
**Título: Servidor de Terminales Ligeras**

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

**Autor (es):** Ernesto Díaz Vázquez

Alberto Antonio Ferral Sainz

**Tutor:** Ing. Joaquín Quintas Santiago

**Junio 2008**

*"Los postulados del Software Libre son anticapitalistas; el Software Libre es una plataforma tecnológica Socialista, lo que sucede es que no lo vemos"*

*Roso Grimau*

# Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días, del mes de _____ del año _____.

Autores:

_____ _____

Alberto Antonio Ferral Sainz Ernesto Díaz Vázquez

Tutor:

_____

Ing. Joaquín Quintas Santiago

## Agradecimientos

A la Revolución y Fidel Castro por hacer realidad el sueño de esta universidad.

A mis familiares y en especial a mis padres por el apoyo incondicional que me han brindado durante todos estos años.

A la Universidad de las Ciencias Informáticas por la preparación que me ha dado.

A mis amigos por estar siempre presente en los buenos y malos momentos.

A la Dra. Silvia Vázquez por su importante colaboración y por estar siempre que necesitamos su ayuda.

A el micha por su incondicional ayuda.

A todas aquellas personas que de una u otra forma hicieron posible el desarrollo de este trabajo.

Alberto

A la Revolución por darme la oportunidad de estudiar en esta Universidad.

A mis hermanos por creer en mí y seguir mi ejemplo aunque no siempre ha sido el mejor.

A mis abuelos, mis tíos y mis primos por preocuparse en todo momento de mi carrera.

A mi madre y a mi padre sin su ayuda nunca hubiera podido llegar hasta aquí.

A mi novia por aguantarme más de lo que debiera.

A el proyecto Nova y Nova-FAR por la ayuda recibida durante estos años, en especial a Yoandy, Anielkis, Yunior, Laynoll, Mijaíl, Vladimir, Noel y Ángel porque los momentos que pasamos al principio fueron sin duda los mejores.

A Vega y Michel por ayudarme a encontrar un mejor camino en este año.

A todos los que pusieron su granito de arena en este trabajo.

Y por último, no por eso menos importante, a mis mejores amigos Yandry, Marvin, Ernesto y Maurys por estar siempre conmigo "*no matter what*".

Ernesto

**Dedicatoria**

Dedico este trabajo a:

Mis padres: Sin ellos este sueño no se hubiera hecho realidad.

Mis amigos: Alioscha, Henry, Heiner, Noli, Yaimara, Yoe, el Micha, Kiki, Vega, El Michel.

Mi familia: mi hermanita Angélica, mis tíos, mis primos, en especial a mi tía querida Ide por soportarme estos 5 años, a mis abuelas.

Mis compañeros de proyecto: Abel, Marvin, Neto, Angel, Migue, Miranda, Félix.

Todos los que confiaron y creyeron en mí y en que se haría realidad este sueño.

Alberto

Este trabajo se lo dedico a mi madre que no pudo estar conmigo en otros momentos importantes de mi vida como estudiante. Para ti “chamaca” por ser el mejor ejemplo que una madre puede dar a sus hijos.

Ernesto

**Resumen**

En la actualidad la industria informática avanza rápidamente y muchos usuarios, en su mayoría *gamers*, se ven obligados a deshacerse de sus equipos ante la sed de recursos del software utilizado. Así los ordenadores se han convertido en máquinas de usar y tirar y podemos encontrar en la basura equipos que podrían ser perfectamente utilizados como ordenadores de escritorio. En algunas de estas máquinas basta con instalar GNU/Linux para conseguir un sistema de escritorio, pero otras ya sea por carecer de disco duro o por ser auténticas reliquias (Pentium I o Pentium II) requieren de ayuda para volver a la vida. Esa ayuda es la que se implementará en esta solución para la distribución NOVA. En este trabajo se describe una solución para terminales ligeras con el objetivo de reutilizar computadoras que no tengan discos duros, en entornos heterogéneos. Esto significa un entorno con diferentes tipos de hardware, y con diferentes configuraciones. Además se plantea como una forma de ahorro para las empresas, con respecto a mantenimiento y personal requerido para esto, pues se concentraría toda la información en un servidor central, y sería más fácil la contención de virus, así como su mantenimiento en general. La implementación de las terminales ligeras posibilita crear condiciones más adecuadas para los usuarios, contribuye enormemente al ahorro de energía, algo muy necesario en estos días en nuestro país, y significa además un enorme ahorro de recursos, tanto monetarios como humanos.

**Palabras clave:**

- Sed de recursos
- Ahorro de energía
- Entornos heterogéneos.

## Índice

Agradecimientos ..... I

Dedicatoria ..... II

Resumen ..... III

Introducción ..... 1

**CAPÍTULO I: Fundamentación Teórica de un Servidor para Terminales Ligeras ..... 6**

1.1 Introducción ..... 6

1.2 Necesidad de crear un servidor para terminales ligeras en la distribución Nova ..... 7

1.3 ¿Qué es Software Libre? ..... 8

1.4 GNU Nova ..... 10

1.5 Gentoo: Distribución base de Nova ..... 10

1.5.1 Ventajas de Gentoo ..... 11

1.5.2 Desventajas de Gentoo ..... 12

1.5.3 Características específicas ..... 13

1.6 Principales Sistemas para Terminales Ligeras libres ..... 13

1.6.1 LTSP (Linux Terminal Server Project) ..... 14

1.6.2 PXES (Universal Linux Thin Client) ..... 16

1.6.3 2X ThinClientServer ..... 18

1.6.4 Diet-PC (Diskless Embedded Technology Personal Computer) ..... 19

1.6.5 Netstation ..... 20

1.6.6 Thinstation ..... 21

1.7 Principales Sistemas para Terminales Ligeras bajo licencia ..... 21

1.7.1 eLux NG ..... 22

1.7.2 Citrix Metaframe ..... 22

1.7.3 Terminal Services ..... 22

1.7.4 Neoware ..... 23
1.7.5 Wyse ..... 23
1.7.6 WinConnect Server XP ..... 23
1.8 Conclusiones parciales ..... 26
**Capítulo 2: Descripción del funcionamiento de un Servidor para Terminales Ligeras ..... 27**
2.1 Introducción ..... 27
2.2 Terminal Ligera ..... 27
2.3 Funcionamiento de una Terminal Ligera ..... 28
2.4 ¿Por qué PXE, DHCP, TFTP y NFS? ..... 29
2.5 Kernel Linux ..... 30
2.5.1 Soporte en el kernel del servidor y de las imágenes a utilizar en los clientes ..... 31
2.6 PXELINUX ..... 33
2.7 DHCP (Dynamic Host Configuration Protocol) ..... 34
2.8 TFTP (Trivial File Transfer Protocol) ..... 38
2.9 NFS (Network File System) ..... 39
2.10 Initramfs ..... 41
2.10.1 Initramfs en comparación con initrd ..... 41
2.10.2 Implementación ..... 42
2.10.3 Lenguajes ..... 44
2.10.4 Herramientas ..... 45
2.10.5 Bibliotecas ..... 47
2.10.6 Usos ..... 48
2.11 Ventajas de las Terminales Ligeras ..... 50
2.12 Requerimientos ..... 51
2.13 Conclusiones parciales ..... 52
**Capítulo 3: Descripción de la Solución ..... 53**

3.1 Introducción ..... 53
3.2 Arquitectura ..... 53
3.3 Configuración de la imagen servidora ..... 53
3.4 Configuración de la imagen cliente ..... 60
Conclusiones Generales ..... 67
Recomendaciones ..... 68
Referencias Bibliográficas ..... 69
Bibliografía ..... 71
Anexos ..... 73
Glosario de términos ..... 79

## Introducción

En la actualidad las terminales ligeras han alcanzado un desarrollo vertiginoso, aparejado al desarrollo de las redes y la informática, ganando cada día más significación por la gama de ventajas que traen consigo, ya sea en cuanto a ahorro de energía, facilidades de mantenimiento y administración. Por otra parte el mundo y los sistemas se mueven hacia la *Web* y para eso solo es necesario emplear un navegador, por lo que no hace falta utilizar grandes recursos computacionales. Las terminales ligeras son potencialmente una supersolución para socializar de forma más barata el empleo de la computación, lo que puede tener una gran repercusión en nuestro país pues Cuba ha identificado desde muy temprano la conveniencia y la necesidad de dominar e introducir en la práctica social las Tecnologías de la Información y las Comunicaciones; y lograr una cultura digital como una de las características imprescindibles del hombre nuevo, lo que facilitaría a nuestra sociedad acercarse más hacia el objetivo de lograr un desarrollo sostenible y más que nada a la informatización de todos los sectores.

Las terminales son ordenadores de bajas prestaciones que únicamente necesitan los periféricos pantalla, teclado y ratón, por lo que se puede reutilizar ordenadores viejos. 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. Lo que es realmente impresionante de todo esto es que se pueden tener montones de estaciones de trabajo, todas servidas desde un único servidor GNU/Linux. ¿Hasta cuántas estaciones puede preguntarse? Pues bien, esto depende del tamaño del servidor y de las aplicaciones que has de utilizar. Desde estos ordenadores los usuarios tienen acceso a Internet y otra serie de programas, como el paquete de ofimática, programas de conversación, de compresión de ficheros. Cada uno de estos ordenadores suele tener una instalación completa de una versión de Windows o Linux más el resto de aplicaciones que cada centro quiera ofrecer. Esto facilita la ampliación del número de terminales, ya que sólo es necesario enchufar la nueva terminal. Las terminales consumen menos energía y son más silenciosos. El reutilizar ordenadores viejos no genera o limita la generación de desechos inorgánicos y contaminantes.

El uso de software libre nos garantiza la libertad de acceso al conocimiento. Se busca con ello reaprovechar las computadoras viejas y/u obsoletas, valiéndose de este como medio. Variadas son las razones, entre ellas pueden mencionarse: la robustez del software y las actualizaciones periódicas, los bajos requerimientos, porque al tener acceso al código fuente es posible aprender de él y mejorarlo, y por sobre todo, la ya señalada libertad de acceso al conocimiento. El Software Libre es una de las revoluciones informáticas más grandes de los últimos años, muchos países han decidido migrar los escritorios de la administración pública, y también privada. Razones más que suficientes para utilizar la distribución Nova, la cual surgió por la necesidad de una plataforma que garantizara la compatibilidad de las aplicaciones que están en desarrollo con sistemas libres, además de las características propias de la metadistribución, y por considerarse en estos días la distribución de la sociedad cubana, por ser desarrollada en Cuba por un grupo de estudiantes de la Universidad de las Ciencias Informáticas.

Definitivamente se parte de la consideración de que la tecnología no ha penetrado más que superficialmente en las instituciones educativas, y en mayor medida, en las escuelas de enseñanza media. Lamentablemente el estado actual del parque informático en las escuelas no crea ningún entusiasmo a los docentes, pues la falta de computadoras que funcionen correctamente y la baja potencia de las mismas, es algo común. Se espera, por ello que el presente trabajo siente sólidas bases tecnológicas y que facilite la incorporación y renovación del parque informático a un costo muy bajo en relación a las soluciones tradicionales.

Comparar a las instituciones educativas con las empresas, salvando las obvias diferencias puede servir para contextualizar la problemática que se plantea cuando la actualización del parque informático se transforma en una necesidad imperiosa. En las empresas la incorporación de la tecnología se utiliza para lograr una producción más eficiente y, consecuentemente, disminuir los costos. Este concepto es posible trasladarlo a las escuelas. No implica eso afirmar que haya que reducir el tiempo empleado por los alumnos para trabajar con sus aprendizajes, sino de proveerles mejores recursos para potenciarlos.

Por otra parte, las PCs o computadoras personales de escritorios tradicionales ponen tanto poder en las manos de los empleados que estos pueden hacer todo tipo de actividades no permitidas, desde perder el tiempo en Internet hasta descargar accidentalmente virus. Por

esta razón, en el mundo, las organizaciones están eliminando las PCs de los escritorios y las están reemplazando por sistemas de "terminales ligeras". Se da a cada empleado una pantalla de computadora, un teclado y un ratón, pero es la computadora central la que almacena todos los datos y efectúa la mayor parte de los procesos, algo que reduce los costos de mantenimiento y vuelve mucho más fácil hacer un seguimiento y restringir cómo los trabajadores utilizan sus máquinas.

A lo largo de los años las terminales ligeras han aparecido y desaparecido de las oficinas. Ahora vuelven debido a los crecientes costos de mantenimiento de las redes y las exigencias a las empresas de tener más seguridad y mantener mejores registros. La firma de investigación de mercado IDC (*International Data Corporation*) pronostica que para el presente año, 2008, las terminales ligeras representarán casi el 10% del mercado de computadoras de escritorio en empresas grandes y medianas, cifra que contrasta con el 5,4% que representaban en el 2007. El número y variedad de las terminales ligeras ha estado creciendo. *Neoware Systems Inc.* presentó una que cuesta US$199 por terminal, una ganga frente a los US$800 que suele costar una PC. El fabricante californiano de chips *PMC-Sierra Inc*. anunció que estaba organizando un grupo de empresas de microprocesadores para que trabajen con fabricantes chinas para crear una computadora de red de US$150 que contenga software de fuente abierta. **[1]**

Estas cifras nos dan fe del significativo ahorro en divisas que representaría la implementación de las terminales ligeras en nuestro país. Se puede mejorar en gran medida la situación que existe con el parque informático de muchas instituciones educacionales del país, además de que el coste global del servidor es mucho menor que en sistemas tradicionales de servicios centralizados usando software propietario, ya que este tipo de software tradicionalmente requiere más recursos.

Pero, en general, el atractivo de las terminales ligeras no es su bajo precio. Algunas cuestan tanto como las PCs de escritorio, dependiendo, por ejemplo, de si utilizan el sistema operativo *Windows* o uno menos costoso. Algunas terminales ligeras suelen costar más que las PCs tradicionales debido al software y hardware que las conecta a la red central. Los ahorros se producen en la gestión de las computadoras, ya que los costos de mantenimiento bajan

radicalmente. Aunque en nuestro caso no debe existir preocupación por los costes en software pues por ser esta solución gratuita, supone un ahorro importante en pagos de licencias. Así, no debe desembolsarse cantidad alguna por el sistema operativo, pues Nova es desarrollado en nuestro país. Este concepto de ahorro es común a muchos de los sistemas libres. Incorporando nuestra solución a alguno de los escritorios libres, es posible añadir multitud de aplicaciones para entornos académicos, profesionales, domésticos sin que ello implique gastos económicos.

Las terminales ligeras proveen de 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). Puesto que los procesos no se ejecutan en las terminales sino en el servidor, que además alberga el software y los datos, la administración de la solución que se propone se reduce a la administración de un sólo equipo: el servidor. Desciende, por tanto, drásticamente el número de horas que debe dedicarse a mantenimiento y configuraciones, se simplifica la administración del sistema y de los datos, y se aumenta el control en todos los sentidos. Es importante mencionar el ahorro en mano de obra cualificada que esto significa, así como las mejoras en simplicidad y seguridad.

La implementación en Cuba de las terminales ligeras puede ser una magnífica solución para muchas empresas e instituciones por la situación económica que está viviendo el país y la escasez de recursos, pues los ahorros en *hardware* y personal de mantenimiento serían significativos, además de la recuperación de muchas computadoras que se encuentran en desuso por falta de piezas.

Como es posible apreciar, las redes de terminales tienen muchas posibilidades en todo tipo de infraestructura (aulas de informática, despachos, etc.). La posibilidad de centralizar la gestión de todos los equipos en el servidor lleva a la idea de gestionar varios, e incluso muchos, servidores de manera remota. Esto significa que, de una manera planificada, se podrían mantener muchas terminales con un equipo de personas muy reducido como se ha apuntado anteriormente.

**Problemas más frecuentes en los PCs tradicionales:**

Cada ordenador tiene instalado un sistema operativo.

Modificación de la configuración.

Sin una revisión técnica, en poco tiempo el ordenador deja de responder como uno espera.

Cada ordenador es un foco de virus.

Bajo nivel de seguridad.

No está siempre disponible la asistencia de un técnico informático.

Cada ordenador expuesto al uso de diferentes usuarios.

**Problema a resolver:** Inexistencia en la distribución Nova de una solución centralizada para terminales ligeras.

**Objeto de estudio:** El objeto de estudio de este trabajo son las soluciones para terminales ligeras.

**Idea a defender:** La elaboración de una solución informática para la instalación y configuración de terminales ligeras usando Nova posibilitará un mayor aprovechamiento de los recursos monetarios, de hardware y de software en entornos controlados.

**Objetivo general:** Elaborar una solución informática que permita la instalación y configuración de terminales ligeras usando Nova.

**Objetivos específicos:**

- Identificar las diferentes soluciones existentes de servidores de terminales ligeras.
- Analizar el proceso de instalación y configuración de terminales ligeras.
- Realizar el proceso de configuración de la imagen del servidor de terminales ligeras.
- Realizar el proceso de configuración de la imagen a usar por las terminales ligeras.

# CAPÍTULO I: Fundamentación Teórica de un Servidor para Terminales Ligeras

## 1.1 Introducción

En este capítulo se define el objetivo de la creación de un servidor de terminales ligeras con la distribución Nova, además de los sistemas más usados de terminales ligeras con sus características, ventajas, desventajas y tablas comparativas que dan una mayor visión de lo conveniente que sería usar estas terminales por las computadoras tradicionales. Además de algunas definiciones que son importantes resaltar.

**Tabla comparativa 1.1 Conceptos fundamentales**

| CONCEPTO | PC TRADICIONAL | TERMINALES LIGERAS |
|---|---|---|
| Costo | Mayor a los $800,00 dólares | Precio regular $199,00 dólares |
| Seguridad | Exposición de la información y vulnerabilidad a “hacking” y virus | No expone información, ni es vulnerable a hacking y virus |
| Mantenimiento | Dificultades para controlar y dar mantenimiento oportuno a cada computadora individual | Solo se requiere actualizar la PC principal (las terminales ligeras no requieren mantenimiento) |
| Energía | Superior a los 250 W por PC | Máximo 5W por TL (95% menos que una PC normal) |
| Ruido/Calor | El CPU genera ruido y calor. | Sin ruido, con emisiones eléctricas con un menor nivel de riesgo. |
| Acceso Remoto | Limitado acceso remoto | Se puede acceder a la PC principal desde cualquier sitio. |
| Actualización | Reposición de equipo cada 2 o 3 | Solo la PC principal necesita |

| | | |
|---|---|---|
| | años. | actualizaciones. |
| Espacio físico | Ocupan demasiado espacio físico | Diseño compacto que genera grandes beneficios de optimización de espacio físico |

### 1.2 Necesidad de crear un servidor para terminales ligeras en la distribución Nova

En los últimos años la Informática ha tenido un desarrollo vertiginoso en Cuba. Su campo de acción ha ido creciendo a la vez que la sociedad ha ido alcanzando una cultura adecuada a las necesidades y a los avances tecnológicos que se suceden a diario en el mundo entero.

