REPÚBLICA DEL ECUADOR
Escuela Politécnica Nacional
"E SCIENTIA HOMINIS SALUS"

La versión digital de esta tesis está protegida por la Ley de Derechos de Autor del Ecuador.

Los derechos de autor han sido entregados a la "ESCUELA POLITÉCNICA NACIONAL" bajo el libre consentimiento de los autores.

Al consultar esta tesis deberá acatar con las disposiciones de la Ley y las siguientes condiciones de upso:

- Cualquier uso que haga de estos documentos o imágenes deben ser sólo para efectos de investigación o estudio académico, y usted no puede ponerlos a disposición de otra persona.

- Usted deberá reconocer el derecho del autor a ser identificado y citado como el autor de esta tesis.

- No se podrá obtener ningún beneficio comercial y las obras derivadas tienen que estar bajo los mismos términos de licencia que el trabajo original.

El Libre Acceso a la información, promueve el reconocimiento de la originalidad de las ideas de los demás, respetando las normas de presentación y de citación de autores con el fin de no incurrir en actos ilegítimos de copiar y hacer pasar como propias las creaciones de terceras personas.

Respeto hacia sí mismo y hacia los demás.

ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DISTRIBUCIÓN DE GNU/LINUX PARA USO EN LOS LABORATORIOS DE LA CARRERA DE ELECTRÓNICA Y REDES DE INFORMACIÓN

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

CRISTIAN OSWALDO GILER EGÜEZ
criler@hotmail.com

ALEXANDER WLADIMIR SANTAMARÍA SALVADOR
alxs_ws@hotmail.com

DIRECTOR: DR. ENRIQUE MAFLA GALLEGOS
mafla@epn.edu.ec

Quito, Enero 2011

DECLARACIÓN

Nosotros, Cristian Giler Egüez y Alexander Santamaría Salvador, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Cristian Giler Egüez

Alexander Santamaría Salvador

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por: Cristian Giler Egüez y Alexander Santamaría Salvador, bajo mi supervisión.

Dr. Enrique Mafla Gallegos
DIRECTOR DE PROYECTO

AGRADECIMIENTO

Mis más sinceros agradecimientos:

A Dios,

A mis padres Gloria y Juvenal, por su amor, por el apoyo incondicional y sus lecciones de vida que han forjado la persona que soy,

A mis hermanos Mariela y Carlos, por compartir amor, comprensión y hermandad en los buenos y malos momentos,

A toda la comunidad Politécnica: a sus profesores, y a todos los colegas y compañeros que de alguna manera han incidido en mi formación profesional,

A Enrique Mafla; por medio de sus directrices ha logrado que nuestro proyecto sea de la calidad que la EPN lo merece,

A tod@s los amig@s que han compartido conmigo este largo (laaaaaaaargo) camino de convertirse en Ingeniero (de la Poli), camino que no ha sido el más fácil; lleno de largas jornadas de trabajo, amanecidas, experiencias, y claro está, sin olvidar: días, tardes, noches, y madrugadas de felicidad y ocio.*

A Daniel Carrasco, promotor de la idea fundamental de este proyecto,

A Alexander Santamaría (Alex!), por aunar fuerzas para el desarrollo de este proyecto; por impulsarme en momentos extenuantes y hacerme ver que se puede ir más allá de los límites que uno piensa que existen.

A todos: simplemente GRACIAS!

Cristian Oswaldo Giler Egüez

(*) Andrés S. Patricia N. Patricio P. Jesús G. Salomé V. Luis E. Mishel B. Ma.Elena C. Christiam M. Marco V. Dennys Ch. Ramiro V. Ángel J. Víctor L. Soraya A. Tulia P. Paula C. Javier Y. Gustavo R. Luis A. Daniel C. Ramiro C. Richard M. Alex S. Anita R. Danilo A. Diana A. Luigi C. Juan A. Fernando L. Pachu L. Morita T. .....y MUUUUUUUUUCH@S MÁS, que las 266 páginas del proyecto no alcanzarían para nombrarlos.

AGRADECIMIENTO

Gracias a Dios,

A todas las personas que me dieron su apoyo para cumplir esta meta,

A mi familia, a "la inmensa", a mis compañeros de lucha, y a los ingenieros que guiaron este proyecto.

Alexander Wladirmir Santamaría Salvador.

DEDICATORIA

"A mi padre."

Cristian

DEDICATORIA

"Para mi abuelito!!!"

Alexander Santamaría.

