridas en un árbol. Uno de los paquetes disponibles para descarga en el sitio LTSP es el paquete Local Netscape, el cual instala montones de archivos en el directorio `/opt/ltsp/i386/usr/local/netscape`. Cosas como las clases java, los archivos de ayuda, los binarios y los scripts son puestos aquí.

Netscape no necesita ninguna librería del sistema adicional, por lo que nada debe ser agregado al directorio `/opt/ltsp/i386/lib`. Muchas aplicaciones sí requieren de librerías adicionales.

Por eso, ¿cómo puedes determinar lo que necesita tu aplicación? Con el comando **ldd**.

Asumamos que quieres hacer que una determinada aplicación corra localmente. Hemos escogido **gaim** como ejemplo. **gaim** es un cliente para AOL Instant Messenger, que te permite comunicarte con otra gente en foros AOL.

La primero que necesitas hacer es encontrar el binario **gaim**. En RedHat 7.2, está ubicado en el directorio `/usr/bin`.

Una vez que lo has localizado, puedes correr **ldd** con él:

```
[jam@server /]$ ldd /usr/bin/gaim
        libaudiofile.so.0    => /usr/lib/libaudiofile.so.0 (0x40033000)
        libm.so.6            => /lib/i686/libm.so.6 (0x40051000)
        libnsl.so.1          => /lib/libnsl.so.1 (0x40074000)
        libgnomeui.so.32     => /usr/lib/libgnomeui.so.32 (0x4008a000)
        libart_lgpl.so.2     => /usr/lib/libart_lgpl.so.2 (0x4015d000)
        libgdk_imlib.so.1    => /usr/lib/libgdk_imlib.so.1 (0x4016c000)
        libSM.so.6           => /usr/X11R6/lib/libSM.so.6 (0x40191000)
        libICE.so.6          => /usr/X11R6/lib/libICE.so.6 (0x4019a000)
        libgtk-1.2.so.0      => /usr/lib/libgtk-1.2.so.0 (0x401b1000)
        libdl.so.2           => /lib/libdl.so.2 (0x402df000)
        libgdk-1.2.so.0      => /usr/lib/libgdk-1.2.so.0 (0x402e3000)
        libgmodule-1.2.so.0  => /usr/lib/libgmodule-1.2.so.0 (0x40319000)
        libXi.so.6           => /usr/X11R6/lib/libXi.so.6 (0x4031d000)
        libXext.so.6         => /usr/X11R6/lib/libXext.so.6 (0x40325000)
        libX11.so.6          => /usr/X11R6/lib/libX11.so.6 (0x40333000)
        libgnome.so.32       => /usr/lib/libgnome.so.32 (0x40411000)
```

```
        libgnomesupport.so.0 => /usr/lib/libgnomesupport.so.0 (0x40429000)
        libesd.so.0          => /usr/lib/libesd.so.0 (0x4042e000)
        libdb.so.2           => /usr/lib/libdb.so.2 (0x40436000)
        libglib-1.2.so.0     => /usr/lib/libglib-1.2.so.0 (0x40444000)
        libcrypt.so.1        => /lib/libcrypt.so.1 (0x40468000)
        libc.so.6            => /lib/i686/libc.so.6 (0x40495000)
        libz.so.1            => /usr/lib/libz.so.1 (0x405d1000)
        /lib/ld-linux.so.2   => /lib/ld-linux.so.2 (0x40000000)
```

El listado de arriba muestra todas las librerías que el programa **gaim** utiliza dinámicamente.

La mayoría de los programas que usan librerías compartidas, confían en el cargador dinámico **ld−linux** para localizarlas y cargarlas. Algunos programas, en cambio, cargan sus librerías manualmente con la función **dlopen()**. Para esas aplicaciones, **ldd** no mostrará las librerías. En ese caso, **strace** puede ser utilizado para rastrear la ejecución del programa, para que puedas ver las llamadas **dlopen()** con el nombre de la librería como argumento.

Una vez que las librerías han sido recogidas, deben ser copiadas en los lugares correctos dentro del árbol `/opt/ltsp/i386`.

9.5. Lanzando aplicaciones locales

En X Window, los programas típicamente correr relativos a donde el administrador de ventanas corre. Esto significa que, si el administrador de ventanas corre en el servidor, mostrando su salida por la estación de trabajo, entonces cualquier programa que es lanzado también correrá en el servidor, enviando su salida por la terminal.