Hoy Cuba enfrasca su trabajo a informatizar los sectores más fuertes, tal es el caso de la educación y la salud. Sin embargo la tarea no resulta tan fácil. Unido al desarrollo y la aplicación de la informática, se ha planteado la necesidad de sustentar el accionar informático en el Software Libre. Esto se debe en gran medida a la dependencia tecnológica que implica el Software Privativo y los obstáculos que engloba; sobre todo para los países del tercer mundo quienes se ven limitados por las costosas patentes. Otra ventaja que motiva la migración al Software Libre es la posibilidad que este ofrece de compartir los conocimientos alcanzados y los resultados obtenidos en aras de otros más novedosos.

Nova surge como idea de un grupo de estudiantes de la Universidad de las Ciencias Informáticas, con el objetivo de desarrollar una distribución local que se adapte a las necesidades de migración de servicios y aplicaciones de nuestra universidad. Por ser desarrollada en una universidad cubana permite que pueda ser orientada y optimizada acorde a las necesidades nacionales o propias de migración y desarrollo de *software.* Es basada en Gentoo y a pesar de ser muy joven posee características muy interesantes y fáciles de aprovechar además la línea de desarrollo que está siguiendo Nova es realizar las configuraciones que se proponen en este trabajo para que la distribución este preparada para soportar terminales ligeras.

Realizar un servidor de terminales ligeras en la distribución Nova trae consigo muchas ventajas, que pueden ayudar de una forma rápida y económica a resolver el ya conocido problema de hardware que presentan muchas instituciones cubanas. Por otra parte, debido a la adquisición del país de terminales ligeras para su utilización en diferentes campos, se necesitará soporte para su puesta en marcha, pues las terminales ligeras pueden ser muy baratas pero el costo del software que se utiliza para ponerlas en funcionamiento eleva su precio al de una computadora tradicional e incluso hasta un poco más.
En la actualidad en Cuba no se ha desarrollado ningún sistema similar al que se propone en este trabajo, lo que da fe de la necesidad de su estudio, pues las potencialidades de las terminales ligeras son cada día mayores.

### 1.3 ¿Qué es Software Libre?

Software Libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito **(libertad 0)**.
- La libertad de estudiar cómo funciona el programa, y adaptarlo a las necesidades propias **(libertad 1)**. El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias, con lo que se puede ayudar a otros usuarios **(libertad 2)**.
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. **(libertad 3)**. El acceso al código fuente es un requisito previo para esto.

Un programa es software libre si los usuarios tienen todas estas libertades. Así pues, se debería tener la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución, a cualquiera persona y en cualquier lugar. El ser libre de hacer esto significa (entre otras cosas) que no es un requerimiento pedir o pagar permisos.

También se debería tener la libertad de hacer modificaciones y utilizarlas de manera privada en el trabajo o el ocio, sin siquiera tener que anunciar que dichas modificaciones existen. Si se publican los cambios, no es preciso avisar a nadie en particular, ni de ninguna manera en particular.

La libertad para usar un programa significa la libertad para cualquier persona u organización de emplearlo en cualquier tipo de sistema informático, para cualquier clase de trabajo, y sin tener obligación de comunicárselo al desarrollador o a alguna otra entidad específica.

La libertad de distribuir copias debe incluir tanto las formas binarias o ejecutables del programa como su código fuente, sean versiones modificadas o sin modificar (distribuir programas de modo ejecutable es necesario para que los sistemas operativos libres sean fáciles de instalar). Está bien si no hay manera de producir un binario o ejecutable de un programa concreto (ya que algunos lenguajes no tienen esta capacidad), pero debe existir la prerrogativa de distribuir estos formatos si se encuentra o desarrolla la manera de crearlos.

Para que las libertades de hacer modificaciones y de publicar versiones mejoradas tengan sentido, se requiere tener acceso al código fuente del programa. Por lo tanto, la posibilidad de acceder al código fuente es una *conditio sine qua non* para el software libre.

Para que estas libertades sean reales, deben ser irrevocables mientras no se haga nada incorrecto; si el desarrollador del software tiene el poder de revocar la licencia aunque no se le haya dado motivos, el software no es libre.

Son aceptadas, sin embargo, ciertos tipos de reglas sobre la manera de distribuir software libre, mientras no entren en conflicto con las libertades centrales. Por ejemplo, *copyleft* es la regla que implica que, cuando se redistribuya el programa, no se pueden agregar restricciones para denegar a otras personas las libertades centrales. Esta regla no entra en conflicto con las libertades centrales, sino que más bien las protege.

***'Software libre'*** no significa '*no comercial*'. Un programa libre debe estar disponible para uso comercial, desarrollo comercial y distribución comercial. El desarrollo comercial del software libre ha dejado de ser inusual; el software comercial libre es muy importante.

Pero el software libre sin '*copyleft*' también existe. Existen razones importantes por las que es mejor usar '*copyleft*', pero si los programas son software libre sin ser '*copyleft*', también resulta posible utilizarlos.

Cuando se habla de software libre, es mejor evitar términos como: ` regalo' o `gratuidad', porque esos términos destacan como lo importante el precio, y no la libertad. **[2]**

### 1.4 GNU Nova

GNU Nova se creó originalmente para apoyar el proceso de migración de la Universidad de las Ciencias Informáticas. Una distribución de GNU/Linux propia para suplir las necesidades de la docencia, la producción y el trabajo de oficina bajo los estándares del software libre. Se creó a partir de Gentoo para aprovechar la flexibilidad que brinda este sistema. Con Gentoo como base se podría personalizar GNU Nova compilando los paquetes desde su base inicial de dependencias.

Actualmente GNU Nova es un proyecto donde se crean distribuciones de GNU/Linux a la medida. Se enfoca en obtener un sistema destinado a un objetivo específico cumpliendo con requerimientos pedidos por el usuario.

Parte del proceso de desarrollo de Nova consiste en la creación de una imagen base en la cual estarán las aplicaciones comunes para todas las soluciones. A partir de esta imagen base se instalará el software específico hasta llegar al estado final. Desde sus inicios a partir de Gentoo, GNU Nova ha usado Portage para la gestión de aplicaciones en el sistema, pero *emerge* se consideró no viable para los productos finales ya que la compilación es tediosa para el usuario estándar. Se tomó el acuerdo de que los usuarios finales obtendrían un sistema con Portage pero solo para mantener la compatibilidad con Entropy el manejador de aplicaciones perteneciente a Sabayon, un proyecto que mantiene relaciones con GNU Nova.

### 1.5 Gentoo: Distribución base de Nova

En el año 1999 *Daniel Robbins* se propuso crear una nueva distribución de GNU/Linux, a pesar de no contar con suficientes recursos que le permitieran competir con *Debian o Red Hat*. Por tal motivo se decidió a automatizar el proceso de crear los paquetes lo más comprimidos posible. La distribución se crearía a sí misma desde cero; los usuarios seguirían el mismo proceso y compilarían todos sus binarios, eliminando la grasa superflua y quedándose con un sistema optimizado al máximo. El motivo de que esta distribución se tardara tanto en aparecer, se debió a sus propias particularidades.

El nombre de Gentoo viene de uno de los pingüinos más veloces en el agua, sin duda refiriéndose a lo rápido que van los binarios compilados para nuestra máquina y nuestras necesidades. Esta distribución no se limita a empaquetar unos cuantos programas específicos de algún campo. Más bien intenta ser una distribución universal; y con más de 10,000 paquetes y dando soporte a 6 arquitecturas.

### *1.5.1 Ventajas de Gentoo*

- Tiene un sistema totalmente optimizado y personalizado. Mediante el fichero *make.conf* se definen los criterios por defecto a la hora de compilar, como librerías o arquitectura del ordenador a usar. Así, si se tiene un ordenador antiguo sin grabadora ni nada, los programas se instalarán sin soporte para la misma ahorrando tanto espacio como recursos.
- El *kernel*, el cual también se tiene que configurar, tiene soporte solo para lo que le interese al usuario. Además se pueden poner los módulos embebidos, ahorrando tiempo de carga.
- Gentoo, además, debe ser instalado a piezas desde cero, por lo que no se tendrán programas innecesarios, lo que significa que se puede decidir cual usar. No tiene nada por defecto, pudiendo elegir cualquier gestor de ventanas si necesidad de bajar versiones especificas.
- A la hora de compilar, Gentoo le da la posibilidad al usuario de poder hacer que los programas se adapten a su gusto y personalizar cada paquete de forma individual.
- Gentoo Cuenta con grandes repositorios, los cuales son actualizados con gran rapidez. Al usar códigos fuente resulta muy fácil añadir nuevos programas. De aun no tener suficiente,

podremos recurrir al *overlay*, que son los no oficiales. Debido al sistema tan sencillo de gestión de paquetes mediante *ebuilds* (pequeños documentos con las dependencias del programa, lugar de descarga y pequeño *script* de instalación), puedes bajarte un *ebuild* casero y añadirlo al repositorio sin muchos problemas.

### *1.5.2 Desventajas de Gentoo*

- La principal ventaja de Gentoo también es su principal desventaja. El hecho de tener que compilar todo puede volverse muy cansado, pues generalmente es un proceso lento que puede llegar a tardar hasta varias horas por paquetes (*openoffice* por ejemplo).
- Hay que instalar a mano prácticamente todo, y algo tan sencillo como configurar el ratón puede volverse una locura si no se ha configurado correctamente el *kernel*.
- Es un sistema totalmente vacío que se tendrá que rellenar, y cualquier cosa, por simple y tonta que parezca, si no se configura, no estará. Da igual que parezca algo fundamental, habrá que instalarlo, como el cliente dhcp o los *drivers* del teclado para las X. Esto puede provocar grandes errores que le pueden hacer perder al usuario gran cantidad de tiempo.
- Poner en marcha un sistema completo, o actualizar un sistema que ha estado desatendido durante una temporada, puede requerir una respetable cantidad de tiempo (horas o incluso días si el ordenador es muy antiguo), mientras se descargan y compilan todos los paquetes nuevos.

A pesar de las desventajas se puede concluir que Gentoo permite por regla general una actualización sin problemas, a diferencia de otras distribuciones donde puede llegar a resultar complicado o casi imposible. Esta actualización también es posible a partir de binarios precompilados, lo que requiere menos tiempo.

Gentoo da una sensación de dinamismo controlado bastante curiosa: está siempre a medio hacer, y aún así consigue generar sistemas estables con las últimas versiones de los paquetes. Es un hecho que cada día son más los usuarios que se deciden a instalar esta distribución en su computadora y parece que muchos de ellos terminan dejándola ahí, quitando el protagonismo a su anterior distribución. Su aproximación minimalista a la configuración de paquetes permite saber exactamente qué realmente se tiene en la máquina.

Si se quiere un sistema adaptado al gusto particular de cada cual y se tienen algunos conocimientos previos de GNU/Linux, Gentoo es la distribución ideal para lograrlo. **[3]**

### *1.5.3 Características específicas*

- Posee un sistema de fabricación de binarios, *ebuild*, el cual tiene su origen en el software que diseñó Robbins para ayudarle a hacer su primera distribución; además de los *flags* para el compilador, se puede elegir qué dependencias incluir, globalmente y para paquetes individuales. Esto evita dependencias a interfaces determinadas y software no deseados por el usuario (como por ejemplo el software arts).
- Otra novedad que posee a nivel mundial es *Portage*. Se trata de una grandísima colección de software indexado, categorizado y aún mejor: verificado. El sistema se quejará y no dejará instalar los paquetes que estén en estado *masked*, es decir "enmascarados". Ciertos programas no funcionan en algunas arquitecturas (por ejemplo, OpenOffice.org versión 1.x no compilaba para amd64). Otros no están todavía probados; se quedan en cuarentena hasta que se verifique que funcionan. Hay incluso paquetes incompatibles entre sí: no se pueden tener varios servidores de correo andando al mismo tiempo. Así, cuando el software pedido no cumple ciertas condiciones, se "enmascara" para que no cause problemas.
- Posee una herramienta que hace que el trabajo de compilar el núcleo del sistema sea mínima, *Genkernel*, la cual hace poco fue portada para otras distribuciones como *Debian*, y que tiene la característica de que autodetecta el hardware y automáticamente construye un *kernel* específico para este, además, en caso de que se quieran agregar módulos adicionales que provean determinado soporte, se puede hacer mediante un menú en modo gráfico. **[4]**

## 1.6 Principales Sistemas para Terminales Ligeras libres

Con el aumento del desempeño de las redes, se ha masificado el uso de las terminales ligeras en aplicaciones comerciales de alta demanda, su uso principal es como POS (*Point of*

*service*). Principalmente gracias a sus facilidades administrativas y capacidades de personalización. Aquí describimos algunas de ellas con sus características principales y algunas tablas comparativas de acuerdo a su funcionalidad.

### *1.6.1 LTSP (Linux Terminal Server Project)*

LTSP es el acrónimo de *Linux Terminal Server Project (www.ltsp.org)*. Este proyecto fue fundado por Jim McQuillan en 1999 con la intención de que ordenadores con escasos recursos pudieran acceder a un ordenador central, ejecutando todas las acciones en ése en vez de en el propio equipo. Dicho de otra manera, convertir los PCs tradicionales en simples terminales del servidor central. LTSP es un proyecto libre de comunidad, sin duda el más popular relacionado con terminales ligeras, y su impacto en determinados sectores de la informática es enorme.

Tras arrancar la terminal y obtener su dirección IP y la localización en el servidor del núcleo que debe cargar, la terminal obtiene el núcleo mediante TFTP (Trivial File Transfer Protocol) y lo ejecuta. Este núcleo del LTSP lleva una imagen de un sistema de archivos que es cargada en memoria como un *ramdisk* (parte de memoria que es asignada para usarla como si se tratase de una partición de disco) y lanza una serie de *scripts* (que serán comentados más adelante) que montarán el sistema de archivos raíz que se haya preparado en el servidor a través de NFS (Network File System). Finalmente, se iniciarán las X Window y se enviará una petición XDMCP al servidor, que permitirá ingresar en el servidor.

Con lo cual, se tiene el sistema de archivos raíz montado mediante NFS desde el servidor GNU/Linux. Esto es una diferencia importante respecto al proyecto PXES que se comentará a continuación, ya que PXES no es dependiente del sistema operativo del servidor (lo que permite que pueda ser ejecutado en sistemas Windows, por ejemplo), y no monta el sistema de archivos raíz mediante NFS.

Anteriormente se ha descrito someramente la carga del sistema LTSP por la terminal, pero sin entrar en los detalles concretos que se definen a continuación y que permiten obtener una visión clara del proceso que hay que seguir para lograr el objetivo propuesto:

1. La terminal arranca y mediante Etherboot realiza una petición DHCP a la red, que es respondida por el servidor DHCP que le proporciona su IP y la localización del núcleo a descargar.
2. Mediante TFTP la terminal contacta con el servidor y se descarga el núcleo, que es cargado en memoria y al que se le pasa el control a continuación.
3. El *kernel* inicializa el sistema y los periféricos que reconozca.
4. El *kernel* carga una pequeña imagen *ramdisk* en memoria y la monta temporalmente como sistema de archivos raíz.
5. El *kernel* ejecuta el *script linuxrc* (/linuxrc) que realiza los siguientes procesos
    - Busca en el bus PCI alguna tarjeta de red. Por cada dispositivo PCI que encuentre, realiza una búsqueda en el archivo niclist (/etc/niclist) para encontrar alguna coincidencia. Una vez encontrada una coincidencia, el nombre del módulo de driver NIC es guardado para posteriormente cargarlo en el kernel. Si la tarjeta es ISA, el nombre del módulo del driver debe ser indicado en la línea de comandos del kernel.
    - Ejecuta el cliente DHCP dhclient que realiza una nueva petición DHCP para hallar la ruta del directorio raíz a montar por NFS.
    - Dhclient recibe la información DHCP del servidor y ejecuta el script dhclient-script (/etc/dhclient-script), que configura la interfaz de red eth0 con la información obtenida.
    - En este momento el sistema de archivos raíz está montado en la RAM, por lo que en este momento se monta un nuevo sistema de archivos raíz mediante NFS desde el servidor (por defecto el directorio exportado es /opt/ltsp/i386. Para montar este directorio como raíz el script linuxrc realiza pivot_root (intercambio del sistema de archivos raíz), por lo que el sistema de archivos NFS será montado como /, y el sistema de archivos anterior será montado en /oldroot.
6. Se ejecuta el init (/sbin/init), que realiza los siguientes procesos
    - Init lee el fichero inittab (/etc/inittab) y de acuerdo a éste comienza a preparar el sistema.
    - Se ejecuta el comando rc.local mientras el sistema esté en el estado sysinit.

- El script rc.sysinit crea un ramdisk de 1MB para almacenar lo que vaya a ser escrito ó modificado. Este espacio será montado como /tmp
- El sistema de archivos /proc es montado.
- Se lee el fichero de configuración lts.conf (/etc/lts.conf), cuyos parámetros serán establecidos como variables de entorno para usar por el script rc.sysinit.
- Se crea el archivo de intercambio swap y se habilita mediante el comando swapon.
- Se configura la dirección de red loopback (127.0.0.1).
- Se monta el directorio /home del usuario.
- Se crean el directorio /tmp y subdirectorios donde se guardarán los archivos temporales y se crea en él el fichero syslog.conf (/tmp/syslog.conf) que contiene información de a qué host de la red debe enviarse la información de los logs.

7. Se cambia el runlevel a 5, con lo que se ejecutarán todas las instrucciones contenidas en inittab (/etc/inittab)
    - Se inicia una sesión de las X Windows System con el comando startx (/etc/screen.d/startx), que proporciona al usuario una interfaz gráfica.
    - El servidor de las X Windows System enviará una petición XDMCP (X Display Manager Control Protocol) al servidor XDM (X Display Manager) que le responderá con una pantalla de inicio de login de usuario. **[5]**

### *1.6.2 PXES (Universal Linux Thin Client)*

PXES (Universal Linux Thin Client) es un proyecto iniciado por Diego Torres y que en este momento ya ha alcanzado la versión 1.0. Es un proyecto más reciente que el de LTSP, e incorpora interesantes variaciones respecto a éste.

Tras arrancar la terminal y obtener su dirección IP y la localización en el servidor del núcleo que debe cargar, la terminal obtiene el núcleo de la minidistribución PXES, que se ejecuta íntegramente en la memoria RAM de la terminal.

Con lo cual, tenemos el sistema de archivos montado en la memoria RAM de la terminal. Además, PXES viene con una utilidad PxesConfig, que permite crear fácilmente imágenes a medida para el hardware y prestaciones que se requieran de la terminal, permitiendo que la terminal arranque distintos tipos de sesiones, como XDMCP, sesión RDP en un servidor Microsoft Windows ó una interesante opción que consiste en iniciar una sesión local de X Windows con un escritorio muy simple que comentaremos más adelante.