TABLA DE CONTENIDOS

TABLA DE CONTENIDOS vi

ÍNDICE DE FIGURAS x

ÍNDICE DE TABLAS xii

CAPITULO 1 1

MARCO TEÓRICO, PROBLEMÁTICA, ANÁLISIS DE LA SITUACIÓN ACTUAL Y ANÁLISIS DE REQUERIMIENTOS. 1

1.1 COMPONENTES PRINCIPALES DE UN SISTEMA OPERATIVO BASADO EN GNU/LINUX 1

1.1.1 KERNEL LINUX 2

1.1.2 BIBLIOTECAS DEL SISTEMA 3

1.1.3 UTILITARIOS DEL SISTEMA Y DEL USUARIO 3

1.2 IDENTIFICACIÓN DEL PROBLEMA 4

1.2.1 APROVECHAMIENTO DE HORAS CLASE 4

1.2.2 DISPONIBILIDAD Y FUNCIONALIDAD DE HERRAMIENTAS 5

1.3 METODOLOGÍA PARA EL DESARROLLO DEL PROYECTO 5

1.3.1 ANÁLISIS DE LA SITUACIÓN ACTUAL DE LOS LABORATORIOS. 6

1.3.2 ANÁLISIS DE LOS REQUERIMIENTOS DEL SISTEMA. 6

1.3.3 DISEÑO DEL SISTEMA 7

1.3.4 CONSTRUCCIÓN DEL SISTEMA. 7

1.3.5 PRUEBAS DEL SISTEMA. 8

1.4 DISEÑO DE LOS CUESTIONARIOS 8

1.4.1 ENTREVISTA DE LA GESTIÓN DE SERVICIOS DE TI EN LOS LABORATORIOS 10

1.4.2 ENCUESTA DE LA SITUACIÓN ACTUAL Y DE REQUERIMIENTOS . 11
1.5 ANÁLISIS DE RESULTADOS DE LOS CUESTIONARIOS 18
1.5.1 ANÁLISIS DE RESULTADOS DE LA ENTREVISTA DE LA GESTIÓN DE SERVICIOS DE TI EN LOS LABORATORIOS 18
1.5.2 ANÁLISIS DE RESULTADOS DE LA ENCUESTA DE LA SITUACIÓN ACTUAL DE LOS LABORATORIOS Y DE REQUERIMIENTOS FUNCIONALES. 29
CAPÍTULO 2 38
DISEÑO Y CONSTRUCCIÓN DEL SISTEMA 38
2.1 DISEÑO DEL SISTEMA A IMPLEMENTAR 38
2.1.1 CAPAS DEL SISTEMA 39
2.1.2 KERNEL A IMPLANTAR 39
2.1.3 MÓDULOS DEL KERNEL 40
2.1.4 COMPILADORES Y LIBRERÍAS 41
2.1.5 ESTRUCTURA DE DIRECTORIOS 41
2.1.6 PAQUETES DE SOFTWARE 44
2.2 CONSTRUCCIÓN DEL SISTEMA 44
2.2.1 CONSTRUCCIÓN DEL SISTEMA BASE 45
2.2.2 INSTALACIÓN DE HERRAMIENTAS REQUERIDAS Y COMPLEMENTARIAS DEL SISTEMA 100
2.3 CREACIÓN DEL LIVECD 118
2.3.1 REEMPLAZO DEL KERNEL 119
2.3.2 CONFIGURACIÓN DEL KERNEL 119
2.3.3 LIBERACIÓN DE ESPACIO 120
2.3.4 CONSTRUCCIÓN DE DATOS 120

2.3.5 CREACIÓN DEL ISO 121
CAPÍTULO 3 124
PRUEBAS DEL SISTEMA 124
3.1 PRUEBAS PARA VERIFICAR EL CUMPLIMIENTO DEL ALCANCE DEL PROYECTO 125
3.1.1 ESPACIO EN DISCO QUE OCUPA EL SISTEMA 126
3.1.2 TIEMPO DE ARRANQUE DEL SISTEMA 127
3.1.3 COMPATIBILIDAD CON INTERFACES 129
3.1.4 COMPATIBILIDAD CON PROCESADORES 132
3.1.5 COMPATIBILIDAD CON MONITORES 133
3.2 PRUEBAS PARA VERIFICAR EL FUNCIONAMIENTO DE LAS HERRAMIENTAS INSTALADAS 135
3.2.1 GESTIÓN DE REDES 135
3.2.2 SISTEMAS OPERATIVOS 146
3.2.3 HARDWARE DE CONECTIVIDAD 148
3.2.4 SEGURIDAD DE REDES 156
3.2.5 APLICACIONES DISTRIBUIDAS 159
CAPÍTULO 4 162
CONCLUSIONES Y RECOMENDACIONES 162
4.1 CONCLUSIONES 162
4.2 RECOMENDACIONES 164
CAPÍTULO 5 166
BIBLIOGRAFÍA Y ANEXOS 166
5.1 BIBLIOGRAFÍA 166
5.2 ANEXOS 168

5.2.1 ANEXO A – GUÍA PARA LA ENTREVISTA DE LA GESTIÓN DE SERVICIOS DE TI EN LOS LABORATORIOS 168

5.2.2 ANEXO B - ENCUESTA DE LA SITUACIÓN ACTUAL Y DE REQUERIMIENTOS..... 175

5.2.3 ANEXO C – FHS (FILESYSTEM HIERARCHY SYSTEM) 179

5.2.4 ANEXO D – GUÍA DE IMPLEMENTACIÓN..... 190

5.2.5 ANEXO E – SCRIPT COMPRESSDOC 228

5.2.6 ANEXO F – SCRIPT PARA CONSTRUCCIÓN DE DATOS DEL LIVE CD 240

5.2.7 ANEXO G – SCRIPT PARA LA CREACIÓN DEL ISO 243

5.2.8 ANEXO H – GUÍA DE USUARIO..... 244

ÍNDICE DE FIGURAS

Figura 1.1 Componentes de un sistema operativo basado en Linux..... 2

Figura 1.2 Metodología de soporte al servicio..... 20

Figura 1.3 Metodología de provisión del servicio 24

Figura 1.4 Horas a la semana promedio de uso del sistema Linux. 30

Figura 1.5 Tiempo de respuesta a solicitudes de instalación de nuevos paquetes. .. 31

Figura 1.6 Disponibilidad de herramientas al momento de querer utilizarlas. 32

Figura 1.7 Solución eficaz a problemas. 33

Figura 1.8 Nivel de soporte técnico de los administradores de los laboratorios. 34