El truco es hacer que el servidor le diga a la estación de trabajo que lance el programa. Esto es realizado comúnmente por el comando **rsh**.

Aquí hay un ejemplo de cómo correr **gaim** en la terminal:

```
HOST=`echo $DISPLAY | awk -F: '{ print $1 }'`
rsh ${HOST} /usr/bin/gaim -display ${DISPLAY}
```

El ejemplo de arriba puede ser ejecutado en una ventana de **xterm**, o puede ser puesto en un script de shell, y ser lanzado por un icono desde el escritorio.

Lanzar el Netscape Local es similar, pero una variable de entorno adicional nececita configurarse antes de correr el programa.

```
HOST=`echo $DISPLAY | awk -F: '{ print $1 }'`
rsh ${HOST} MOZILLA_HOME=/usr/local/netscape \
       /usr/local/netscape/netscape -display ${DISPLAY}
```

Capítulo 10. Ejemplos de configuración

Casi cualquier aspecto de una estación de trabajo puede ser configurado con entradas en `lts.conf`, el cual está usualmente ubicado en el directorio `/opt/ltsp/i386/etc`.

10.1. Mouse serial (ratón de puerto serie)

He aquí un ejemplo de las entradas en `lts.conf` para un ratón serial estándar de 2 botones:

```
X_MOUSE_PROTOCOL    = "Microsoft"
X_MOUSE_DEVICE      = "/dev/ttyS0"
X_MOUSE_RESOLUTION  = 400
X_MOUSE_BUTTONS     = 2
X_MOUSE_EMULATE3BTN = Y
```

10.2. Mouse con rueda PS/2

He aquí un ejemplo de las entradas en `lts.conf` para un Intellimouse:

```
X_MOUSE_PROTOCOL    = "IMPS/2"
X_MOUSE_DEVICE      = "/dev/psaux"
X_MOUSE_RESOLUTION  = 400
X_MOUSE_BUTTONS     = 5
X_ZAxisMapping      = "4 5"
```

10.3. Impresona USB en una ThinkNIC

La estación de trabajo ThinkNIC tiene un puerto USB, que puede ser utilizado para conectar una impresora local. He aquí un ejemplo de las entradas requeridas en el archivo `lts.conf`:

```
MODULE_01           = usb-ohci
MODULE_02           = printer
PRINTER_0_DEVICE    = /dev/usb/lp0
PRINTER_0_TYPE      = S
```

10.4. Forzando una estación de trabajo a cargar un servidor XFree86 3.3.6

Por defecto, XFree 4.1.0 será utilizado en una estación de trabajo. Si por el contrario quieres forzar a una terminal a utilizar un servidor 3.3.6 mas viejo, primero necesitarás instalar el paquete XFree86 3.3.6 apropiado. Luego, necesitarás agregar la entrada en el archivo `lts.conf`. Este es un ejemplo para especificar el servidor X **SVGA**:

```
XSERVER             = XF86_SVGA
```

Capítulo 11. Otras fuentes de información

11.1. Referencias en línea

1. The LTSP home page

   www.LTSP.org

2. Diskless−Nodes HOW−TO document for Linux

   www.linuxdoc.org/HOWTO/Diskless−HOWTO.html

3. Etherboot Home Page

   etherboot.sourceforge.net

4. The Rom−O−Matic site

   www.Rom−O−Matic.net

5. XFree86−Video−Timings−HOWTO

   www.linuxdoc.org/HOWTO/XFree86−Video−Timings−HOWTO.html

6. The Linux NIS(YP)/NYS/NIS+ HOWTO

   www.linuxdoc.org/HOWTO/NIS−HOWTO.html

11.2. Publicaciones Impresas

1. Managing NFS and NIS
   Hal Stern
   O'Reilly & Associates, Inc.
   1991
   ISBN 0−937175−75−7

2. TCP/IP Illustrated, Volume 1
   W. Richard Stevens
   Addison−Wesley
   1994
   ISBN 0−201−63346−9

3. X Window System Administrator's Guide
   Linda Mui and Eric Pearce

O'Reilly & Associates, Inc.
1993
ISBN 0−937175−83−8
(Volume 8 of the The Definitive Guides to the X Window System)