La principal diferencia entre estos dos proyectos radica en el montaje del sistema de archivos raíz, que PXES lo hace de forma local en la RAM mientras que LTSP hace uso del NFS para montarlo a través del servidor, lo que puede provocar un incremento considerable de la carga de red y del servidor si no se realiza adecuadamente. Sin embargo, PXES está limitado por la memoria RAM de la terminal ligera, que debe ser suficiente para permitir el montaje de todo el sistema de archivos, mientras que con LTSP al utilizar NFS permite una mayor flexibilidad en este aspecto. Con lo cual se tiene que estudiar detenidamente las características de cada proyecto concreto para elegir la solución que más conveniente resulte.

**Funcionamiento**

Cuando la terminal ligera se inicia envía una señal a la red identificándose con la MAC de su tarjeta de red y permanece a la espera de indicaciones de un servidor DHCP a la escucha, como está comentado en el apartado de arranque por red.

Si el servidor reconoce esta MAC enviará los datos de asignación de red a ese equipo (nombre de host, ip, rutas, máscara de red, etc.) y a continuación le enviará a través de FTP los archivos indicados en su configuración.

En ese momento el cliente pide, vía el servidor que el dhcp le ha indicado, una imagen de boot loader (NP). Más tarde la terminal recibe todo el sistema operativo necesario para el funcionamiento del cliente; este sistema operativo se ejecutará íntegramente en la memoria RAM de la terminal ligera.

Básicamente hay tres tipos de imágenes: la imagen Etherboot (con extensión .nbi), las imágenes pxes (con extensión .initrd) y las imágenes squashfs (con extensión .squashfs (comprimidas). Luego también veremos que se puede usar imágenes Etherboot o imágenes pxes especialmente preparadas para usar con GRUB.

Finalizada la carga de la microdistribución PXES, el cliente podrá mostrar el login gráfico que esté ejecutando el servidor. Si existen cuentas de usuario, se podrá trabajar con las aplicaciones que estén autorizadas, teniendo en cuenta que todas las aplicaciones se ejecutan y guardan en el servidor y que el ordenador local sólo es usado para mostrar en pantalla el resultado de las operaciones realizadas en el servidor.
La manera más rápida de tener PXES funcionando es usar las imágenes preconfiguradas (PreBuilt) disponibles en la Web oficial. Esta solución es ideal también para usuarios de Windows, ya que no es necesario un entorno linux para construir las imágenes, tan sólo habrá que configurar los servicios DHCP y TFTP.
Dependiendo si el arranque es a través de PXE o de disquete se bajarán las imágenes .initrd o bien .nbi respectivamente. Además también se encuentra disponible asimismo una imagen ISO lista para grabar en cd-rom .iso. **[5]**

### *1.6.3 2X ThinClientServer*

2X ThinClientServer hace que el movimiento computación de una terminal ligera sea fácil al entregar una solución que convierte PCs existentes en terminales ligeras y además ofrece administración centralizada de los dispositivos de una terminal ligera de cualquier proveedor (HP, Neoware, Wyse, Maxspeed o más). Las configuraciones de conexiones de usuario, dispositivos de hardware (RDP / ICA / NX), tamaño de pantalla, aplicaciones a las que el usuario tiene acceso, servidores de Terminal y VMware escritorios virtuales, pueden ser controladas centralmente con base a dispositivo, usuario, grupo o departamento (Directorio Activo / Cuentas Locales), todo esto a través de una interfaz basada en Web. La edición 2X ThinClientServer Enterprise, incluye soporte para servidores Citrix y además servicio de soporte técnico, contrario a la edición gratuita 2X ThinClientServer PXES.

***Características principales de 2X ThinClientServer***

- Convertir PCs existentes en terminales ligeras.
- Administrar centralmente configuraciones de conexión de usuario con base en usuario, grupo o departamento.

- Limita el acceso a usuarios a Citrix o aplicaciones publicadas con 2X, en vez de darle acceso a todo el escritorio.
- Independiente del proveedor de terminales ligeras: Administra centralmente cualquier terminal ligera o PC.
- Soporta virtualmente todas las terminales ligeras y hardware de computadores.
- Ejecuta aplicaciones publicadas en Citrix o 2X en el mismo escritorio.
- Múltiples escritorios completos por terminal ligera.
- Soporte para escritorios publicados con 2X.
- Soporte para impresión y redirección de sonido en aplicaciones publicadas con 2X.
- Más soporte a hardware con Linux kernel 2.6.18.2.
- Soporte de motor de conexión en caliente.
- Mejorada apariencia y versatilidad del escritorio administrado.
- Soporte de auto inicio de sesión de terminal ligera.
- Soporte de vigilancia de terminal ligera.
- Herramienta de reporte de utilización por usuario.
- Herramienta de reporte de utilización por cliente. **[6]**

### *1.6.4 Diet-PC (Diskless Embedded Technology Personal Computer)*

Diet-PC (Diskless Embedded Technology Personal Computer) consiste en un sistema operativo Linux embebido que se ejecuta por completo en la memoria RAM de la terminal ligera. Este sistema es descargado del servidor de imágenes mediante TFTP. El sistema lanza un pequeño programa que se conecta al servidor a través de un protocolo IP, de modo que el cliente pueda ejecutar un entorno gráfico como X11, RDP, etc.

Diet-PC no tiene la facilidad de configuración que otros proyectos de similares características como PXES ó LTSP, ya que está pensado para que los desarrolladores puedan seleccionar los componentes necesarios para su sistema y así optimizarlo a sus necesidades. A diferencia de los proyectos anteriormente comentados, Diet-PC no es configurado a través de un archivo

central de configuración, sino que realizará dicha configuración basándose en la detección automática del sistema local y en una mínima dependencia del servidor.
Otro punto importante es que el sistema Linux que carga la terminal está adecuado a los estándares Linux en lugar de utilizar alternativas recortadas que emplean otras soluciones. Por lo tanto, el rendimiento del sistema puede ser inferior al de otros métodos, requiriendo una mayor cantidad de memoria RAM en la terminal que otras alternativas.
Diet-PC puede servirse desde servidores Windows además de Linux, utilizando un protocolo de ventanas Windows (RDP por ejemplo). **[5]**

**Tabla comparativa 1.2 LTSP vs PXES vs Diet-PC**

| | **LTSP** | **PXES** | **Diet-PC** |
|---|---|---|---|
| Versión | LTSP 4.1 | PXES 1.0 | Diet-PC 2.0 |
| S.O. admitidos | Linux | Linux, Windows | Linux, Windows |
| RAM en el cliente | 32 MB | 16-32 MB | 32-64 MB |
| Dispositivos locales | Disco duro, diskette, CD-Rom, impresora | Disco duro, diskette, CD-Rom, impresora | |
| Montaje del sistema de archivos raíz | Montaje por NFS | RAM del cliente | RAM del cliente |
| Sesiones admitidas | XDMCP | XDMCP, escritorio local, RDP, NoMachine NX, FreeNX, ICA, VNC | XDMCP, RDP |
| Configurador | Configurador modo texto | Configurador gráfico (PxesConfig) | Sin configurador |

### *1.6.5 Netstation*

Es una distribución de Linux que permite convertir computadoras en *thin clients* que soportan la gran mayoría de los protocolos de conectividad. Permite arrancarlos desde la red o desde un dispositivo como diskete, CD, disco duro o flash.

Los protocolos que soporta esta distribución son bastante amplios (X- Terminal XDM, TightVNC, SSH, Citrix ICA, Tarantella,...). Se trata pues, de múltiples clientes accediendo concurrentemente usando administración local de ventanas.
Permite la autodetección de la tarjeta de red, tarjeta de vídeo y ratón. También se puede destacar que soporta paquetes ".nps" dinámicos (puede cargarse en tiempo de ejecución). Dispone de configuración centralizada usando TFTP para obtener los ficheros de configuración facilitando el mantenimiento. **[5]**

### *1.6.6 Thinstation*

Thinstation es un distribución en Linux para *thin client*, para convertir un PC en un *thin client* soportando la gran mayoría de los protocolos de conectividad: Citrix ICA, No Machine NX, MS Windows Terminal Services (RDP), Tarantella, X, telnet, tn5250, VMS term y SSH. Puede ser arrancado por red, usando Etherboot/PXE o desde un dispositivo local floppy/CD/HD/flash-disk. La última versión es 2.0.2 (27 de mayo de 2005). Permite generar imágenes RAM personalizadas sin instalar nada en los servidores.
Comparte con Netstation el acceso de múltiples clientes trabajando concurrentemente usando administración local de ventanas. Permite, como Netstation, la autodetección de la tarjeta de red, tarjeta de vídeo y ratón. Puede soportar paquetes ".pkg" dinámicos (puede cargarse en tiempo de ejecución). Dispone también de configuración centralizada usando TFTP para obtener los ficheros de configuración. Soporta Samba para compartir discos e impresoras de las terminales ligeras. **[5]**

## 1.7 Principales Sistemas para Terminales Ligeras bajo licencia

Existen varias alternativas no libres para crear una red de terminales ligeras creadas por compañías con fines comerciales. Entre ellos se destacan:

### *1.7.1 eLux NG*

Es un sistema operativo embebido basado en Linux. El usuario y el administrador no necesitan conocimientos de Linux para utilizar o configurarlo. La interfaz de usuario es amigable y se puede configurar fácilmente mediante paneles de control. El sistema operativo es muy compacto para dejar capacidad a las aplicaciones y lograr un arranque rápido del *thin client*. El eLux NG es una solución de sobremesa completa, que facilita al usuario un acceso rápido, confortable y seguro a Windows y otros servidores en un ambiente cliente/servidor. En un ambiente cliente/servidor las aplicaciones se ejecutan en un servidor central. En la terminal, "*thin client*" o terminal ligera, se instala una aplicación cliente. Con esta aplicación el *thin client* se conecta al servidor correspondiente. Este sistema permite el acceso a multitud de plataformas, basadas en RDP, ICA o X entre otras. **[5]**

### *1.7.2 Citrix Metaframe*

Uno de los productos más populares en entornos de empresa. Con la única instalación de un cliente (independientemente del sistema operativo utilizado) de Citrix se puede acceder a todo el juego de aplicaciones de la empresa, estando estas solo instaladas en un servidor. Así pues, Citrix proporciona un nivel de acceso centralizado y seguro para la gestión de los datos empresariales más importantes. Utiliza el protocolo ICA (siglas en inglés de Arquitectura de computación independiente). **[5]**

### *1.7.3 Terminal Services*

Es la opción proporcionado por Microsoft en algunos de sus productos para la instauración de entornos de terminales ligeras. Se basa en el protocolo RDP, sin admitir otras opciones. Comenzó con Windows NT for Terminal Services y actualmente existen versiones avanzadas de los sistemas operativos de Microsoft (Windows 2000, Windows XP) que incluyen soporte de este protocolo como servidor. En cuanto a la parte cliente es fácilmente disponible de forma

gratuita desde la página Web de la propia empresa, que incluso tiene en cartera de productos la salida al mercado de un sistema operativo optimizado para trabajar como terminal ligera.

### *1.7.4 Neoware*

En realidad no es un solo producto como tal, sino una empresa que dispone de multitud de soluciones relacionadas con las terminales ligeras, tanto hardware como software para acoplar a ellos. Entre estos productos cabe destacar su sistema operativo Linux personalizado, el software ezManager para administrar terminales ligeras o Teemtalk que sirve para conectar con casi cualquier entorno terminal ligera/servidor.

### *1.7.5 Wyse*

Es similar a Neoware, una empresa de soluciones para terminales ligeras que facilita tanto hardware como software. En este caso la solución comercial que nos proporciona basada en Linux recibe el nombre de WinTerm Linux y su sistema de administración de terminales ligeras, Wyse Rapport.

### *1.7.6 WinConnect Server XP*

Es una solución de software que convierte el computador anfitrión Windows XP que no dispone del servicio "*Terminal Services*" de Microsoft en un servidor RDP permitiendo que dispositivos como terminales, aplicaciones de Internet/Información, Tablet PCs y PDAs, puedan conectarse con él para ejecutar aplicaciones de Windows simultánea e independientemente a través de cualquier red. La solución es similar por lo tanto a la de Microsoft, pero disminuyendo el coste adicional. No obstante, plantea algunas mejoras respecto al sistema de Microsoft, como son la posibilidad de trabajar con mayor número de

colores o de que el flujo de audio MP3 o WAV generado en el servidor suene en la terminal ligera. **[5]**

**A continuación se muestran varias tablas en las que se resumen todas las alternativas vistas anteriormente con su información al detalle:**

**Tabla comparativa 1.3**

**Hardware de las terminales ligeras**

| | RAM Mínimo | RAM Recomendado | RAM Óptimo | Ratón | Soporte de sonido |
|---|---|---|---|---|---|
| LTSP | 16 MB | 32 MB | >32 MB | Serial o PS/2 | Sí |
| PXES | 16 MB | 32 MB | >32 MB | Serial o PS/2 | Sí |
| DIET-PC | 32MB | 64MB | 64 MB | Serial o PS/2 | Sí |
| Netstation | 8MB usando TinyX | 16 MB / 32MB | 32MB | Serial o PS/2 | No |
| Thinstation | 8MB usando TinyX | 16 MB / 32MB | 32MB | Serial PS/2 y USB | No |

**Tabla comparativa 1.4**

**Dispositivos locales en la Terminal Ligera**

| | Diskette | Disco Duro | CD-Rom | Impresoras | Dispositivos Serial | Audio | Memoria de Almacenamiento Flash USB |
|---|---|---|---|---|---|---|---|
| LTSP | Sí | Sí | Sí | Paralelo y USB | No | Sí | No |
| PXES | Sí | Sí | Sí | Paralelo, Serial y Usb | Lectores de Códigos de | Sí | No |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | Barras | | |
| Netstation | Sí | Sí | Sí | Paralelo y Usb | No | Sí | Sí |
| Thinstation | Sí | Sí | Sí (BlackBox) | Paralelo y Usb | No | Sí | Sí |
| DIET-PC | Sí | Sí | Sí | Paralelo y Usb | No | Sí | No |

**Tabla comparativa 1.5 Otras características**

| | **Sesión de texto (Telnet ó SSH)** | **Múltiples clientes simultáneos usando administrador de ventanas local** | **Autodetección de tarjetas de red, video y mouse** | **Administración centralizada usando TFTP para obtener los archivos de configuración** | **Administración remota de los clientes vía Telnet SSH** |
|---|---|---|---|---|---|
| LTSP | Sí | Sí | Sí | No | No |
| PXES | Sí | Sí | Sí | No | No |
| Netstation | Sí | Sí | Sí | Sí | No |
| Thinstation | Sí | No | Sí | Sí | No |
| DIET-PC | Sí | No | Sí | Sí | No |

## 1.8 Conclusiones parciales

En la búsqueda realizada de soluciones de factible implementación, y luego de un estudio exhaustivo de las tecnologías ofrecidas por el mercado (privativo y libre), sin duda la balanza se inclina hacia soluciones de código abierto (libres).

Es el momento oportuno de aprovechar los desarrollos basados en Software Libre, debido a la necesidad actual y la realidad en la que estamos inmersos. De esta manera se abrirían las puertas de la tecnología por igual, sin distinción entre países del primer y tercer mundo.

# Capítulo 2: Descripción del funcionamiento de un Servidor para Terminales Ligeras

## 2.1 Introducción

Este capítulo explica cómo funcionan y la ventajas generales asociadas a la solución para terminales ligeras que se propone. Se incide posteriormente en aquellas características y configuraciones asociadas a estos sistemas.

## 2.2 Terminal Ligera

Las terminales ligeras son estaciones de trabajo conectadas a un servidor central. Siguiendo una estructura de cliente/servidor el cliente (terminal) es una ventana hacia el servidor central donde se encuentran instalados el sistema operativo y los diferentes programas. Esta tecnología hace posible el acceso a múltiples terminales (monitor, teclado, ratón) de bajas prestaciones utilizando un único equipo completo (servidor central).

**Figura 1.1 Terminal ligera**

## 2.3 Funcionamiento de una Terminal Ligera

Al arrancar un ordenador, el equipo lee la información del disco duro que necesita para que se pueda operar con él. Así, están el sistema operativo y las aplicaciones instalados en la propia máquina. La idea de esta propuesta es que el sistema operativo así como el escritorio y las aplicaciones se encuentren en un servidor. En el cliente se almacena en memoria solo las partes que pueden sufrir cambios durante el proceso de corrido del sistema. Todos los servicios y aplicaciones se ejecutan en el servidor. De esta manera, el cliente no necesita disponer de unidad de almacenamiento y, además, al no ejecutar más que una mínima parte de las tareas, no es necesario que sea una máquina muy potente. Un viejo Pentium I bastaría para utilizar infinidad de aplicaciones actuales de manera plenamente satisfactoria.

Con esta tecnología, un usuario puede acceder a escritorios Linux y navegar, realizar trabajos ofimáticos, escuchar música, etc. Además permite a un usuario utilizar una terminal ligera en multitud de entornos con un rendimiento análogo al de cualquier PC y contrariamente a lo que pudiera parecer, no es necesario disponer de un servidor demasiado potente.

Existen otros proyectos con objetivos y tecnologías similares a la solución que se propone en este trabajo, tanto libres como propietarios. La idea de las terminales ligeras no es nueva y multitud de compañías la han explotado desde el principio de la informática. De hecho, esta arquitectura de "terminales tontos" es anterior a la aparición del PC, si bien, fue reemplazada en los 80 por los ordenadores de sobremesa ahora experimenta un resurgimiento. **[7]**

**FUNCIONAMIENTO INTERNO TÉCNICAMENTE**

El BOOTING DISKLESS requiere un servidor de DHCP para determinar de una MAC suministrada su ubicación en la red y darle una IP fija. Además se requiere una placa base que permita BOOTEO PXE por red. Un servidor TFTP para poder trasmitir el ROOM de arranque PXE y posteriormente pasar por el mismo servidor el kernel del sistema (el núcleo del mismo) el cual se configurará según los argumentos especificados en la configuración del

PXE , para que después de esto el kernel sepa donde buscar los ficheros de sistema en el servidor . Los cuales se trasmitirán por NFS.