Figura 1.9 Componentes básicos del sistema (1) 35

Figura 2.1 Capas del sistema a Implementar 39

Figura 2.2 Estructura de directorios a implementar. 43

Figura 2.3 Procesos involucrados en la construcción del sistema básico 46

Figura 2.4 Subprocesos involucrados en los preparativos para la construcción. 47

Figura 2.5 Subprocesos involucrados en la construcción del sistema temporal..... 50

Figura 2.6 Subprocesos involucrados en la construcción del sistema básico 56

Figura 2.7 Ficheros en donde se montan los ficheros virtuales 57

Figura 2.8 Procesos involucrados en la instalación de herramientas requeridas y complementarias 100

Figura 2.9 Captura de /etc/profile 106

Figura 2.10 Captura de /etc/bashrc 107

Figura 2.11 Captura de ~/.bashrc 107

Figura 2.12 Captura de ~/.bash_profile 108

Figura 2.13 Procesos involucrados en la creación del LiveCD..... 118

Figura 2.14 Captura de la creación del LiveCD 122
Figura 2.15 Captura de finalización del proceso de creación del live CD. 122
Figura 2.16 Captura del arranque del LiveCD 123
Figura 2.17 Captura del arranque de los módulos del kernel del LiveCd 123

ÍNDICE DE TABLAS

Tabla 1.1 Parámetros utilizados para definir el tamaño de la muestra 13
Tabla 1.2 Asignación de valores para los parámetros de la muestra 13
Tabla 1.3 Estructura de la encuesta 18
Tabla 1.4 Hardware del laboratorio de redes 29
Tabla 1.5 Software pedido por materia 36
Tabla 2.1 Herramientas para ejecutar bancos de pruebas. 53
Tabla 2.2 Herramientas para brindar mayor funcionalidad. 55
Tabla 2.3 Estructura de las tablas para mostrar la información de los paquetes 60
Tabla 2.4 Características del paquete Man-pages 61
Tabla 2.5 Características del paquete GlibC 61
Tabla 2.6 Características del paquete Binutils 62
Tabla 2.7 Características del paquete GCC 63
Tabla 2.8 Características del paquete Berkeley DB 64
Tabla 2.9 Características del paquete Sed 64
Tabla 2.10 Características del paquete Coreutils 65
Tabla 2.11 Características del paquete M4 66
Tabla 2.12 Características del paquete Bison 66
Tabla 2.13 Características del paquete Ncurses 67
Tabla 2.14 Características del paquete Procps 68
Tabla 2.15 Características del paquete Libtool 69
Tabla 2.16 Características del paquete Perl 69
Tabla 2.17 Características del paquete Readline 70
Tabla 2.18 Características del paquete Zlib 70

Tabla 2.19 Características del paquete Autoconf 71
Tabla 2.20 Características del paquete Automake 71
Tabla 2.21 Características del paquete Bash 72
Tabla 2.22 Características del paquete Bzip2 72
Tabla 2.23 Características del paquete Diffutils 73
Tabla 2.24 Características del paquete File 74
Tabla 2.25 Características del paquete Findutils 74
Tabla 2.26 Características del paquete Flex 75
Tabla 2.27 Características del paquete GRUB 75
Tabla 2.28 Características del paquete Gawk 76
Tabla 2.29 Características del paquete Gettext 76
Tabla 2.30 Características del paquete Grep 77
Tabla 2.31 Características del paquete Groff 77
Tabla 2.32 Características del paquete Gzip 78
Tabla 2.33 Características del paquete Inetutils 79
Tabla 2.34 Características del paquete kbd 79
Tabla 2.35 Características del paquete Make 79
Tabla 2.36 Características del paquete Man-DB 80
Tabla 2.37 Características del paquete Module-Init-Tools 81
Tabla 2.38 Características del paquete Patch 82
Tabla 2.39 Características del paquete Psmisc 82
Tabla 2.40 Características del paquete Shadow 84
Tabla 2.41 Características del paquete Sysklogd 84
Tabla 2.42 Características del paquete Sysvinit 85
Tabla 2.43 Características del paquete Tar 86

Tabla 2.44 Características del paquete Texinfo 86
Tabla 2.45 Características del paquete Udev..... 88
Tabla 2.46 Características del paquete Util-linux 89
Tabla 2.47 Características del paquete Vim..... 89
Tabla 2.48 Niveles de ejecución de SysVinit..... 91
Tabla 2.49 Características del paquete Autofs..... 101
Tabla 2.50 Características del paquete PCI Utilities 102
Tabla 2.51 Características del paquete Usbutils 102
Tabla 2.52 Características del paquete Dhcpd..... 103
Tabla 2.53 Características del paquete Wget..... 103
Tabla 2.54 Características del paquete Netsnmp..... 110
Tabla 2.55 Características del paquete Traceroute..... 111
Tabla 2.56 Características del paquete Wireshark..... 112
Tabla 2.57 Características del paquete Fping..... 112
Tabla 2.58 Características del paquete Nmap 113
Tabla 2.59 Características del paquete Nmap 113
Tabla 2.60 Características del paquete Apache..... 114
Tabla 2.61 Características del paquete ProFTPD..... 114
Tabla 2.62 Características del paquete Bind..... 115
Tabla 2.63 Características del paquete Minicom..... 115
Tabla 2.64 Características del paquete OpenSSL 116
Tabla 2.65 Características del paquete MIT Kerberos 116
Tabla 2.66 Características del paquete IPSec 117
Tabla 2.67 Características del paquete OpenLDAP..... 117
Tabla 2.68 Características del paquete Heartbeat 118