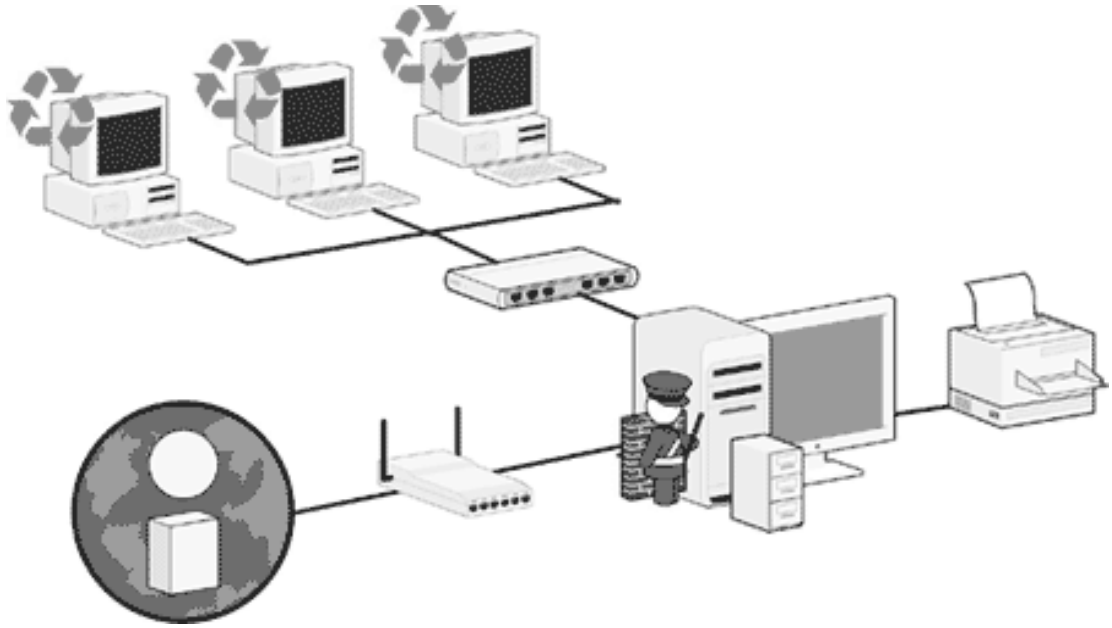


**Figura 1.2 Estructura de un servidor para terminales ligeras**

## 2.4 ¿Por qué PXE, DHCP, TFTP y NFS?

La primera vez que se inicia un nuevo equipo cliente habilitado para inicio remoto PXE, éste utiliza el protocolo DHCP para solicitar una dirección de Protocolo Internet (IP) y la dirección IP de un servidor DISKLESS BOOTING. Como parte de la solicitud inicial el servidor DHCP le facilita la conexión con el servidor TFTP un protocolo utilizado para transmisión de datos en red, una vez es pasado el ROOM PXE se genera un micro entorno el cual es capaz de cargar el kernel también pasado por TFTP y luego a su vez el kernel ya es capaz de por NFS solicitar todos los archivos del S.O y completar el arranque del mismo. **[8]**

## 2.5 Kernel Linux

Primero que todo es importante conocer qué es un kernel y algunas de sus principales características. El kernel extraoficialmente pudiera decirse que es una multi-aplicación o integración de muchas de estas que son ejecutadas en un mismo espacio de tiempo, una de sus principales tareas es la de garantizar un medio de ejecución (tal vez multitareas) para otras aplicaciones, con el objetivo de que estas últimas puedan ser ejecutadas. Linux es el principal intermediario entre un sistema operativo y su entorno de hardware. A la hora de iniciar un sistema operativo, el primero que cobra vida es el kernel ya que como puede deducirse, es la base o sostén de cualquiera de estos.

El kernel Linux es una gran pieza ejecutable compuesta por un determinado número de segmentos más pequeños que a su vez también son ejecutables, una de las razones por las que se dice que este es monolítico. Algunos de estos componentes se pueden separar del kernel convirtiéndolos en módulos de forma tal que siempre van a tener un lugar reservado por el kernel a la hora de ser cargados (ver anexo 1). El punto es que cuando el kernel se ejecuta, detrás se ejecutan todos sus componentes internos, y si posee gran cantidad de estos, pues ocuparía mucho espacio en memoria (hasta 60MB aproximadamente), sería poco eficiente, y hasta podría provocar algunos conflictos en la detección del hardware. Por tal razón, una buena práctica de compilación del kernel Linux, es compilar un kernel pequeño con una gran cantidad de módulos.

A la hora de iniciar el sistema operativo, los programas de detección de hardware juegan un papel muy importante, estos son los encargados de reconocer automáticamente todos los dispositivos existentes y crear reportes que luego son usados por otros programas que lo necesitan, un ejemplo importante es el reciente *udev*. En algunos casos, este tipo de software es capaz de cargar automáticamente determinados módulos cuando se les necesite.

El initramfs tiene mucho que ver en este asunto, ya que tiene que ser capaz de detectar tarjetas de red, y diferentes medios de almacenamientos para poder iniciar el sistema operativo (ver anexo 2), se le cambió el estilo para que el lector pueda entender mejor de

forma ilustrativa la secuencia o proceso en que se inicia un sistema operativo haciendo uso del kernel y del initramfs.

A grandes rasgos cuando se enciende el ordenador:

(1) El BIOS encuentra el gestor de arranque alojado en alguno de los medios de almacenamiento mediante el selector de inicio.

(2) El gestor o cargador de arranque descomprime el kernel con sus parámetros y el initramfs (en caso de que se use) en la memoria RAM y les delega el control.

(3) Una vez que el kernel se encuentra en ejecución, decide mediante los parámetros pasados por línea de comandos si ejecutar primero el initramfs o el sistema operativo directamente. (3.1) si se ejecuta el initramfs, este hace algunos preparativos iniciales y realiza el paso 4.

(4) Comienza el proceso de inicio real del sistema operativo.

Después de la anterior explicación, se puede concluir que el initramfs no tiene sentido sin el kernel Linux. Su principal tarea consiste en servir de puente o transición entre el kernel y el sistema operativo. ¿Será posible omitir este puente?, la respuesta es sí, pero en muy pocos casos, ya que precisamente en este puente radica la potencia de los sistemas que usan esta técnica. **[9]**

### *2.5.1 Soporte en el kernel del servidor y de las imágenes a utilizar en los clientes*

Para garantizar el soporte adecuado en el servidor son necesarias las siguientes opciones marcadas en la configuración del kernel:

**[*] Networking support**

NET = y

**[*] TCP/IP networking**

INET = y

**[*] IP: multicasting**

IP_MULTICAST = y

**<*> NFS file system support**

NFS_FS = y

**[*] Provide NFSv3 client support**

NFS_FSv3 = y

**<*> NFS server support**

NFSD = y

**[*] Provide NFSv3 server support**

NFSDv3 = y

En el cliente deben estar presentes las siguientes opciones:

**[*] Networking support**

NET = y

**[*] TCP/IP networking**

INET = y

**[*] IP: kernel level autoconfiguration**

IP_PNP = y

**[*] IP: DHCP support (NEW)**

IP_PNP_DHCP = y

**<*> NFS file system support**

NFS_FS = y

**[*] Provide NFSv3 client support**

NFS_FSv3 = y

**<*> NFS server support**

NFSD = y

**[*] Provide NFSv3 server support**

NFSDv3 = y

**[*] Provide NFS server over TCP support**

NFSD_TCP = y

**[*] Root file system on NFS**

ROOT_NFS = y

**[*] /proc file system support**

PROC_FS = y

**[*] Virtual memory file system support (former shm fs)**

TMPFS = y

## 2.6 PXELINUX

PXELINUX es el cargador de arranque por red utilizado en sustitución a LILO y GRUB y será distribuido mediante el TFTP. Es esencialmente un conjunto mínimo de instrucciones que indica al cliente dónde encontrar su kernel y sistema de archivos inicial y permite varias opciones del kernel.

Este cargador es un archivo binario que se encuentra en el paquete **sys-boot/syslinux**. Se debe instalar el paquete y buscar el archivo en la dirección **/usr/lib/syslinux/pxelinux.0**. Es necesario copiar el archivo para el tftproot. Para configurar el cargador es necesario crear el directorio **/pxelinux.cfg** y dentro de este crear el archivo de configuración **default** que será el archivo de configuración por defecto del cargador de arranque. El binario pxelinux.0 buscará en el directorio pxelinux.cfg un archivo cuyo nombre sea la dirección IP del cliente en hexadecimal. En caso de no encontrarlo quitará el último digito del nombre del archivo y volverá a intentar hasta que se quede sin dígitos. En versiones superiores a la 2.05 de **boot-sys/syslinux** el archivo pxelinux.0 primero realiza una búsqueda intentando encontrar un archivo cuyo nombre sea la dirección MAC del cliente. Si este archivo no es encontrado se efectúa la rutina ya descrita. Y si se queda sin dígitos se usará el archivo **default**. Para configurar el archivo **default** son necesarias las siguientes variables:

- **label** <> – especifica que vienen opciones para esa etiqueta.
- **kernel** <> – define el kernel que será cargado por la red.
- **APPEND** – define las opciones de arranque para el kernel:
    1. **ip=**<> – especifica los parámetros de configuración de red, por omisión se usa dhcp.
    2. **root_type=**nfs – define que tipo de root se cargará en el initramfs.

3. **nfsroot=**<ip:/ruta/imagen> – especifica el ip del servidor y la ruta en el mismo del sistema de archivos que será cargado por la red.
4. **initrd=**<> – define el initramfs a cargar por la red.
5. **nfsparms=**<> – define parámetros del montaje del sistema de archivos, separados por coma. Entre los que se encuentran: hard, nolock, wsize=1024, rsize=1024, posix.
6. **debug** – para el arranque del sistema para realizar algún soporte mediante busybox.
7. **quiet** – no muestra las salidas del kernel en el arranque del sistema.

## 2.7 DHCP (Dynamic Host Configuration Protocol)

*Dynamic Host Configuration Protocol*(Protocolo de Configuración Dinámica de Ordenadores), o DHCP, es un protocolo de red en el que el servidor bajo el que está corriendo provee los parámetros de configuración necesarios a las máquinas conectadas a la red que así lo soliciten. Mediante DHCP se asignarán de forma totalmente automática y transparente parámetros como la puerta de enlace, la máscara de subred, la DNS o la propia dirección IP.

El protocolo admite tres tipos de asignación de direcciones IP, que pueden combinarse entre sí:

- **Manual / Estática** - La asignación se realiza a partir de la lectura de una tabla de direcciones introducida manualmente por el administrador del servidor. Habitualmente, la máquina que recibe la asignación estática tiene igualmente configurada una dirección MAC que no debería repetirse en toda la red. De esta forma, dicha máquina recibe siempre la misma dirección IP, independientemente de dónde y cuándo se realice la conexión.

- **Automática e ilimitada** - Una vez que el administrador ha determinado un rango de direcciones disponibles, la asignación se realiza de forma permanente hacia el cliente que la solicita y hasta que éste la libera.
- **Dinámica y limitada** - Cada cliente obtiene su dirección al iniciar la interfaz de red. Mediante este método, las direcciones dentro del rango elegido por el administrador se reutilizan con cada máquina y durante un tiempo determinado. Con esta asignación se facilita enormemente la entrada de nuevas máquinas a la red de forma dinámica.

Básicamente, DHCP se dividirá en dos partes bien diferenciadas: un protocolo encargado de intercambiar los parámetros de red específicos para cada cliente y un mecanismo encargado de la asignación de las direcciones. Por otra parte, y de forma habitual, el servidor de DHCP se estructurará a partir de dos bases de datos: una estática, al uso de BOOTP, protocolo anterior a DHCP y compatible con éste y otra con una pila de direcciones disponibles, que será la encargada de facilitar los datos en una asignación automática o dinámica.

El funcionamiento sobre el papel de DHCP es bastante simple: el servidor DHCP recibe una petición del cliente y se chequea la base de datos estática en busca de alguna dirección asignada a la máquina que realiza la petición. Si existe una entrada para la dirección física que realiza la petición, se devuelve la dirección almacenada que corresponda. Si no se encuentra nada, el servidor selecciona una dirección disponible de la base de datos dinámica y se asigna de forma temporal a la máquina que lo solicita.

Si el servidor no tiene instalado DHCP es necesario instalarlo. El paquete a instalar es **net-misc/dhcp.** La instalación del paquete deja un fichero de configuración en la ruta **/etc/dhcp.conf**. Este fichero será leído durante la carga del protocolo DHCP y en él se configuran todas las opciones del mismo. Cualquier modificación realizada sobre este fichero será tenida en cuenta cada vez que el demonio de DHCP se inicie.

La configuración del fichero se realiza mediante opciones. A continuación se muestran algunas opciones necesarias para el servidor:

- **default-lease-time** <duración> - Especifica la cantidad de tiempo, en segundos, que será mantenida una asignación de direcciones, siempre y cuando el cliente no haya especificado algo concreto. Ejemplo: *default-lease-time 604800*;
- **max-lease-time** <duración> - Especifica la cantidad máxima de tiempo, en segundos, que será mantenida una asignación de direcciones. No está sujeta a esta especificación la asignación dinámica BOOTP. Ejemplo: *max-lease-time 604800*;
- **allow** client-updates - Permite la actualización de las asignaciones *allow* de un cliente a requerimiento de este, o bien las asignaciones se actualizan cuando el servidor así lo requiera. Ejemplo: *allow booting*;
- **authoritative** - La configuración correcta para la red es la definida en el servidor DHCP. Poner este parámetro al comienzo del archivo de configuración supone que el servidor DHCP reasignará direcciones a los clientes mal configurados por el motivo que sea, incluida una configuración nueva del servidor.
- **use-host-decl-names** <on | off> **-** Usa el alias de la declaración de host como nombre de la máquina en caso de declaración de host. Ejemplo: *use-host-decl-names on*;
- **ddns-update-style** <tipo> - Define el método de actualización automática de las DNS. Los valores pueden ser: ad-hoc, interim y none. Ejemplo: *ddns-update-style none*;
- **get-lease-hostnames** <true | false> - Define si se busca el nombre del nodo al host y lo usa en la opción hostname. Ejemplo: *get-lease-hostnames true*;
- **shared-network** - Declaración de Subred compartida. Ejemplo:

  *shared-network WORKSTATIONS {*

  *#opciones*

  *}*

- **subnet** - Declaración de Subred. Ejemplo:

  *subnet 10.1.1.0 netmask 255.255.255.0 {*

  *#opciones*

*}*

- **option broadcast-address <**IP> - Dirección de difusión. Ejemplo: option *broadcast-address 10.1.1.255*;
- **option routers** <IP, [IP ...]> - Se definen una serie de routers (en la práctica, puertas de enlace), listadas en orden de preferencia, disponibles para el acceso al exterior por parte del cliente. Ejemplo: *option routers 10.1.1.254*;
- **option domain-name** <nombre> - Nombre de dominio que usará el cliente en una resolución de nombres vía DNS. Normalmente, será el nombre de dominio que se añadirá al host que realiza la petición de asignación. Ejemplo: *option domain-name "uci.cu"*;
- **option domain-name-servers** <IP, [IP ...]> - Define el nombre de los servidores DNS. Ejemplo: *option domain-name-servers 10.0.0.3, 10.0.0.4*;
- **range** ip-menor ip-mayor - En una declaración de subred, este parámetro define el rango de direcciones que serán asignadas. Pueden darse dos instrucciones range seguidas del modo:

  range 10.1.1.11 10.1.1.100;

  range 10.1.1.125 10.1.1.210;

- **class** – Declaración de la clase. Ejemplo:

  *class "nova" {*

  *#opciones*

  *}*

- **vendor-option-space** PXE - Al menos 1 de las opciones PXE específicas del fabricante deben ser definidas para que el ROM de arranque de los clientes se de cuenta que el servidor tiene las características PXE.
- **next-server** IP - IP del servidor TFTP. Ejemplo: *next-server 10.1.1.12*;

- **filename** "dirección" – Ruta de la imagen a cargar dentro del tftproot. Ejemplo: *filename "nova/pxelinux.0"*;
- **option root-path** "dirección del tftproot" – Define la ruta completa del servidor TFTP (IP y tftproot). Ejemplo: option root-path "10.1.1.12:/diskless/nova/" **[10]**

## 2.8 TFTP (Trivial File Transfer Protocol)

El Trivial File Transfer Protocol (Protocolo de transferencia de archivo trivial), o TFTP, es un protocolo de transferencia muy simple con funcionalidades parecidas a las de una versión básica de FTP. TFTP puede ser utilizado para transferir pequeños archivos entre ordenadores en una red, en este caso cuando una terminal ligera arranca desde un servidor de red.

Ya que TFTP utiliza UDP, no hay una definición formal de sesión, cliente y servidor. Sin embargo, cada archivo transferido vía TFTP constituye un intercambio independiente de paquetes, y existe una relación cliente-servidor informal entre la máquina que inicia la comunicación y la que responde.

La máquina X, que da inicio a la comunicación, envía un paquete RRQ (read request/petición de lectura) o WRQ (write request/petición de escritura) a la máquina Y, la que recibe información, que contiene el nombre del archivo y el modo de transferencia.

Y responde con un paquete ACK (acknowledgement/confirmación), que también sirve para informar a X del puerto de la máquina Y al que tendrá que enviar los paquetes restantes.

La máquina origen envía paquetes de datos numerados a la máquina destino, todos excepto el último conteniendo 512 bytes de datos. La máquina destino responde con paquetes ACK numerados para todos los paquetes de datos.

El paquete de datos final debe contener menos de 512 bytes de datos para indicar que es el último. Si el tamaño del archivo transferido es un múltiplo exacto de 512 bytes, el origen envía un paquete final que contiene 0 bytes de datos.

El servidor TFTP les va a dar a los clientes el kernel y un sistema de archivo inicial. Todos los kernels y sistemas de archivos de las imágenes clientes deben estar almacenados en el servidor TFTP.

Si el servidor no tiene instalado TFTP es necesario instalarlo. Un servidor TFTP altamente recomendado está disponible en el paquete **net-ftp/tftp-hpa,** este servidor ha sido escrito por el autor de SYSLINUX y funciona muy bien con pxelinux**.** La instalación del paquete deja un fichero de configuración en la ruta **/etc/conf.d/in.tftp**. para configurar este archivo son necesarias las siguientes variables:

- **INTFTPD_PATH** - Especifica la ruta al directorio raíz o tftproot.
- **INTFTPD_OPTS** – Opciones de la línea de comandos:
  1. La opción **-l** indica que este servidor escucha en modo *stand alone* así que no necesita ejecutar inetd.
  2. La opción **-v** indica que los mensajes de registro/error deben ser mostrados.
  3. La opción **-s ${INTFTPD_PATH}** especifica el directorio raíz de tu servidor tftp. **[11]**

## 2.9 NFS (Network File System)

El *Network File System* (Sistema de archivos de red), o NFS, es un protocolo de nivel de aplicación según el Modelo de OSI (Interconexión de Sistemas Abiertos u *Open System Interconnection*). Es utilizado para sistema de archivos distribuido en un entorno de red de computadoras de área local. Posibilita que distintos sistemas conectados a una misma red accedan a ficheros remotos como si se tratara de locales.

Características:

- El sistema NFS está dividido al menos en dos partes principales: un servidor y uno o más clientes. Los clientes acceden de forma remota a los datos que se encuentran almacenados en el servidor.

- Las estaciones de trabajo locales utilizan menos espacio de disco duro debido a que los datos se encuentran centralizados en un único lugar pero pueden ser accedidos y modificados por varios usuarios, de tal forma que no es necesario replicar la información.
- Los usuarios no necesitan disponer de un directorio "home" en cada uno de las máquinas de la red local, sino que pueden crearse en el servidor para posteriormente poder acceder a ellos desde cualquier máquina a través de la infraestructura de red.
- También se puede compartir a través de la red dispositivos de almacenamiento lo que puede reducir la inversión y mejorar el aprovechamiento de hardware en una red local.