Tabla 3.1 Estructura de las tablas para mostrar el procedimiento de pruebas..... 125
Tabla 3.2 Prueba, Espacio que ocupa el sistema 127
Tabla 3.4 Prueba, Tiempo de arranque del sistema..... 128
Tabla 3.5 Prueba, Compatibilidad con interfaces 131
Tabla 3.6 Prueba, Compatibilidad con procesadores..... 132
Tabla 3.7 Prueba, Compatibilidad con monitores..... 134
Tabla 3.8 Prueba, Net-snmp 136
Tabla 3.9 Prueba, Traceroute..... 138
Tabla 3.10 Prueba, Wireshark..... 142
Tabla 3.11 Prueba, Fping 143
Tabla 3.12 Prueba, Nmap 146
Tabla 3.12 Prueba, Java 147
Tabla 3.13 Prueba, Apache..... 150
Tabla 3.14 Prueba, FTP 152
Tabla 3.15 Prueba, DNS 153
Tabla 3.16 Prueba, Minicom..... 155
Tabla 3.17 Prueba, OpenSSL 157
Tabla 3.18 Prueba, Kerberos 158
Tabla 3.19 Prueba, IPSec 158
Tabla 3.20 Prueba, OpenLDAP..... 160
Tabla 3.21 Prueba, Heartbeat 161

PRESENTACIÓN

Se construirá una distribución de tipo Live CD de GNU/Linux adaptada a las necesidades de los usuarios de los laboratorios. El proyecto analizará la situación actual de los laboratorios de la carrera. Se estudiará la gestión de los servicios de TI respecto al sistema operativo Linux en uso. Se analizarán los requerimientos de los usuarios de los laboratorios. Ésta distribución no incluirá herramientas que no van a ser utilizadas. El objetivo del proyecto es optimizar el uso del tiempo de las clases prácticas y el uso de recursos de hardware del laboratorio.

El sistema operativo GNU/Linux es utilizado a lo largo de toda la carrera de Electrónica y Redes de Información de la EPN. En el mercado existen muchas distribuciones de GNU/Linux, pero ninguna de ellas provee todo el software necesario para las clases prácticas. Además estas distribuciones traen consigo una gran cantidad de paquetes y herramientas que no se utilizan.

El proyecto planteado pretende evitar consumo innecesario de recursos importantes como: espacio en disco duro, memoria RAM, tiempo de instalación y tiempo de arranque del sistema operativo. Además de traer consigo la mayoría de herramientas necesarias para las clases prácticas. Con esto se evita la espera de la instalación correcta de cada paquete necesario por parte de los administradores de laboratorio o de los mismos usuarios de los laboratorios. El fin es aprovechar todas las horas clase en el laboratorio, lo que es la principal preocupación de profesores y estudiantes de la carrera.

El proyecto también pretende evidenciar la forma en que se gestionan los servicios de TI en los laboratorios. En la actualidad no se tiene conocimiento de la forma en que se gestionan los mismos respecto al sistema operativo Linux en uso. No se conocen procedimientos de gestión de problemas, gestión de cambios, gestión de versiones, gestión de configuraciones, gestión de capacidades entre otros.

El presente proyecto ayudará al lector a conocer a profundidad todos los procesos involucrados en implantar una distribución GNU/Linux partiendo de código fuente.

RESUMEN

El presente proyecto está compuesto básicamente de tres partes: un estudio de análisis de requerimientos y de la situación actual de los laboratorios de la carrera respecto al sistema operativo basado en Linux. Luego se describen todos los procesos, subprocesos y tareas involucrados en la implantación del sistema. Finalmente se realizan las pruebas para verificar el funcionamiento correcto del sistema. A continuación se describe brevemente lo realizado en cada capítulo.

El capítulo 1 contiene el marco teórico respecto a los componentes principales de un sistema GNU/Linux. Consta además el análisis de la problemática actual de los laboratorios. También contiene una descripción de la metodología utilizada a lo largo de todo el proyecto y los análisis de situación actual y de requerimientos con sus respectivos resultados.

En el capítulo 2 se describen todos los procesos seguidos para la construcción del sistema y las utilidades que tiene el mismo. Además se incluye el proceso seguido para la construcción del entregable final: la distribución LiveCD basada en GNU/Linux, TESIX-0.9.

En el capítulo 3 se incluyen todas las pruebas realizadas divididas en 2 grupos: pruebas para verificar que se cumpla el alcance del proyecto y pruebas para verificar que se cumplan los requerimientos funcionales. Esto es el correcto funcionamiento de todas las herramientas requeridas instaladas como se planteó en el alcance del proyecto.

En el capítulo 4 se incluyen las conclusiones y las recomendaciones luego de la terminación del proyecto. Por último la bibliografía utilizada y los anexos constarán en el capítulo 5.

CAPITULO 1

MARCO TEÓRICO, PROBLEMÁTICA, ANÁLISIS DE LA SITUACIÓN ACTUAL Y ANÁLISIS DE REQUERIMIENTOS.

Este capítulo define el proyecto "Distribución GNU/Linux para uso en los laboratorios de la carrera de Electrónica y Redes de Información", para la realización de este proyecto se describe el marco teórico acerca de los componentes que conforman el sistema GNU/Linux, además el capítulo presenta el problema que existe en el laboratorio de la carrera con el uso del sistema Linux. Posteriormente se describe el procedimiento a seguir para obtener los datos necesarios para construir la distribución GNU/Linux. Para finalizar el capítulo se realiza el análisis de la situación actual, basado en la metodología ITIL[1], y el análisis de requerimientos del sistema a ser creado.