El protocolo NFS es el primer objetivo a la hora de configurar un servidor para terminales ligeras, ya que el mismo es el que hace posible que el sistema de archivos pueda ser visto desde todas las máquinas que integran la red.
Para que este protocolo funcione el sistema operativo debe tener las siguientes características:
1. Instalando el paquete **net-fs/nfs-utils**.
Es necesario verificar la instalación del paquete **nfs-utils** ya que contiene las herramientas a nivel de usuario de servidor y cliente para utilizar las características NFS del kernel.

2. Configurando el archivo **/etc/exports**.

El archivo **/etc/exports** define como, a quién y qué se debe exportar a través de NFS. El primer campo indica el directorio que será exportado y el siguiente indica a quién y cómo. Este campo puede dividirse en dos partes: a quién se le debe permitir montar ese directorio en particular, y qué puede hacer el cliente con él. Aquí se muestran algunas de las opciones que pueden elegirse:

- ***ro*** | ***rw*** Con la opción *ro* el directorio será compartido en modo lectura. Esta opción está por defecto. Con la opción *rw* se permitirá tanto acceso de lectura como de escritura.
- ***sync*** | ***async*** *sync* es la opción recomendada, ya que se ha de respetar el protocolo NFS, es decir, no se responden a las peticiones antes de que los cambios realizados

sean escritos al disco. Con la opción *async* se permite mejorar el rendimiento y agilizar el funcionamiento global, pero supone un riesgo de corrupción de archivos o del sistema de ficheros en casos de caídas del servidor y/o errores de éste.

- ***root_squash*** | ***no_root_squash*** | ***no_all_squash*** | ***all_squash*** *root_squash* indica que un cliente identificado como root tendrá acceso al directorio con privilegios de un usuario anónimo. Si seleccionamos la opción *no_root_squash* evitaremos esto, y si indicamos *all_squash*, entonces aplicaremos esto último a todos los usuarios, no sólo root.

  Ejemplo:

  /diskless/nova 192.168.1.21(async, rw, no_root_squash, no_all_squash)

  En el ejemplo anterior se exporta desde el servidor a una máquina con permisos de escritura. Si se quisiera exportar a las restantes máquinas de la red sin permisos de escritura sería de la siguiente forma:

  /diskless/nova 192.168.1.0/24 (sync, ro, no_root_squash, no_all_squash) **[12]**

## 2.10 Initramfs

### *2.10.1 Initramfs en comparación con initrd*

En comparación, initramfs es un sistema más conveniente y simple para gestionar por los administradores que los sistemas previos basados en discos RAM, ya que el código externo alojado en el disco RAM inicial puede ser editado fácilmente sin privilegios de administrador, y ya que hay una menor indirección: no hay necesidad de hacer un disco virtual, formatearlo y proveer al kernel con capacidad para manejar sistemas de archivos más allá de los requisitos mínimos para leer un archivo cpio comprimido.

Desde que el sistema original basado en discos RAM se popularizó, un nuevo sistema de archivos basado en memoria RAM más flexible, conocido como tmpfs o shmfs, se ha

convertido en un componente estándar del kernel. Este sistema es mucho más flexible y eficiente que el disco RAM de tamaño fijo original en muchos aspectos: no requiere formateo, y utiliza tanta memoria como se necesite para contener los datos.

También se usaba otro sistema similar, ramfs, que da al sistema initramfs su nombre. Actualmente los usuarios pueden elegir qué sistema de archivos dinámico en RAM utilizar.

Además del código para tmpfs, que se utiliza en el kernel para casi todas las configuraciones de Linux, el único requisito para hacer un disco virtual desde el que arrancar era añadir la posibilidad de descomprimir un paquete de archivos (los desarrolladores del kernel escogieron utilizar el formato de archivo cpio). Los archivos descomprimidos se almacenan en un sistema de archivos parecido a tmpfs. Este sistema se conoce como initramfs.

### *2.10.2 Implementación*

Pasando a sus características físicas, el kernel y el initramfs deben estar en lugares accesibles de forma sencilla por el firmware de arranque de la computadora o el cargador de arranque por software de Linux (grub).

El gestor de arranque es el encargado de cargar el kernel en memoria e iniciarlo, pasándole la dirección de memoria del initramfs. Cuando termina de cargarse el kernel, este trata de determinar el tipo de sistema de ficheros que usa el initramfs y leerlo. Si la imagen de initramfs es un comprimido gzip cpio, será descomprimido por el kernel para un espacio temporal de tipo tmpfs (en memoria) para luego convertirse en el sisma de ficheros root inicial, mini-sistema ó miniroot.

Ahora, si la imagen contiene otro sistema de ficheros, entonces se usa otra técnica que es la de usar el dispositivo “/dev/ram” (debe esta compilado estáticamente en el kernel).

Este dispositivo de bloque es montado como sistema de ficheros de root inicial. Muchas distribuciones usan el ext2 comprimido como imágenes de initrd por su relativa sencillez, otros

usan imágenes cramfs que son capaces de no usar casi espacio en memoria ya que no requiere ser descomprimido para su uso.

Una vez que es cargado el root inicial, el kernel ejecuta “/init” ó "/linuxrc" como primer proceso. Cuando este termina, el kernel asume que el verdadero root esta en “/sbin/init” para poder comenzar con la ejecución del mismo. El objetivo del initramfs es iniciar ó como se le dice ordinariamente, "bootear" al sistema root real. Para lograr esto, el initramfs realiza algunos preparativos previos y monta el sistema de ficheros root real, es decir, con él se podrá acceder al sistema de ficheros root real donde sea que este se encuentre ubicado. Esta característica le aporta mucha flexibilidad a los sistemas operativos ya que estos pueden tomar vida en cualquier ambiente de hardware.

El “/linuxrc” no puede por si solo montar el “/”(root) verdadero, ya que eso dejaría los scripts y las herramientas en el sistema de archivo raíz inicial inaccesible para cualquier tarea de limpieza final. En su lugar es montado en un punto temporal y cambiado a su lugar con pivot_root (que fue introducido específicamente para este propósito). Esto deja el sistema raíz inicial en un punto de montaje (como "/initrd") donde los scripts de inicio normal pueden posteriormente desmontarlo para liberar memoria ocupada por el initramfs. Además existe otra herramienta llamada chroot ó switch_root (Busybox) que es la que cambia el entorno de miniroot a root real. Estos trabajos llevan consigo que luego de haber hecho el cambio de root, se deberá liberar aquella memoria que se usó para el root inicial.

Muchas distribuciones implementan el “/linuxrc” ó “/init” como un script de shell (Bash), con un shell mínimo, generalmente para esto se usa Busybox.

La calidad o fortaleza del initramfs depende en cierto grado de la cantidad de módulos (drivers en inglés) que posea en su interior, para lograr una mejor detección o localización del root real (real root en inglés). El mismo carga los drivers necesarios para que el kernel pueda detectar los discos duros ide, scsi, sata, firewire, usb, cdrom, un servidor NFS por la red, sistemas RAID, LVM y hasta root comprimidos ó encriptados. También tienen como objetivo reducir el tamaño del kernel Linux ya que existen algunos tipos de hardware con memoria limitada, y así seria posible cargar solamente los módulos necesarios. Esto último trae como

desventaja que entonces hay que detectar el hardware dinámicamente o por deducción para localizar el sistema de ficheros root real.

Con versiones anteriores al kernel 2.4 esto no era posible ya que el mismo hallaba el root real a través de opciones especiales de arranque. Estas eran frecuentemente frágiles, inflexibles y duplicaban la funcionalidad existente en utilidades disponibles en el espacio del usuario. Pero ya con la aparición del kernel 2.6 el esquema de initramfs resolvía este problema introduciendo una etapa extra en el proceso de arranque del kernel.

### *2.10.3 Lenguajes*

Los lenguajes de programación más usados en un initramfs:

1. Bash es un shell de Unix (intérprete de órdenes de Unix) escrito para el proyecto GNU. Su nombre es un acrónimo de bourne-again shell (otro shell bourne) — haciendo un juego de palabras (born-again significa renacimiento) sobre el Bourne shell (sh), que fue uno de los primeros intérpretes importantes de Unix. Los programadores generalmente optan por desarrollar scripts de bash en lugar de aplicaciones compiladas en el espacio de usuario ya que en muchas ocasiones resulta mucho más práctico el mantenimiento del código.
2. C es un lenguaje de programación creado en 1972 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL. Al igual que B, es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

   Para mantener la compatibilidad con las bibliotecas de C los desarrolladores prefieren que las aplicaciones que se usan en un initramfs estén implementadas en C.

Convenientemente el desarrollador puede prescindir del Bash, en dependencia del objetivo que se persiga.

### *2.10.4 Herramientas*

Existen pocas herramientas para la construcción automática de un initramfs, entre ellas están el mkinitrd y genkernel. Esta última nativa de Gentoo esta diseñada para compilar un kernel y posteriormente crea un initramfs usando scripts de bash.

Para los principiantes, la mejor forma de crear un initramfs con las características que se deseen es haciéndolo a "mano", es decir, sin ninguna herramienta que lo genere automáticamente. La lista de aplicaciones necesarias para crear un initramfs son:

- interprete de bash
    - El intérprete de bash es necesario para automatizar algunas tareas como la copia de los módulos, creación de dispositivos y directorios.
- consola de bash
    - Sirve para hacer operaciones manuales que no necesitan de la automatización.
- mkdnod
    - Se usa para crear dispositivos imprescindibles en el initramfs tales como:
        - /dev/null, /dev/console, /dev/tty1

Advertencia: Cuando se crea un initramfs, es necesario que esté presente al menos el dispositivo "/dev/console", ya que mediante este es que se pueden imprimir los mensajes de la salida estándar.

- mkdir
    - Se usa para crear los directorios.
- cp

- Destinado a copiar ficheros.

- cpio
  - Empaquetador de fichero como el tar.
- gzip
  - Compresor de ficheros.
- cat
  - Lector de ficheros, imprime el contenido de un fichero en la salida estándar.
- grep
  - Se usa para hallar expresiones regulares en un fichero.
- sed
  - Editor de texto no interactivo, se usa para cambiar expresiones regulares dentro de un texto o flujo de texto. La lista de programas que necesita el initramfs internamente es la que sigue:
  - busybox
    - Busybox es un programa poco conocido ya que su principal objetivo esta dedicado a los sistemas embebidos y el initramfs pudiera considerarse uno. En su interior posee la mayoría de los comandos UNIX.
    - Existen cuatro formas más comunes de ser ejecutado:

      1. Mediante accesos directos
         a. “ln -sf /bin/busybox /bin/cp”, este comando crea un acceso directo en el directorio /bin para el comando “cp” que funciona igual que el “cp” estándar.
      2. Mediante llamadas directas
         a. “/bin/busybox cp” es similar al comando “cp”
      3. Mediante alias.

a. “alias cp='/bin/busybox cp'” se crea un alias para “cp” y luego es posible ejecutar a “cp” de forma tradicional.

4. Mediante scripts.

a. Se crea un script con el nombre del comando y que en su interior contenga el comando deseado directamente, ejemplo: “/bin/busybox cp”

- Además, busybox puede ser compilado solamente con los comandos deseados.
- En todos los casos, busybox no excede los 2 MB de tamaño.

- alternativos a busybox
  - Como alternativa a busybox se pueden tomar los binarios y bibliotecas necesarios de su sistema operativo.
    - El comando ldd se utiliza para saber que bibliotecas necesita cada binario.
  - Otras alternativas de la propia inspiración.

### *2.10.5 Bibliotecas*

En todo tipo de sistemas UNIX tiene un papel primordial el uso de bibliotecas compartidas (“shared libraries” en inglés) ya que disminuye drásticamente el tamaño de los binarios ejecutables, en el initramfs se utilizan este tipo de bibliotecas con el objetivo de lograr que quede lo más pequeño posible.

- Glibc es la biblioteca estándar de C GNU. En los sistemas que la usan, esta proporciona y define las llamadas al sistema y otras funciones básicas, es utilizada por casi todos los programas. Es muy usada en los sistemas GNU y sistemas basados en el kernel Linux. Es muy portable y soporta gran cantidad de plataformas de hardware. En los sistemas Linux se instala con el nombre de libc.so.6.

- UClibc es una biblioteca estandar de C diseñada para sistemas Linux embebidos y es mucho más pequeña que glibc.
- Dietlibc comparte la misma idea de uClibc pero esta diseñada para crear los binarios lo más pequeños posible.
- Newlib mantenida por los desarrolladores de Red-Hat es similar a uClibc y dietlibc. Advertencia: Cuando se compila busybox con enlaces estáticos a bibliotecas, provoca comportamientos erráticos y corruptos en sus comandos.

La opción más segura es que se compilen los binarios con enlaces dinámicos para evitar desestabilizar el sistema en una fase tan crucial.

Para usar alguna de estas bibliotecas en la creación de un initramfs el usuario puede basarse en los siguientes criterios:

1. Glibc:
    - Permite la creación rápida y fácil pero tiene como desventaja que se consume más espacio en memoria debido al tamaño de los archivos ejecutables.
2. uClibc, newlib, dietlibc:
    - Es más complejo a la hora de compilar los binarios con estas bibliotecas, pero los sus tamaños resultan considerablemente más pequeños permitiendo añadir mayor cantidad de archivos ejecutables en el initramfs.

### *2.10.6 Usos*

La invención del initramfs ha revolucionado el mundo de Linux haciendo posible que se pueda iniciar un sistema operativo basado en Linux desde cualquier medio informático de almacenamiento.

*LiveCD ó LiveDVD*

Uno de los ejemplos más populares es el LiveCD ó LiveDVD. Estos generalmente poseen una imagen comprimida del sistema operativo en su interior y el initramfs debe ser capaz de detectar primero el CD ó DVD, luego gestionar el contenido de la imagen comprimida, unir los directorios del sistema con la memoria RAM para simular una escritura en el sistema y por último iniciarlo.

*LiveUSB*

Los más conocidos por LiveUSB funcionan desde una memoria flash, una MP3, MP4 o disco duro externo con conexión por USB. Al igual que los CD y DVD, el LiveUSB contiene básicamente una imagen comprimida del sistema operativo y el proceso de arranque con el initramfs es muy similar a estos, con la peculiaridad de que los datos pueden guardarse de forma persistente.

*Terminales Ligeras*

Uno de los casos más interesantes del uso del initramfs es el de hacer posible que una computadora que en ausencia de cualquier medio de almacenamiento de datos persistentes pueda iniciar un sistema operativo mediante la red.

Básicamente el kernel Linux y el initramfs son cargados desde la red mediante el BIOS, y luego de ejecutados, el initramfs es el encargado de preparar las todas las conexiones necesarias, gestionar la imagen que se encuentra compartida en la red, combinarla con la memoria RAM y por último inicia el sistema real.

*Otros*

En los últimos tiempos, el volumen de información que se maneja cada vez es mayor y más delicado, por lo que ciertas entidades o particulares han tenido que recurrir al cifrado de los datos por causa de robo o manipulación de los mismos. Una forma muy segura de mantener los datos en secreto o protegidos es en volúmenes o particiones encriptadas. Si el sistema operativo está en alguno de estos volúmenes, pues ni el BIOS, ni el gestor de arranque ni el kernel mismo podrán accederle, por lo que aquí de nuevo entra en acción el initramfs, ya que este permite ejecutar ciertos programas en el espacio de usuario que son capaces de

gestionar este tipo de volúmenes y descifrarlos preguntando una clave para posteriormente iniciar el sistema en cuestión.

Ejemplos menos comunes son los de iniciar un sistema desde arreglos de particiones de tipo LVM y RAID comúnmente usadas en servidores. Incluso un initramfs puede servir como sistema operativo por sí mismo, con la desventaja de que solo se aloja en la memoria RAM, y una vez que se reinicie el ordenador, todos los cambios se habrán perdido.

Algunas distribuciones como es el caso de Debian, usan el initramfs como sistema operativo para efectuar la instalación del mismo. **[9]**

## 2.11 Ventajas de las Terminales Ligeras

**A grandes rasgos las ventajas que aporta el uso de las terminales ligeras:**

- Actualización: cuando surja la necesidad de actualizar el software o el hardware, será necesario enfrentar estos cambios solamente en el servidor.
- Comodidad: el *login* de usuario es independiente de las terminales, dando la posibilidad de registrarse en distintas terminales ubicadas geográficamente dispersas, manteniendo el escritorio, sus preferencias y "permisos".
- Costos: se recuperan equipos obsoletos. No hace falta que las terminales lleven disco rígido, disqueteras o lectoras de CD. Además, al ser una alternativa libre y gratuita, el costo de licencias de software es cero.
- Mantenimiento: como sólo se debe administrar el servidor, es posible reducir el mantenimiento, en comparación con el esquema de nodos aislados tradicional. En una red que contiene 25 nodos, de la forma tradicional hay que estar actualizando el software de cada máquina, en cambio haciendo uso de la tecnología de clientes ágiles, sólo es necesario actualizar una sola, lo que conlleva a un importante ahorro de tiempo y dinero.
- Backup: como toda la información reside en el servidor es más fácil y necesaria la tarea de hacer copias de seguridad.

- Seguridad: únicamente se debe ejercer control sobre una máquina. Definiendo los tipos de usuario y grupos a los que pertenece se evita el borrado (accidental y/o intencional) de programas, datos y configuraciones.
- Ampliación: si queremos agregar una nueva terminal, solo hay que "enchufar" el hardware y en minutos tendremos una terminal totalmente operable con software instalado, configurado, actualizado y probado.
- Robustez (Hardware): Como las terminales pueden no poseer disco rígido, la posibilidad de su rotura es cero.
- Ecología: los clientes consumen menos energía y son más silenciosos. Al no necesitar disco rígido, lectoras de CD y casi ningún periférico, se obtienen terminales más livianos, más pequeños y que consumen menos corriente eléctrica. **[13]**

## 2.12 Requerimientos

Supongamos que nuestro servidor es un Pentium IV de 2 Ghz con 512MB de RAM, y que nuestra red es de 100Mbits conmutada por un switch. La máquina que consideramos ideal como cliente es un Pentium 166Mhz. con 32Mb. de RAM y placa de video de 1MB. El rendimiento es óptimo, las ventanas abren al instante, no existen delays y es virtualmente imposible distinguir si estamos ante una PC obsoleta sin disco rígido o delante del servidor. Es válido aclarar que se han anunciado resultados similares (en cuanto a rendimiento) con equipos 486DX4 y 16MB de RAM. En instituciones como colegios, oficinas, ministerios, etc., esta tecnología no tiene rival en cuanto a mantenimiento y conservación, ahorros energéticos, estructurales y de despliegue. La separación que se puede realizar entre centro de calculo/datos y punto de acceso a la información (Desktop) la hacen muy adecuada para lugares cuyas condiciones ambientales sean "agresivas".