1.1 COMPONENTES PRINCIPALES DE UN SISTEMA OPERATIVO BASADO EN GNU/LINUX [1] [2]

Esta sección describe los componentes básicos de los sistemas operativos basados en GNU/Linux. También explica la funcionalidad de cada parte del sistema.

La distribución creada en el proyecto es una distribución GNU/Linux. Esto es: una distribución de software basada en el núcleo Linux, bajo licencia GPL[2] de GNU. En esta distribución se incluyen paquetes de software que satisfacen las necesidades de los usuarios de los laboratorios de computación de la carrera de Redes. Dentro de esta distribución GNU/Linux, para cumplir el licenciamiento GPL se incorporan

[1] ITIL: Librería de Infraestructura de Tecnologías de Información
[2] GPL: Licencia Pública General, está orientada a proteger la libre distribución, modificación y uso del software.

aplicaciones o controladores que son libres. Se entiende por “libre” como de libre ejecución, copia, modificación y distribución.

El sistema operativo GNU/Linux se compone de tres partes principales que son:

A. El núcleo o kernel Linux.
B. Las bibliotecas del sistema.
C. Los utilitarios del sistema y del usuario.

Figura 1.1 Componentes de un sistema operativo basado en Linux[3]

1.1.1 KERNEL LINUX

El kernel es el componente principal del sistema operativo. Proporciona toda la funcionalidad para ejecutar procesos y servicios del sistema, como por ejemplo: gestión de memoria, gestión de archivos, manejo de usuarios, gestión de los procesos, etc.

[3] Fuente: Sistemas Operativos, Galvin Silverschatz

Dentro del kernel se contemplan dos niveles de ejecución de los procesos. Modo usuario y modo privilegiado o *root*. Cuando se ejecuta la llamada del núcleo, se está ejecutando en modo privilegiado. Cuando se ejecuta código que no pertenece al núcleo, se está ejecutando en modo usuario. Al poseer dos niveles diferentes de ejecución se garantiza que a nivel de hardware ningún programa de usuario pueda acceder a recursos generales del sistema, tampoco interactuar directamente con otros procesos.

El objetivo de poseer niveles de ejecución es garantizar que programas maliciosos o mal diseñados no perjudiquen el sistema u otros procesos. Cada proceso tiene su propia zona de memoria y no puede acceder fuera de ella de manera incidental.

1.1.2 BIBLIOTECAS DEL SISTEMA [3]

Las bibliotecas del sistema contienen las funciones para realizar las “llamadas al sistema”, a través de la cuales las aplicaciones pueden interactuar con el núcleo. Todas las funciones necesarias para la ejecución de aplicaciones se implementan en las bibliotecas del sistema. Las aplicaciones efectúan llamadas a las bibliotecas del sistema para obtener un servicio.

Las bibliotecas también ofrecen rutinas que no corresponden a “llamadas al sistema”. Por ejemplo: algoritmos de ordenación, funciones matemáticas y rutinas de manipulación de cadenas.

1.1.3 UTILITARIOS DEL SISTEMA Y DEL USUARIO [1]

Los utilitarios del sistema constituyen las herramientas que el usuario espera encontrar en un sistema completo. Algunos utilitarios del sistema podrían invocarse una sola vez para asignar valores iniciales y configurar algún aspecto del sistema. En contraste, los demonios podrían ejecutarse en el sistema de forma permanente,

hasta que el sistema o el demonio sean parados o desactivados. Por ejemplo: responder a conexiones entrantes de red.

Los utilitarios de usuario son aquellos que realizan tareas habituales dentro del sistema. Los utilitarios de usuario realizan tareas como: listar los directorios, mover archivos, eliminar archivos, etc. Algunos utilitarios pueden desempeñar funciones más complejas como: procesamiento de texto, ordenación de datos, etc.

1.2 IDENTIFICACIÓN DEL PROBLEMA

En esta sección se identifica el problema que existe en los laboratorios de computación de la carrera de Redes en cuanto al uso del sistema Linux en las clases prácticas que demandan su uso. Para identificar el problema se analizan dos factores que se detallan a continuación:

1.2.1 APROVECHAMIENTO DE HORAS CLASE

El aprovechamiento de horas clase en el laboratorio es la principal preocupación de docentes y alumnos. En la actualidad se pierden horas clase debido a no tener disponibles las herramientas necesarias para el desarrollo de las mismas. Ninguna distribución Linux existente en el mercado trae consigo todas las herramientas requeridas para las clases prácticas.

En el proceso de petición de una nueva herramienta se invierte mucho tiempo hasta la instalación de la misma en todas las computadoras de los laboratorios. Además existen casos en que la implementación de nuevas herramientas corre a cuenta de los docentes y estudiantes, retardando más el inicio de las clases.

1.2.2 DISPONIBILIDAD Y FUNCIONALIDAD DE HERRAMIENTAS

Dependiendo de la distribución Linux en uso se tiene un mayor o menor grado de disponibilidad de las herramientas necesarias. Esto genera pérdida de tiempo para los administradores en la búsqueda de versiones compatibles de las herramientas solicitadas por los docentes, y posteriormente en su instalación.

Existe también el riesgo de corromper todo el sistema Linux al momento de instalar herramientas incompatibles con el mismo. Este proceso puede desencadenar en tener que reinstalar todo el sistema operativo.