## 2.13 Conclusiones parciales

Después de un estudio del funcionamiento de las terminales ligeras, al usar el initramfs de la distribución base de Nova, Gentoo, surge un problema al iniciar el sistema, pues el initramfs de Gentoo plantea algunas opciones específicas sin posible variación. Debido a esto se realizaron cambios en dicho initramfs para lograr montar el sistema de archivos con permisos de escritura y otras opciones previamente definidas.

# Capítulo 3: Descripción de la Solución

## 3.1 Introducción

En este capítulo se define la arquitectura del servidor para terminales ligeras propuesto. Además provee una guía de como montar un servidor para terminales ligeras usando la distribución Nova.

## 3.2 Arquitectura

La arquitectura del servidor de terminales ligeras que se propone es el resultado de la unión entre la arquitectura de la distribución Nova y paquetes de aplicaciones que respondan al uso de DHCP, TFTP, NFS y PXE que no es más que la tecnología con la que se podrá arrancar diferentes distribuciones de GNU/Linux en el cliente sin necesidad de disco de almacenamiento, ya que este arranque se hace mediante la red. PXE hace referencia al entorno de ejecución de prearranque (*Preboot eXecution Environment*) es un entorno para arrancar e instalar el sistema operativo en computadores desde una red. PXE está relacionado con varios protocolos de red como los antes mencionados, IP, UDP entre otros.

## 3.3 Configuración de la imagen servidora

Se parte de una imagen base de la distribución Nova, preferiblemente sin entorno gráfico así es más pequeña y permite más procesamiento al tener menos servicios ejecutándose.
Primero se instalarán los softwares requeridos:

```
# net-misc/dhcp
# net-fs/nfs-utils
# net-ftp/tftp-hpa
# sys-boot/syslinux
```

Para hacerlo se abrirá una terminal y como root se pondrá el siguiente código:

```
# equo install net-misc/dhcp net-fs/nfs-utils net-ftp/tftp-hpa
  sys-boot/syslinux
```

Este código instalaría los binarios predeterminados de Nova en caso de necesitar algún soporte que no esté incluido se usaría *emerge*. De la siguiente forma:

```
# emerge net-misc/dhcp net-fs/nfs-utils net-ftp/tftp-hpa sys-
  boot/syslinux
```

Una vez instalados los softwares se compila el kernel con el soporte necesario. Para esto se necesitará el software *sys-kernel/genkernel* y algún fuente de kernel en el software *sys-kernel/gentoo-sources* en caso de que no estén instalados:

Usando *equo*.

```
# equo install sys-kernel/genkernel sys-kernel/gentoo-sources
```

Usando *emerge*.

```
# emerge sys-kernel/genkernel sys-kernel/gentoo-sources
```

Se ejecuta *genkernel* con la opción *--menuconfig* que permite configurar el kernel mediante un menú.

```
# genkernel --menuconfig all
```

Al ejecutar el comando anterior sale una pantalla con el menú de *genkernel*. Las opciones para el servidor son:

```
Device Drivers --->
   [*] Networking support --->
         Networking options --->
               [*] TCP/IP networking
               [*]   IP: multicasting
```

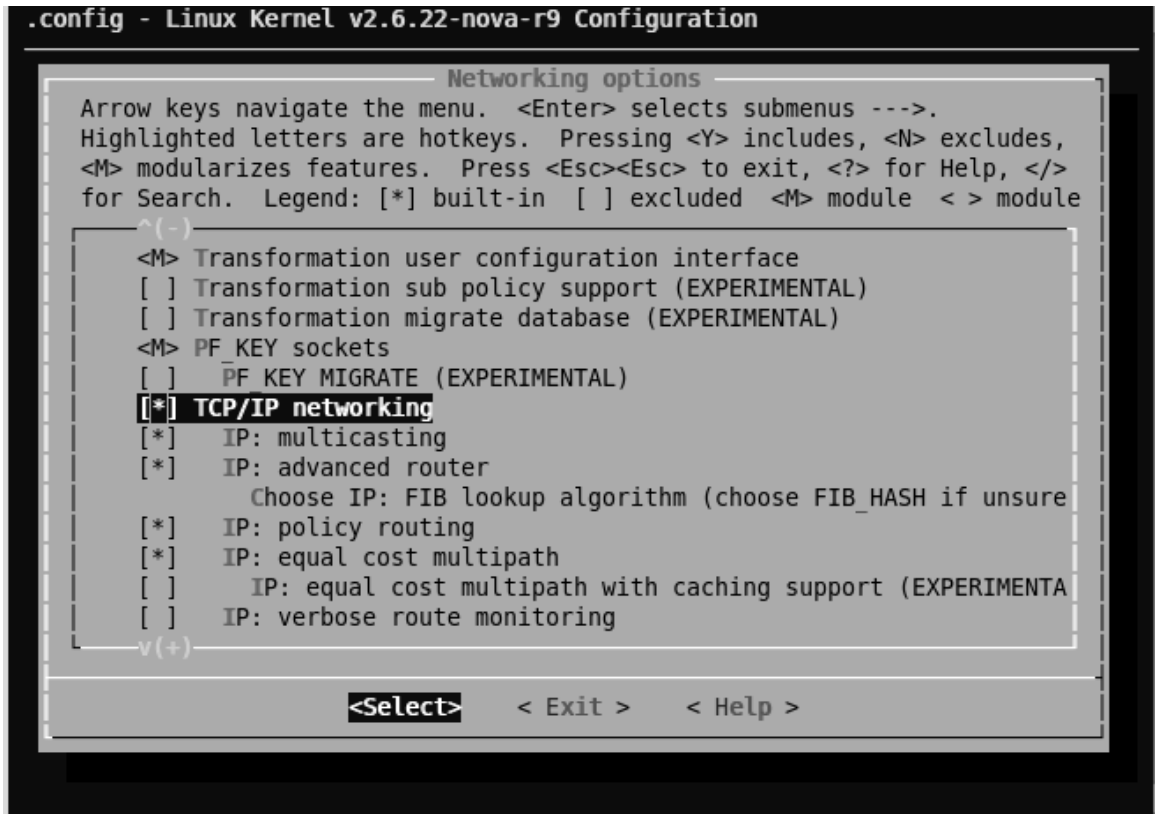



Figura 1.3 Configuración de kernel

```
File systems --->
  Network File Systems --->
      <*> NFS file system support
       [*]     Provide NFSv3 client support
      <*> NFS server support
       [*]     Provide NFSv3 server support
       [*]  Provide NFS server over TCP support
```

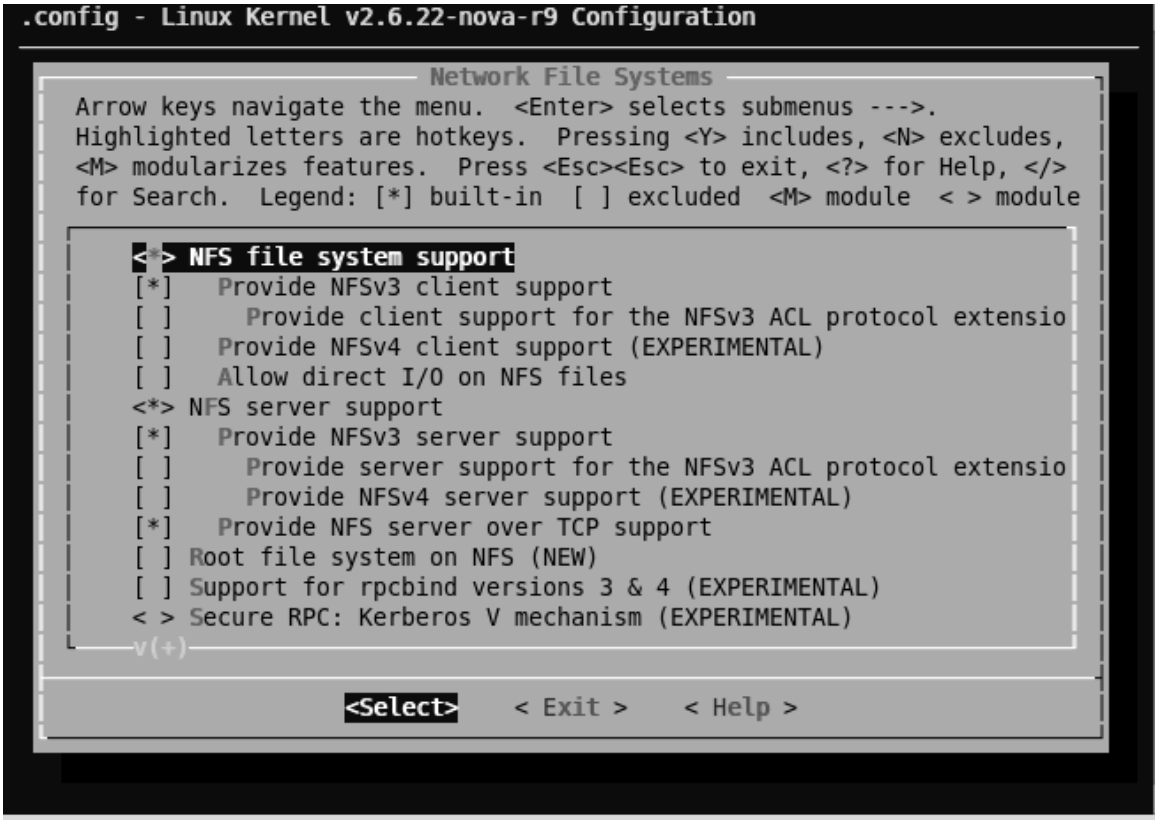



Figura 1.4 Configuración de kernel

Se debe tener en cuenta que estas opciones de compilación del kernel deben ser añadidas a las opciones específicas para el sistema del servidor. Aquí solo se explican las opciones necesarias para un servidor de terminales ligeras.

Al terminar de configurar y compilar el kernel ya se puede configurar DHCP. Para esto se edita el archivo */etc/dhcp/dhcpd.conf* con un editor de texto. En este caso se utilizará el nano.

```
# nano /etc/dhcp/dhcpd.conf
```

Aquí se utilizará la subred, rango, directorios y el IP empleada para la configuración del servidor. El archivo debe quedar de la siguiente forma:

```
default-lease-time 604800;
max-lease-time     604800;
allow booting;
allow bootp;
```

```
authoritative;
use-host-decl-names on;
ddns-update-style none;
get-lease-hostnames true;
shared-network WORKSTATIONS {
     subnet 10.12.167.0 netmask 255.255.255.0 {
          option broadcast-address 10.12.167.255;
          option routers 10.12.167.254;
          option domain-name "uci.cu";
          option domain-name-servers 10.0.0.3, 10.0.0.4;
          range 10.12.167.100 10.12.167.252;
          class "pxeclient" {
               match if substring(option vendor-class-identifier, 0,
9) = "PXEClient";
               vendor-option-space PXE;
               next-server 10.12.167.230;
               filename "nova/pxelinux.0";
               option root-path "10.12.167.230:/diskless/baire/";
          }
     }
}
```

Luego de configurado deberá iniciar el demonio de DHCP y añadirlo al inicio automático en el arranque del servidor. Esto se logra de la siguiente forma:

Iniciando el demonio

```
# /etc/init.d/dhcpd start
```

Añadiendo al inicio automático

```
# rc-update add dhcpd default
```

En caso de hacerse algún cambio en la configuración del archivo después de iniciado el demonio, se deberá reiniciar mediante el siguiente comando:

```
# /etc/init.d/dhcpd restart
```

Ahora deberá crearse un directorio raíz para el TFTP. En este caso se encontrará en */diskless/*. Para crear el directorio se utilizará el comando *mkdir*. Este comando seguido de la opción -p permite crear un camino de directorios entrados seguidos después del mismo. Quedaría de la siguiente forma.

```
# mkdir -p /diskless/
```

Una vez establecido el directorio raíz deberá configurarse TFTP. Para esto se necesitará editar el archivo */etc/conf.d/in.tftpd*. Debe utilizarse un editor de textos.

```
# nano /etc/conf.d/in.tftpd
```

El archivo se edita para especificar donde estará la raíz del TFTP. Debe mostrarse lo siguiente:

```
#/etc/init.d/in.tftpd
#Path to server files from
#Depending on your application you may have to change this.
#This is commented out to force you to look at the file!
#INTFTPD_PATH="/var/tftp/"
#INTFTPD_PATH="/tftpboot/"
INTFTPD_PATH="/diskless/"
#For more options, see in.tftpd(8)
#-R 4096:32767 solves problems with ARC firmware, and obsoletes
#the /proc/sys/net/ipv4/ip_local_port_range hack.
#-s causes $INTFTPD_PATH to be the root of the TFTP tree.
#-l is passed by the init script in addition to these options.
INTFTPD_OPTS="-R 4096:32767 -s ${INTFTPD_PATH}"
```

En este caso el # es un comentario del archivo de configuración. Esto significa que no serán tomados en cuenta a la hora de iniciar el demonio. Es igual para todos los archivos de configuración en esta guía.

Al terminar la configuración se inicia el demonio de TFTP y se añade al inicio automático en el arranque del servidor. Debe seguir los pasos que se describen a continuación:

Iniciando el demonio

```
# /etc/init.d/in.tftpd start
```

Añadiendo al inicio automático

```
# rc-update add in.tftpd default
```

En caso de hacerse algún cambio en la configuración del archivo después de iniciado el demonio, se deberá reiniciar mediante el siguiente comando:

```
# /etc/init.d/in.tftpd restart
```

Ahora se necesita configurar NFS. Para esto se creará el directorio de la imagen a compartir por la red para que sea utilizada por los clientes. Se crea el directorio usando *mkdir*, como ya se ha explicado, en este caso no es necesario la opción -p ya que estará en */diskless/baire* y el directorio */diskless* existe.

```
# mkdir /diskless/baire
```

Una vez creado el directorio se procede a compartirlo. Se hace necesario editar para ello el archivo de configuración */etc/exports*. Se usa un editor de texto como en ejemplos anteriores:

```
# nano /etc/exports
```

Dentro de este se editará para que quede de la siguiente forma:

```
# /etc/exports: NFS file systems being exported.  See exports(5).
/diskless/baire 10.12.167.100(rw,sync,no_root_squash,no_subtree_check)
/diskless/baire
10.12.167.0/24(ro,sync,no_root_squash,no_subtree_check)
```

Después solo quedaría iniciar el demonio de NFS y añadirlo al inicio automático en el arranque del servidor. Se logra de la siguiente forma:

Iniciando el demonio

```
# /etc/init.d/nfs start
```

Añadiendo al inicio automático

```
# rc-update add nfs default
```

En caso de hacerse algún cambio en la configuración del archivo después de iniciado el demonio, se deberá reiniciar mediante el siguiente comando:

```
# /etc/init.d/nfs restart
```

Ahora es necesario copiar el archivo *pxelinux.0* para el directorio creado en el TFTP para el kernel y los archivos de arranque de la imagen especificados en la configuración de DHCP. En

esta guía se utilizará */diskless/nova/*.

Primero se crea el directorio donde se copiará el archivo:

```
# mkdir /diskless/nova
```

Luego se copia el archivo que está en la dirección */usr/lib/syslinux/pxelinux.0*:

```
# cp /usr/lib/syslinux/pxelinux.0 /diskless/nova/
```

El archivo *pxelinux.0* precisa de un directorio *pxelinux.cfg/* donde buscará la configuración de los clientes a levantar. En esta guía se utilizará el archivo *default* para dicha configuración.

Se crea el directorio *pxelinux.cfg*:

```
# mkdir /diskless/nova/pxelinux.cfg
```

Luego se crea el archivo *default* dentro del mismo:

```
# touch /diskless/nova/pxelinux.cfg/default
```

Ahora se edita el archivo con un editor de texto:

```
# nano /diskless/nova/pxelinux.cfg/default
```

El archivo debe tener el kernel, initramfs, dirección del servidor con la imagen compartida y las opciones para el arranque del sistema entre otras cosas. Debe quedar de la siguiente forma:

```
prompt=0
label linux
  kernel kernel
  APPEND ip=dhcp nfsroot=10.12.167.230:/diskless/baire root_type=nfs
initrd=initramfs.cpio.gz                                    vga=791
nfsparms=ro,hard,nolock,wsize=1024,rsize=1024,posix quiet
```

Una vez realizadas las configuraciones anteriores el servidor ya está listo para copiar la imagen que usarán los clientes.

### 3.4 Configuración de la imagen cliente

Antes de configurar la imagen cliente se hizo necesaria la implementación de un nuevo uso al initramfs para Nova, ver anexo 3. La imagen a utilizar será una imagen de nova con entorno gráfico y debe copiarse para el directorio que esté indicado en el archivo */etc/exports*

previamente configurado. En esta guía se utilizará */diskless/baire*. Para copiarla se utilizará el siguiente comando:

```
# cp -r /direccion/de/la/imagen /diskless/baire/
```

Si se copia la imagen compactada habría que descompactarla usando el siguiente comando:
En caso de que sea bzip2

```
# tar -xvjp nombredelaimagen.tar.bz2
```

En caso de que sea gzip

```
# tar -xvpf nombredelaimagen.tar.gz
```

Para compilar el kernel de la imagen existen 2 vías:

1. Compilar en el sistema del servidor y luego copiar el kernel con su initrd en el TFTP y los módulos en la imagen.
2. Hacer un chroot al directorio de la imagen compilarlo dentro de la misma y luego copiar el kernel con su initrd en el TFTP.

Para realizar esta guía se usó la primera vía. Se deben seguir los pasos descritos a continuación:
Realizar un backup de su kernel y sus módulos.

```
# mkdir /backup
# cp /boot/* /backup/
# cp -r /lib/modules/* /backup/
```

Compilar el kernel usando *genkernel.*

```
# genkernel --menuconfig all
```

Al ejecutarse ese comando saldrá en la pantalla el menú del kernel. Ahí se especificarán las siguientes opciones:

```
Device Drivers --->
    [*] Networking support --->
          Networking options --->
               [*] TCP/IP networking
               [*]   IP: multicasting
               [*] IP: kernel level autoconfiguration
```

```
[*]   IP: DHCP support
```

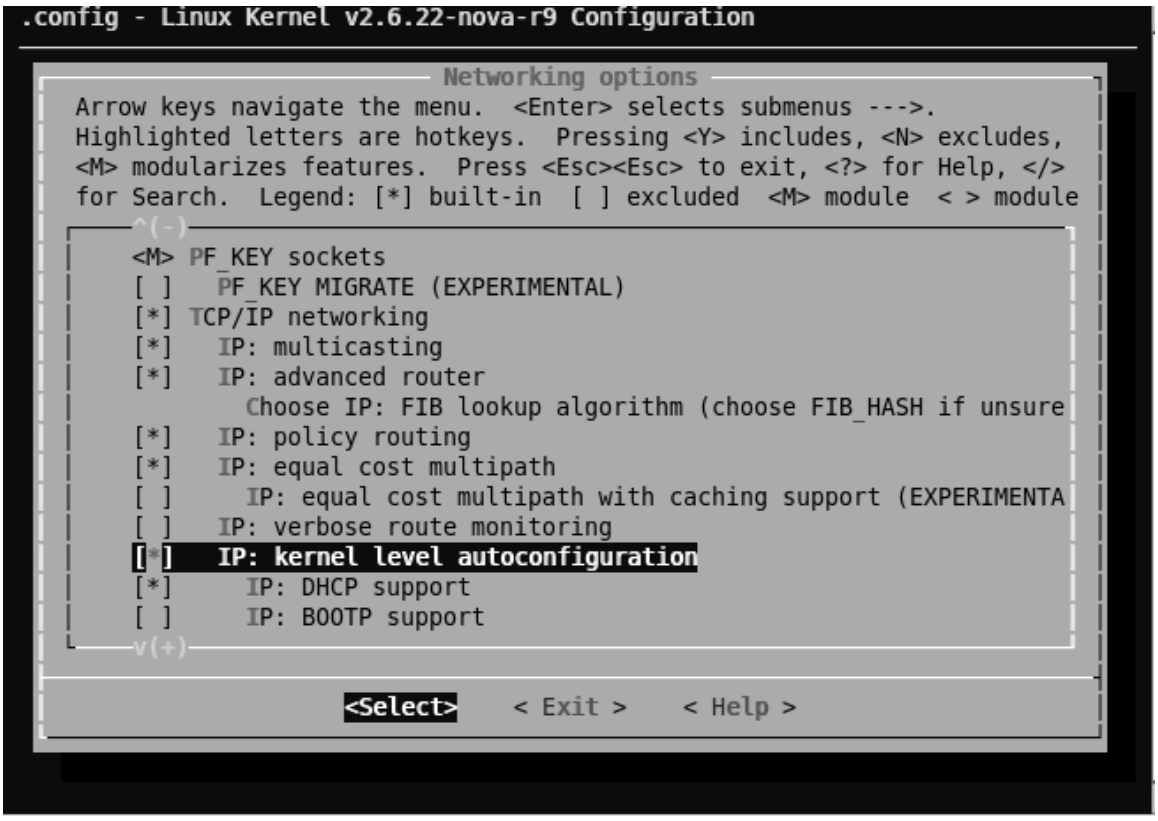



Figura 1.5 Configuración de kernel

```
File systems --->
  Network File Systems --->
      <*> NFS file system support
      [*]       Provide NFSv3 client support
      <*> NFS server support
      [*]       Provide NFSv3 server support
      [*]       Provide NFS server over TCP support
      [*] Root file system on NFS
```

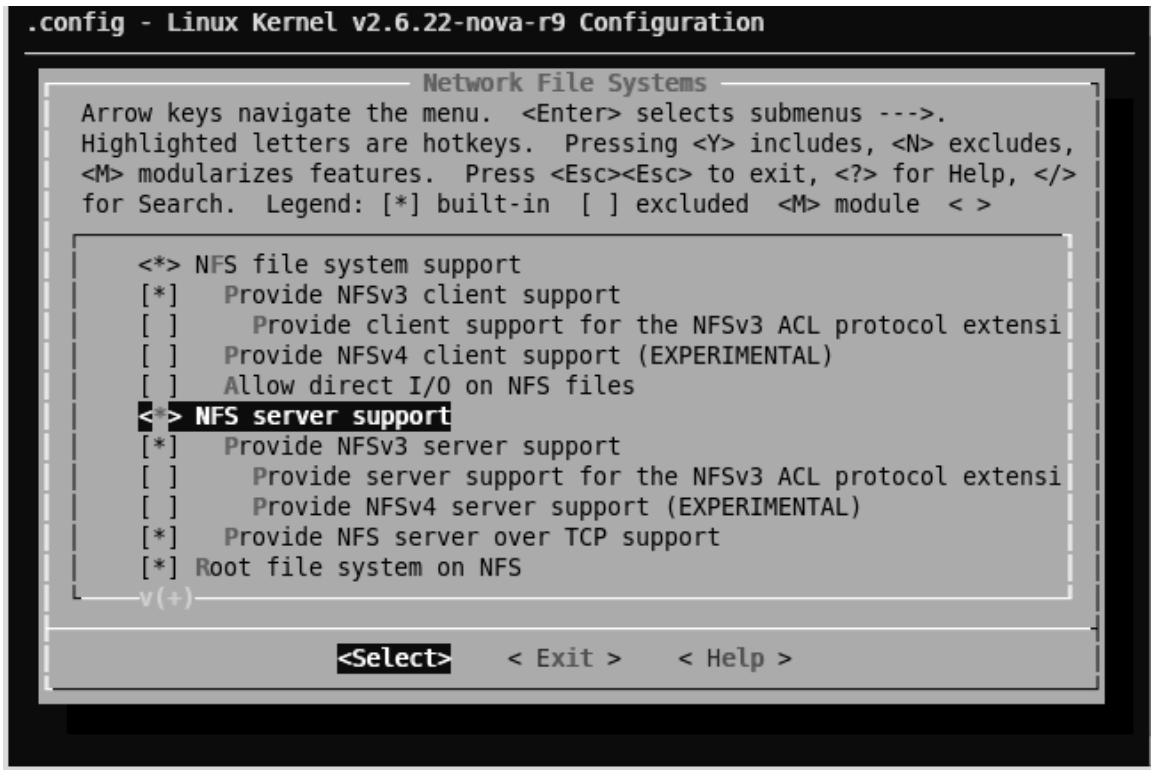



Figura 1.6 Configuración de kernel

```
File systems --->
     Pseudo filesystems --->
          [*] proc file system support
          [*] Virtual memory file system support (former shm fs)
          [*]  Tmpfs Posix Access Control Lists
```

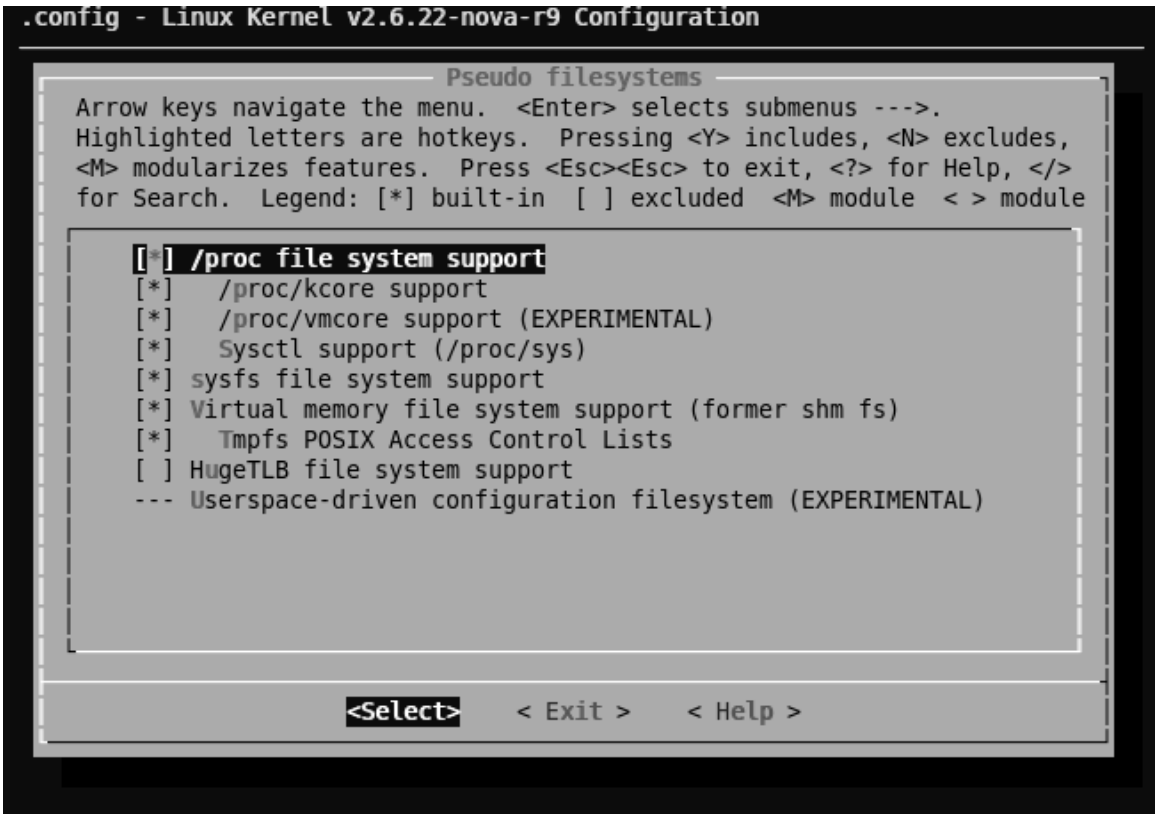



Figura 1.7 Configuración de kernel

Es necesario además incluir dentro del kernel [*] el driver de red del cliente a utilizar, exceptuando e1000, pc32 y SiS900 que se encuentran en el initramfs de nova y pueden ser compilados como módulos. Se debe tener en cuenta que estas opciones de compilación del kernel deben ser añadidas a las opciones específicas para la imagen cliente. Aquí solo se explican las opciones necesarias para una imagen a utilizar por las terminales ligeras.

Al terminal de compilar el kernel, initramfs y los módulos deben copiarse para la imagen. Para ello se utilizarán los siguientes comandos:

```
# cp /boot/* /diskless/baire/boot/
# mv /lib/modules/* /diskless/baire/lib/modules/
```

Ahora se copian para su lugar el kernel, initramfs y los módulos del servidor:

```
# cp /backup/* /boot/
# rm /backup/*
# cp -r /backup/* /lib/modules/
```

Una vez copiados los archivos borramos el directorio backup.

```
# rm -rf /backup
```

Para no tener que cambiar el archivo */diskless/nova/pxelinux.cfg/default* se hace un enlace simbólico (symlink) al kernel y al initramfs que se encuentran en la imagen cliente de la siguiente forma:

```
# ln -sf /diskless/baire/boot/nombredelkernel
  /diskless/nova/kernel
# ln –sf /diskless/baire/boot/nombredelinitramfs
  /diskless/nova/initramfs.cpio.gz
```

De esta forma si se necesita cambiar el kernel o initramfs de la imagen solo hay que cambiar el lugar a donde apunta el enlace simbólico para que apunte al nuevo kernel o initramfs.

Ahora es necesario hacer una limpieza a la imagen ya que como se ha explicado parte del sistema se copia en la RAM del cliente directamente y este espacio debe ser reducido. Esto se realiza como se describe a continuación:

Es necesario mover */etc/gconf* para otro lugar ya que no se puede remover debido a la información que contiene.

```
# mv /etc/gconf/ /usr/
# ln -sf /usr/gconf /etc/gconf
```

Borrar directorios temporales para achicar la imagen. En esto se incluye borrar el portage de la imagen ya que no se necesitará instalar nada desde las imágenes al menos usando un portage propio.

```
# rm -rf /var/tmp/*
# rm -rf /tmp/*
# rm -rf /usr/portage/*
```

Los directorios removidos son los recomendados por esta guía. Pero existen otros que deben ser limpiados de lo que no sea indispensable. Como son */var*, */etc*, */home* y */root* que son los directorios que se copian directamente en la RAM temporalmente, cuando se usa la opción *-ro* en los parámetros del kernel. En caso de no poder borrar alguna información. Se realizan los mismos pasos que con el directorio */etc/gconf*.

Ahora queda configurar el archivo */etc/fstab* el cual realiza el montaje de las particiones del sistema cuando inicia. Se debe especificar en caso de que algún directorio sea exportado fuera del sistema de archivos donde es que está. Este archivo es el que permite que se pueda integrar esta solución con la solución de "Homes Compartidos". Lo que garantiza que el usuario pueda escribir en su directorio */home* y guardar su configuración y acceder a ella desde cualquier cliente.

Después es necesario establecer la contraseña de root para la imagen cliente. Esto es importante ya que en caso de error se necesitará dar soporte a la imagen y puede hacerse remoto especificando en el kernel la opción *rw*.

Al llegar a este momento ya puede intentar iniciar sus terminales ligeras. En caso de que el sistema no levante el entorno gráfico, se deberá configurar. Para esto iniciaremos un cliente con permisos de escritura o *rw*. Una vez dentro del cliente pues se autentica como root en una terminal. Y se ejecuta lo siguiente:

```
# X -configure
# X -config /root/xorg.conf.new
```

Si se logra ver una pantalla de entorno gráfico entonces está listo para levantar el mismo. Se copia el archivo de configuración para */etc/X11/* bajo el nombre de *xorg.conf*. Utilizando el comando:

```
# mv /root/xorg.conf.new /etc/X11/xorg.conf
```

## Conclusiones Generales

El estudio acerca de los principales sistemas para terminales ligeras en el mundo y el estudio de su funcionamiento, llevó a la creación de esta solución para la distribución Nova, cumpliéndose satisfactoriamente los objetivos específicos planteados.

Se logró configurar la imagen servidora y la imagen cliente que cumple con los requerimientos de un sistema para terminales ligeras, obteniéndose una solución final de alta calidad.

El servidor para terminales ligeras que se propone brinda todas las facilidades y flexibilidad que proporcionan los sistemas investigados para terminales ligeras, significando una potencial solución para la sociedad cubana por la independencia tecnológica que trae aparejada.

El principal aporte de esta solución es un initramfs modificado para eliminar restricciones presentes en este, que impedían un mayor desempeño de las imágenes a utilizar por los clientes. Se espera que este trabajo siente sólidas bases para ayudar de forma más rápida a la informatización de la sociedad cubana.

## Recomendaciones

Sería de gran utilidad automatizar los procesos que se realizaron de forma manual en este trabajo con una aplicación visual que brinde mayores opciones de configurabilidad, lo cual queda recomendado como tema de tesis para generaciones futuras. Estas opciones pudieran ser, poder conocer las terminales que están funcionando, brindar distintos tipos de imágenes para distintas terminales, editar archivos de configuración de protocolos y kernel,

Además se recomienda utilizar esta solución en la Universidad de las Ciencias Informáticas y en Cuba en general, por la significación que puede tener en un futuro, pues los problemas de hardware son mayores cada día y por la necesidad de ahorrar energía eléctrica.

**Referencias Bibliográficas**

1. La Fleha. *Tu diario de ciencia y tecnologia.* [En línea] 2008. [Citado el: 17 de Abril de 2008.] Disponible en: http://www.laflecha.net/canales/blackhats/noticias/200501211.

2. FSF America Latina. *Fundación Software Libre America Latina.* [En línea] 2008. [Citado el: 22 de 4 de 2008.] Disponible en: http://www.fsfla.org/svnwiki/about/what-is-free-software.es.html.

3. Donde no llegaron las Hadas. *Aventuras y desventuras de un linuxero.* [En línea] 2008. [Citado el: 22 de 4 de 2008.] Disponible en: http://elchicosinhada.wordpress.com/2007/03/22/mi-distro-gentoo/.

4. ¡Virtual Labs. *Informatica Virtual Research Labs.* [En línea] 2008. [Citado el: 2 de 5 de 2008.] Disponible en: http://www.ivlabs.org/home/?p=321.

5. Estudio de Soluciones Completas de Acceso Gráfico Remoto. [En línea] 2008. [Citado el: 3 de 5 de 2008.] Disponible en: http://torio.unileon.es/~mediawiki/index.php/Estudio_de_soluciones_completas_de_acceso_gr%C3%A1fico_remoto.

6. Insight. [En línea] 2008. [Citado el: 3 de 5 de 2008.] Disponible en: http://es.insight.com/content/2xsoftware/thinclienterver.

7. CPD. *Grupo ContraPunto Digital .* [En línea] 2008. [Citado el: 5 de 5 de 2008.] Disponible en: http://www.grupocpd.com/archivos_documentos/carpeta_reciclared/informe_beneficio_clientes_ligeros_web.sxw/view.

8. Linux para seres humanos. [En línea] 2008. [Citado el: 5 de 5 de 2008.] Disponible en: http://www.linuxparasereshumanos.com/2007/06/06/diskless-booting-en-espaol/.

9. **Fedorovich, Mijail Hurtado.** *INITRAMFS Marca la diferencia.* [Documento] Habana : s.n., 2008.

10. LIBERALIA TEMPUS. *When the things work well.* [En línea] 2008. [Citado el: 6 de 5 de 2008.] Disponible en: http://www.liberaliatempus.com/articulos/linux/configurar-un-servidor-dhcp.html.

11. Wikipedia. *La enciclopedia libre.* [En línea] 2008. [Citado el: 6 de 5 de 2008.] Disponible en: http://es.wikipedia.org/wiki/TFTP.

12. Wikipedia. *La enciclopedia libre.* [En línea] 2008. [Citado el: 6 de 5 de 2008.] Disponible en: http://es.wikipedia.org/wiki/NFS.

13. **SFEIR, Fernando y TASSO, Alejandro.** JEITICS. [En línea] 2008. [Citado el: 7 de 5 de 2008.] Disponible en: http://cs.uns.edu.ar/jeitics2005/Trabajos/pdf/40.pdf.

**Bibliografía**

1. X-Caleta.com. *Winconnect Server XP.* [En línea] 2008. [Citado el: 20 de 3 de 2008.]Disponible en: http://www.x-caleta.com/2008/01/14/winconnect-server-xp.html.

2. Wikipedia la enciclopedia libre. *PXE.* [En línea] 2008. [Citado el: 2 de 4 de 2008.] Disponible en: http://es.wikipedia.org/wiki/PXE.

3. Wikipedia la enciclopedia libre. *Dynamic Host Configuration Protocol.* [En línea] 2008. [Citado el: 5 de 4 de 2008.] Disponible en: http://es.wikipedia.org/wiki/DHCP.

4. Wikipedia la enciclopedia libre. *TFTP.* [En línea] 2008. [Citado el: 5 de 4 de 2008.] Disponible en: http://es.wikipedia.org/wiki/TFTP.

5. Wikipedia la enciclopedia libre. *Network File System.* [En línea] 2008. [Citado el: 6 de 4 de 2008.] Disponible en: http://es.wikipedia.org/wiki/NFS.

6. Soluciones inteligentes. [En línea] 2008. [Citado el: 21 de 2 de 2008.] Disponible en: http://www.solinteligentes.com/terminales.html.

7. LTSP.org. *Linux Terminal Server Project.* [En línea] 2008. [Citado el: 6 de 3 de 2008.] Disponible en: http://www.ltsp.org/.

8. Wikipedia la enciclopedia libre. *LTSP.* [En línea] 2008. [Citado el: 8 de 3 de 2008.] Disponible en: http://es.wikipedia.org/wiki/LTSP.

9. Diskless Embedded Technology PC. *Diet-PC.* [En línea] 2008. [Citado el: 8 de 3 de 2008.] Disponible en:http://spanish.osstrans.net/software/diet-pc.html.

10. Wikipedia la enciclopedia libre. *Cliente liviano.* [En línea] 2008. [Citado el: 8 de 3 de 2008.] Disponible en: http://es.wikipedia.org/wiki/Cliente_liviano.