Para los administradores resulta complicado comprobar si las herramientas pedidas funcionan correctamente en cada computadora en la que se instalaron. Para ello es necesario que los docentes y estudiantes corroboren el funcionamiento de las mismas. Caso contrario, es necesario reinstalar la o las herramientas pedidas.

1.3 METODOLOGÍA PARA EL DESARROLLO DEL PROYECTO

La metodología para el desarrollo del proyecto considera cinco etapas. A continuación se enumeran y describen las etapas a seguir:

1.- Análisis de la situación actual de los laboratorios.
2.- Análisis de requerimientos del sistema.
3.- Diseño del sistema
4.- Construcción del sistema.
5.- Pruebas del sistema.

1.3.1 ANÁLISIS DE LA SITUACIÓN ACTUAL DE LOS LABORATORIOS.

Esta etapa contempla la realización de entrevistas. Las entrevistas se realizan a las personas encargadas del manejo de los laboratorios y a los docentes que hacen uso del sistema operativo Linux en los mismos.

Por medio de las entrevistas se analiza la manera en que se gestionan los servicios de TI respecto del sistema operativo Linux en uso. Para este propósito se utiliza la metodología de ITIL. Esta es la metodología de las mejores prácticas para gestión de servicios de tecnologías de información. La metodología ITIL será aplicada en todos sus puntos.

Con la metodología ITIL se pretende evidenciar la forma en que se controlan, operan y administran actualmente los recursos. No se llega a modelar una metodología para la gestión de servicios de TI en los laboratorios, ya que esto no está contemplado dentro del alcance del presente proyecto.

1.3.2 ANÁLISIS DE LOS REQUERIMIENTOS DEL SISTEMA.

Para el análisis de requerimientos del sistema se realizan encuestas a los profesores que utilizan Linux para impartir sus clases en los laboratorios de la carrera. También se realizan encuestas a una muestra significativa de los estudiantes de la carrera de Redes.

El objetivo de la encuesta es obtener los datos necesarios para la construcción del sistema. Para ello se recaba la información acerca del hardware sobre el cual tiene que ejecutarse el sistema. En la encuesta también se obtiene información acerca de los paquetes y utilitarios necesarios para las clases prácticas.

1.3.3 DISEÑO DEL SISTEMA

El diseño del sistema se basa en los datos obtenidos de las encuestas y entrevistas de las secciones 1.2.1 y 1.2.2. En el diseño se describen: las características del núcleo a implantar, las bibliotecas básicas del sistema, y las capas que conforman el sistema.

Además, en el diseño se presentan la estructura de directorios, con sus respectivos permisos y la metodología que se usa para instalar los paquetes de software.

1.3.4 CONSTRUCCIÓN DEL SISTEMA.

Una vez realizados los análisis de: situación actual y de requerimientos, se obtiene la información de qué utilitarios de sistema y de usuario serán parte del sistema operativo. Para ello se utiliza la sección correspondiente al diseño del sistema como una guía de desarrollo.

La construcción del sistema se realiza sobre un entorno de máquina virtual. La máquina virtual usada para el fin mencionado es VMware®. La utilización del entorno de máquina virtual se justifica debido a la flexibilidad de poder pausar el proceso de creación del sistema en cualquier instante.

Luego de configurar el hardware de la máquina virtual se procede al desarrollo del sistema GNU/Linux. Como punto de partida se utiliza software base que provea un núcleo, compiladores, librerías y utilitarios. Una distribución anfitriona provee el software base necesario para la creación del sistema.

En último lugar, se crea el LiveCD[4] de la distribución construida que es el entregable del proyecto.

1.3.5 PRUEBAS DEL SISTEMA.

Esta sección contempla las pruebas para verificar que se cumplan los requerimientos funcionales obtenidos de las encuestas de análisis de requerimientos. Para ello se comprobará el correcto funcionamiento de las herramientas que se incluyen en el sistema a por cada materia que utiliza Linux.

Además se realizan pruebas para corroborar el cumplimiento del alcance del proyecto. Esto es verificar la compatibilidad con los dispositivos de hardware para evitar problemas con sus manejadores. La realización de las pruebas, así como sus resultados, se detallan en el capítulo 3.

1.4 DISEÑO DE LOS CUESTIONARIOS

En la presente sección se describe el diseño de la entrevista, que sirve para el análisis de la situación actual. La encuesta, por otro lado, es la herramienta para el análisis de requerimientos.

La gestión de los servicios de TI en los laboratorios se divide en dos mundos: el del sistema operativo Windows y el del sistema Linux. Esto debido a que se gestiona de manera diferente las configuraciones, versiones, seguridad y problemas del sistema.

La gestión de configuraciones y versiones está directamente relacionada en el mundo Windows. Luego de requerirse un nuevo programa por parte de los docentes se tiene una nueva configuración de software del sistema así como una nueva

[4] LiveCD: es una distribución basada en Linux que funciona sin instalarse en un computador.

versión del mismo. Esta versión es protegida con el programa Deep Freeze. El programa permite que el sistema siempre arranque a partir de una imagen congelada anteriormente. Se crea una imagen inalterable del sistema que sólo es cambiada por los administradores si un nuevo paquete es requerido. En el mundo Linux no se realiza gestión alguna de configuraciones y versiones.

A nivel de seguridad, en Windows se utiliza un antivirus y también con la ayuda del programa Deep Freeze se trata de evitar el ingreso de virus al sistema. En Linux no se utiliza ningún antivirus o programa para evitar que código malicioso dañe el sistema.

En cuanto a la gestión de problemas, por ejemplo: en Linux se tiene mayor probabilidad de que un programa o un parche queden mal instalados sin que los administradores se den cuenta. Esto debido a que en muchas ocasiones se tiene que trabajar con el código fuente del programa lo que trae más dificultad en la instalación. En Windows es muy fácil darse cuenta que una herramienta quede mal instalada debido a los mensajes gráficos que muestra el sistema.

Debido a las razones expuestas, y a que el proyecto tiene alcance sólo en el mundo Linux, todos los cuestionarios y el estudio a realizarse se limitarán al entorno Linux.

El primer cuestionario destinado sólo a los administradores de los laboratorios es de tipo *entrevista*. Aquí se utilizan los parámetros de la metodología ITIL. La entrevista sirve para conocer la manera en la que se están gestionando los servicios de TI respecto del sistema operativo Linux en uso.

El segundo cuestionario es del tipo *encuesta*. Ésta se enfoca a los profesores que usan Linux en las clases prácticas y a estudiantes de la Carrera. La encuesta contribuye en los análisis de situación actual y de requerimientos. La encuesta sirve

para conocer los requerimientos de software de los usuarios del sistema Linux de los laboratorios. Para la realización de encuesta se calcula el tamaño de la muestra.

1.4.1 ENTREVISTA DE LA GESTIÓN DE SERVICIOS DE TI EN LOS LABORATORIOS

En esta sección se describen: objetivo general, objetivos específicos, metodología a utilizar en la entrevista.

1.4.1.1 Objetivos

Objetivo general

Conocer la forma en la que se gestionan los servicios de TI basados en la metodología ITIL, del sistema operativo Linux en los laboratorios de la carrera de Ingeniería en Electrónica y Redes de Información.

Objetivos específicos: [9]

- Analizar el soporte brindado para los servicios de TI del sistema operativo Linux en los laboratorios de la carrera de Ingeniería en Electrónica y Redes de Información.
- Estudiar la provisión de servicios de TI del sistema operativo Linux en los laboratorios de la carrera de Ingeniería en Electrónica y Redes de Información.

1.4.1.2 Metodología de la entrevista

Para esta entrevista se utilizan preguntas de opción múltiple. En el caso de necesitarse una explicación más amplia de la respuesta se deja el espacio pertinente

para hacerlo. Las preguntas se basan en todos los módulos de la metodología ITIL. Para obtener mayor información de la metodología ITIL visite: http://itil.osiatis.es

Además de realizar preguntas en la entrevista acerca de los módulos de ITIL se aprovecha para obtener información que no se basa en ITIL. Necesariamente se debe tener conocimiento de esta información para la realización del proyecto. Un ejemplo es: La información de configuración actual del hardware de las computadoras de los laboratorios. Con la pregunta expuesta en la entrevista se pretende evitar problemas posteriores al desarrollo con manejadores de hardware.

También incluyen preguntas acerca de la distribución Linux utilizada actualmente, con lo que se pretende evidenciar por qué es utilizada la misma.

Para ver la guía de la entrevista diríjase al Anexo A.

1.4.2 ENCUESTA DE LA SITUACIÓN ACTUAL Y DE REQUERIMIENTOS

1.4.2.1 Objetivos

Aquí se presentan tanto el objetivo general, como objetivos específicos correspondientes a la encuesta de la situación actual y de requerimientos.

Objetivo General

Determinar por medio de un sondeo a profesores y estudiantes que hagan uso del sistema operativo Linux en los laboratorios, el software básico y los paquetes requeridos con los que contará la nueva distribución.

Objetivos Específicos:

- Determinar los componentes básicos del sistema, en base a los paquetes requeridos por profesores y estudiantes.
- Evaluar y evidenciar la manera en que los profesores y estudiantes perciben el nivel de servicio ofrecido en los laboratorios.

1.4.2.2 Metodología de la encuesta [10] [20]

La metodología a usar para formular la encuesta se basa en una técnica descriptiva y cuantitativa. Consiste en una investigación realizada con preguntas abiertas, cerradas, y de valoración .Sirve principalmente para obtener los requerimientos funcionales del sistema. Estos son: todos aquellos paquetes de software indispensables para el funcionamiento del sistema GNU/Linux. Incluyendo las herramientas requeridas por los docentes para sus materias.

Para el desarrollo de la encuesta se constituyen como actividades: establecer la muestra, realizar la encuesta en el campo, y levantar la información recopilada con la misma.

El universo de la población de estudio se compone de 531 estudiantes matriculados en la carrera en el período 2010-2. Además lo constituyen 5 profesores, que dictan las materias en las que se utiliza el sistema operativo basado en Linux. Estos datos fueron facilitados por la secretaría de la carrera y los administradores de los laboratorios.

Sólo se considera a un tipo de usuarios del nuevo sistema: Usuarios Técnicos. Tanto estudiantes y docentes tienen conocimiento del funcionamiento de un sistema basado en Linux.

Para definir el tamaño de la muestra se aplica la siguiente ecuación:

$$n = \frac{k_\alpha{}^2 p * q * N}{\left(e^2 * (N-1)\right) + k_\alpha{}^2 p * q}$$

Parámetro	Significado
n	Tamaño de la muestra.
N	Tamaño total de la población.
k_α	Constante dependiente del nivel de confianza asignado
α	Nivel de confianza, Indica la probabilidad (α) de que los resultados sean ciertos.
e	Error muestral deseado, es la diferencia que puede haber en el resultado obtenido de preguntar a la muestra de la población y el resultado obtenido si se preguntaría a toda la población.
p	Proporción de individuos dentro de la población que poseen la característica de estudio. Este dato es generalmente desconocido y se suele suponer que p=q=0.5 que es la opción más segura.
q	Proporción de individuos dentro de la población que no poseen la característica de estudio.