11. Especulando. *Thinstation.* [En línea] 2008. [Citado el: 10 de 3 de 2008.] Disponible en: http://www.especulando.es/tag/cliente-ligero/.

12. EPATec. *eLux NG.* [En línea] 2008. [Citado el: 11 de 3 de 2008.] Disponible en:http://store.epatec.net/es/product_info.php?products_id=36.

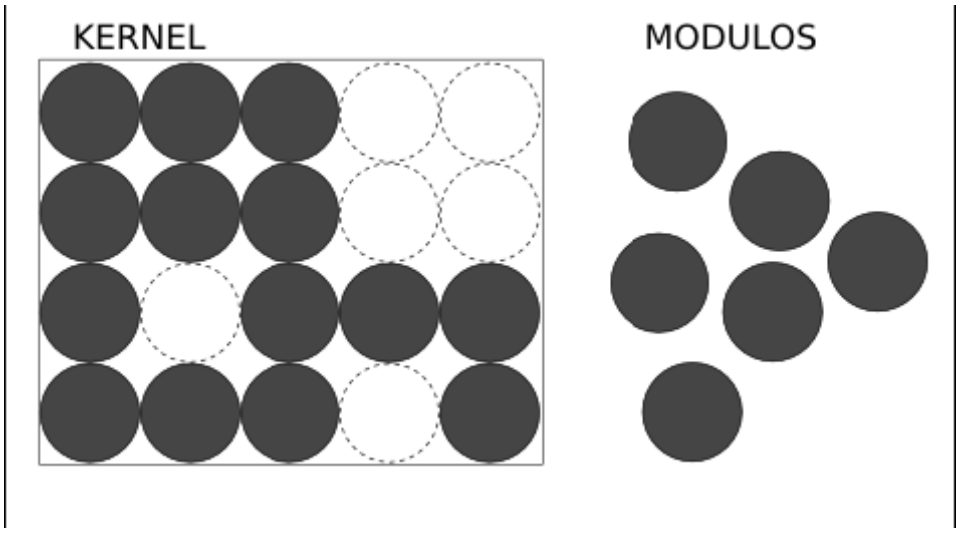
13. S-4. *Cliente Ligero .* [En línea] 2008. [Citado el: 11 de 3 de 2008.] Disponible en: http://www.s-4.es/mt_ClienteLigero.aspx.

14. Microsoft y la interoperabilidad. *Partners de Windows Server 2003 Terminal Services.* [En línea] 2008. [Citado el: 15 de 3 de 2008.] Disponible en: http://www.microsoft.com/spain/interop/customers/TSPartners.mspx.

15. Wyse Global Leader in Thin Computing. *Material thin computing.* [En línea] 2008. [Citado el: 15 de 3 de 2008.] Disponible en: http://www.wyse.com/es/material.asp.

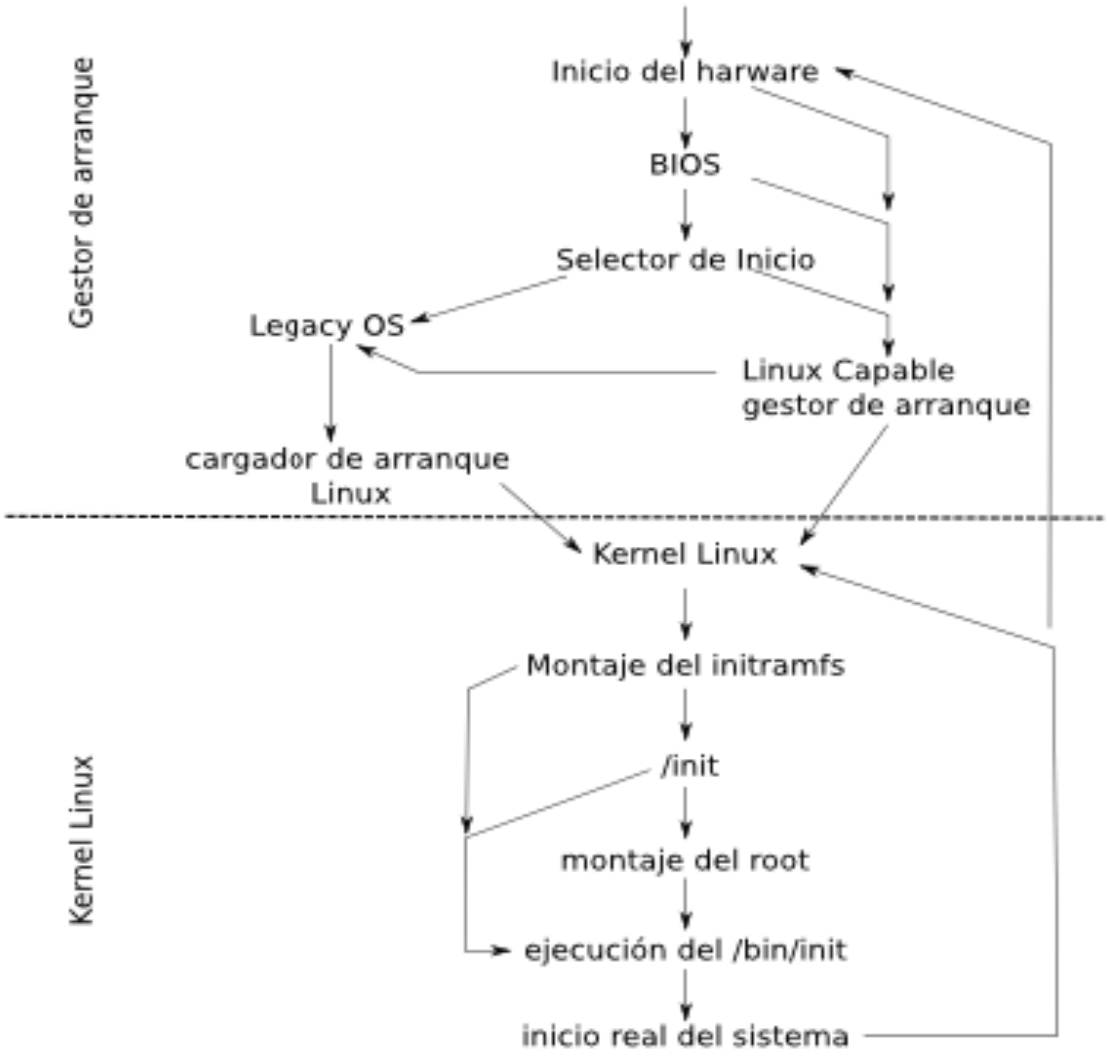
# Anexos

## Anexo I




**Idea gráfica de un kernel Linux y los módulos.**

## Anexo II




**Secuencia de inicio de Linux tomada del sitio de los desarrolladores de Linux.**

## Anexo III

### Initramfs modificado

```
#!/bin/busybox ash

#TODO lo del rw

RO=".nfs"
MIX=".mix"
RW=".rw"
DINAMIC_DIRS="media mnt tmp proc sys dev ${RO}"

for opt in ${CMDLINE}
        do

        case ${opt} in
                ip\=*)
                        IP=`parse_opt "${opt}"`
                ;;
                nfsroot\=*)
                        NFSROOT=`parse_opt "${opt}"`
                ;;
                nfsparms\=*)
                        NFSPARMS=`parse_opt "${opt}"`
                ;;
                aufs)
                        AUFS='1'
                ;;
                esac
        done

init=/linuxrc

mount_the_root(){

        REAL_ROOT="${NFSROOT}"
        if [ ! -z ${REAL_ROOT} ]
                then
                        #good_msg "Montando NFS root desde
${REAL_ROOT}"
                        mount -t nfs -o ${NFSPARMS}  ${REAL_ROOT}
${RO} 2>/dev/null
                        if [ $? != 0 ]
                        then
```

```
                                NFSROOT=
                                mount_the_root
                        fi
                else
                        warn_msg "please type the correct server to
boot"
                        read -p ":" NFSROOT
                        good_msg "trying to boot $RSC${NFSROOT}"
                        mount_the_root
        fi
        }

        exist(){
                ITEM=$1
                shift 1
                case $* in
                        *","$ITEM","* | $ITEM","* | *","$ITEM |
$ITEM)exit 0
                        ;;
                        *)echo $ITEM
                        ;;
                esac
                }

        setup_dhcp(){
                if ! [ -f /init_scripts/udhcpc.scripts  ]
                        then
                        # ALERTA!!!! no alinear este codigo porque
este genera un archivo
                        # en tiempo de ejecucion y puede afectar la
estructura del mismo
                                cat > /init_scripts/udhcpc.scripts
<<EOF
#!/bin/busibox ash

case \${1} in
        bound)
                [ -n "\$broadcast" ] && BROADCAST="broadcast
\$broadcast"
                [ -n "\$subnet" ] && NETMASK="netmask \$subnet"
                [ -n "\$rootpath" ] && echo "\$rootpath" > /rootpath
                #[ -n "\$router" ] && echo "\$router" > /router
                #[ -n "\$dns" ] && echo "\$dns" > /dns
```

```
                busybox ifconfig \$interface \$ip \$BROADCAST
\$NETMASK
        ;;
        deconfig)
                busybox ifconfig \$interface 0.0.0.0
        ;;
esac
EOF
                chmod +x /init_scripts/udhcpc.scripts
                fi
                #trata de iniciar la red mediante DHCP
                udhcpc  -n -s /init_scripts/udhcpc.scripts #-r
rootpath

                }


prepare_root(){

        load_devices "net fs"

        #se inicia la red por dhcp a solicitud de la linea de comandos
        [ ${IP} == "dhcp" ] && setup_dhcp

        if [ "`exist "rw" $NFSPARMS`" != "rw" ]
        then
                #se trabaja directamente sobre el servidor sin hacer
copias en la pc cliente
                warn_msg "Este modo solo soporta un solo cliente"
                good_msg "Los parametros de montaje del nfs son:
$BLC$NFSPARMS"
                good_msg "El servidor es: $BLC$NFSROOT"
                mount -t tmpfs tmpfs ${NEWROOT}
                cd "${NEWROOT}"
                mkdir ${RO}
                mount_the_root
                make_root_links "${RO}"
        else
                mount -t tmpfs tmpfs ${NEWROOT}
                cd "${NEWROOT}"
                #se crean los directorios dinamicos tales como "sys"
"proc" "tmp" ...
                for dir in ${DINAMIC_DIRS}
                do
                        mkdir -p ${dir}
```

```
                done
                #se permite que cualquier usuario pueda escribir en
tmp
                chmod 1777 "tmp"
                mount_the_root
                                if [ ! -z ${AUFS} ]
                                        then
                                                setup_aufs
                                        else
                                                good_msg "Copiando
archivos a la memoria..."
                                                cp -a "${RO}/var"
"${RO}/etc" "${RO}/root" "${RO}/home" "${NEWROOT}/"
                                                make_root_links
"${RO}"
                                fi
        fi
        start_root
        }
```

## Glosario de términos

# A

**Archivos binarios:** Un Archivo binario es un archivo informático que contiene información de cualquier tipo, codificada en forma binaria para el propósito de almacenamiento y procesamiento en ordenadores. Por ejemplo los archivos informáticos que almacenan texto formateado o fotografías. Muchos formatos binarios contienen partes que pueden ser interpretados como texto. Un archivo binario que sólo contiene información de tipo textual sin información sobre el formato del mismo se dice que es un archivo de texto plano. Habitualmente se contraponen los términos 'archivo binario' y 'archivo de texto' de forma que los primeros no contienen solamente texto. En el caso particular de este documento se refiere a aquellos archivos binarios que son ejecutados en el procesador.

# B

**Biblioteca:** Término informático para referirse a las bibliotecas de vínculos dinámicos, conocida también como librería.

**Bug:** Un defecto de software (computer bug en inglés), es el resultado de un fallo o deficiencia durante el proceso de creación de programas de ordenador o computadora (software). Dicho fallo puede presentarse en cualquiera de las etapas del ciclo de vida del software aunque los más evidentes se dan en la etapa de desarrollo y programación. Los errores pueden suceder en cualquier etapa de la creación de software.

# C

**Código fuente:** Puede definirse como:

- Un conjunto de líneas que conforman un bloque de texto, escrito según las reglas sintácticas de algún lenguaje de programación destinado a ser legible por humanos.
- Un Programa en su forma original, tal y como fue escrito por el programador, no es ejecutable directamente por el computador, debe convertirse en lenguaje de maquina mediante compiladores, ensambladores o intérpretes.

Normalmente está destinado a ser traducido a otro código, llamado código objeto, ya sea lenguaje máquina nativo para ser ejecutado por una computadora o bytecode para ser ejecutado por un intérprete.

**Consola:** Una consola es un programa que tiene como objetivo principal, la interacción del usuario con el sistema operativo a base de líneas de comandos. Las consolas más usadas por los usuarios son las consolas virtuales, que son las que se pueden ver desde su entorno gráfico, por ejemplo:

Gnome-terminal, Kterm, Yaquake, etc.

**Consumo de recursos:** Término informático que se refiere a la forma en que el software interactúa con los recursos de hardware de un ordenador.

## D

**Directorio raíz:** Ver la definición de Root.

## E

**Enlaces estáticos:** Una biblioteca estática es aquella que se enlaza en tiempo de compilación (en oposición a una de enlace dinámico, que se enlaza en tiempo de ejecución). La ventaja de este tipo de enlace es que hace que un programa no dependa de ninguna librería (puesto que las enlazó al compilar), haciendo más fácil su distribución.

## G

**Gentoo:** Gentoo Linux es una distribución GNU/Linux orientada a usuarios con cierta experiencia en este sistema operativo, fue fundada por Daniel Robbins, basada en la inactiva distribución llamada Enoch Linux. Ya para el año 2002, ésta última pasa a denominarse Gentoo Linux.

El nombre Gentoo proviene del nombre en inglés del pingüino papúa. Nótese que la mascota de Linux es un pingüino.

**GNU/Linux:** GNU/Linux es, a simple vista, un Sistema Operativo. Es una implementación de libre distribución UNIX para computadoras personales (PC), servidores, y estaciones de trabajo. Fue desarrollado para el i386 y ahora soporta los procesadores i486, Pentium, Pentium Pro y Pentium II,

así como los clones AMD y Cyrix. También soporta máquinas basadas en SPARC, DEC Alpha, PowerPC/PowerMac, y Mac/Amiga Motorola 680x0.

Como sistema operativo, GNU/Linux es muy eficiente y tiene un excelente diseño. Es multitarea, multiusuario, multiplataforma y multiprocesador; en las plataformas Intel corre en modo protegido; protege la memoria para que un programa no pueda hacer caer al resto del sistema; carga sólo las partes de un programa que se usan; comparte la memoria entre programas aumentando la velocidad y disminuyendo el uso de memoria; usa un sistema de memoria virtual por páginas; utiliza toda la memoria libre para cache; permite usar bibliotecas enlazadas tanto estática como dinámicamente; se distribuye con código fuente; usa hasta 64 consolas virtuales; tiene un sistema de archivos avanzado pero puede usar los de los otros sistemas; y soporta redes tanto en TCP/IP como en otros protocolos.

## K

**Kernel:** Ver definición de Núcleo.

## L

**Licencia libre:** Copyleft o copia permitida comprende a un grupo de derechos de propiedad intelectual caracterizados por eliminar las restricciones de distribución o modificación de las que adolece el copyright, con la condición de que el trabajo derivado se mantenga con el mismo régimen de propiedad intelectual que el original.

Bajo tales licencias pueden protegerse una gran diversidad de obras, tales como programas informáticos, arte, cultura y ciencia, es decir prácticamente casi cualquier tipo de producción creativa. Sus partidarios la proponen como alternativa a las restricciones que imponen las normas planteadas en los derechos de autor, a la hora de hacer, modificar y distribuir copias de una obra determinada. Se pretende garantizar así una mayor libertad para que cada receptor de una copia, o una versión derivada de un trabajo, pueda, a su vez, usar, modificar y redistribuir tanto el propio trabajo como las versiones derivadas del mismo. Así, y en un entorno no legal, puede considerarse como opuesto al copyright o derechos de autor tradicionales.

**LINUX:** Núcleo presente en los sistemas GNU/Linux.

**Librería:** Ver la definición de biblioteca.

## N

**Núcleo:** En informática, el núcleo (también conocido en español con el anglicismo kernel, de raíces germánicas como kern) es la parte fundamental de un sistema operativo. Es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema. Como hay muchos programas y el acceso al hardware es limitado, el núcleo también se encarga de decidir qué programa podrá hacer uso de un dispositivo de hardware y durante cuánto tiempo, lo que se conoce como multiplexado. Acceder al hardware directamente puede ser realmente complejo, por lo que los núcleos suelen implementar una serie de abstracciones del hardware. Esto permite esconder la complejidad, y proporciona una interfaz limpia y uniforme al hardware subyacente, lo que facilita su uso para el programador.

## R

**RAM:** La memoria de acceso aleatorio, o memoria de acceso directo (en inglés: Random Access Memory, cuyo acrónimo es RAM), o más conocida como memoria RAM, se compone de uno o más chips y se utiliza como memoria de trabajo para programas y datos. Es un tipo de memoria temporal que pierde sus datos cuando se queda sin energía (por ejemplo, al apagar la computadora), por lo cual es una memoria volátil.

**Rendimiento:** Un término usado en la informática para dar medida de cuán eficaz puede ser un software en el uso de los recursos de hardware, dígase memoria RAM, procesador, disco duro, etc.

**Root:** El término informático “root” (raíz), en la familia de sistemas Unix/Linux/BSD, suele tener doble significado:

1. Directorio raíz “/” del cual se derivan los demás directorios del sistema, es decir, es el directorio padre. Comúnmente los directorios que se desprenden del root son “bin, boot, dev, etc, proc, home, mnt, sbin, sys, root, usr, tmp, var...”
2. Superusuario del sistema, es decir, este tipo de usuarios tiene el permiso de hacer cualquier cosa con el sistema, ser root requiere un alto grado de responsabilidad.

## S

**Sistema operativo:** Un sistema operativo es un software de sistema, es decir, un conjunto de programas de computadora destinado a permitir una administración eficaz de sus recursos. Comienza a trabajar cuando se enciende el computador, y gestiona el hardware de la máquina desde los niveles más básicos, permitiendo también la interacción con el usuario.

**Salida estándar:** Cuando un programa emite un mensaje determinado, este último es capturado por la salida estándar para luego ser procesado o mostrada por otro programa.

**Superusuario:** Usuario de administración en sistemas GNU-Linux, también conocido como root.

## T

**Terminal:** Término derivado del inglés que significa consola.

## U

**UNIX:** Unix (registrado oficialmente como UNIX®) es un sistema operativo portable, multitarea multiusuario; desarrollado, en principio, en 1969 por un grupo de empleados de los laboratorios Bell de AT&T, entre los que figuran Ken Thompson, Dennis Ritchie y Douglas McIlroy.