Tabla 1.1 Parámetros utilizados para definir el tamaño de la muestra

Asignando un porcentaje de confianza del 95%, un porcentaje de error del 10% y como variabilidad p y q de 0.5% se obtiene:

Usuarios	N	k_α	α	e	p	q	n
Docentes y Estudiantes	536	1,96	95%	10%	0,5	0,5	82

Tabla 1.2 Asignación de valores para los parámetros de la muestra

De lo anteriormente expuesto, se concluye que se debe realizar 82 encuestas, 5 a profesores y 77 a estudiantes.

1.4.2.3 Estructura de la encuesta

La encuesta consta de tres secciones que son aplicadas en su totalidad a los profesores. No todas las secciones son aplicadas a los estudiantes debido a que contienen preguntas específicas para docentes. A continuación se detallan las secciones:

La primera sección contiene información general acerca de las materias que dicta el docente y que utilizan el sistema Linux. La lista de materias que requieren el uso de Linux para sus clases prácticas fue proporcionada por los administradores de los laboratorios. Además en esta sección se pretende conocer el tiempo aproximado que cada profesor utiliza el sistema Linux a lo largo de la semana. Es evidente que esta sección no es aplicable a estudiantes.

La segunda sección analiza la percepción de los docentes frente al servicio que reciben por parte de los administradores. La evaluación se la realiza de una forma cuantitativa. Esta sección de la encuesta también será aplicada a estudiantes de la carrera (número definido en el tamaño de la muestra).Con esta sección se tendrá una percepción más completa del nivel de servicio ofrecido.

Finalmente la tercera sección se enfoca en los requerimientos funcionales del sistema. Cuando se pregunta acerca de los componentes básicos del mismo se agrupan los programas según su funcionalidad. Resulta complicado agruparlos según la metodología utilizada para construir el sistema. Existen paquetes distintos que instalan herramientas con una funcionalidad similar o complementaria. Esta sección también se aplica a estudiantes para tener una idea qué componentes son

considerados por los estudiantes como parte básica del sistema operativo basado en Linux.

Para ejemplificar lo mencionado en el párrafo anterior, se plantea lo siguiente: la compresión, descompresión y otras operaciones con archivos son posibles gracias a la instalación de paquetes como: *tar*, y otras herramientas como *mv*, *rm*, etc. Estas son instaladas por el paquete *Coreutils.* Todas estas herramientas se las puede agrupar para el propósito de la encuesta como: programas para manipular directorios y ficheros. De esta manera la encuesta será más sencilla para el entendimiento de todos los encuestados.

La última pregunta recopila los programas que cada docente requiere para impartir su cátedra. Se especifica también el tiempo aproximado en que se utiliza el programa a lo largo del semestre.

En la siguiente tabla se muestra un esquema de las preguntas que incluye la encuesta. En el esquema se muestra el tipo de información que se desea obtener, el tipo de usuario al que va enfocada la pregunta, y la pregunta en si.

Tipo de información: Información General	
Tipo de usuario	***Pregunta***
Docentes	1.-. ¿Qué materias dictadas por usted requieren del sistema operativo Linux en los laboratorios? ☐ Programación ☐ Sistemas Operativos. ☐ Prog. Orientada a Objetos ☐ Prog con Herram. Visuales ☐ Bases de Datos.

	☐ Aplicaciones Distribuidas. ☐ Seguridad en Redes. ☐ Gestión de Redes. ☐ Redes e Intranets. ☐ WLAN. ☐ Hardware de Conectividad ☐ Evaluación de Redes.
Docentes	2.- ¿Aproximadamente cuántas horas por semana utiliza el sistema Linux? ☐ De 1 a 3 horas. ☐ De 4 a 6 horas. ☐ Más de 7 horas.
Tipo de información: Percepción del Nivel de Servicio	
Tipo de usuario	***Pregunta***
Docentes y Estudiantes	3.- Tiempo de respuesta a solicitudes de instalación de nuevos paquetes. ☐1 ☐2 ☐3 ☐4 ☐5 4.- Disponibilidad de herramientas al momento de querer usarlas. ☐1 ☐2 ☐3 ☐4 ☐5 5.- Solución eficaz por parte de los administradores a problemas suscitados en los laboratorios.

	☐1 ☐2 ☐3 ☐4 ☐5 6.- Nivel de soporte técnico ofrecido por parte de la administración del laboratorio. ☐1 ☐2 ☐3 ☐4 ☐5
Tipo de Información: Requerimientos FuncionaleS	
Tipo de usuario	***Pregunta***
Docentes y Estudiantes	7.- ¿Cuáles son los componentes BÁSICOS que considera debe tener un sistema operativo basado en Linux? ☐ Shell (intérprete de comandos). ☐ Programas para manipular directorios y ficheros (tar, bzip2, cat, mv, mkdir, etc.). ☐ Programas para administración de permisos (chroot, chmod, chown, etc). ☐ Programas para búsqueda en disco (find, locate, etc.). ☐ Programas para monitorizar el sistema y sus procesos (df, top, uptime, etc.). ☐ Programas para manejo de red (ifconfig, ping, telnet, ftp, ip, etc). ☐ Editor de texto en modo consola (vim). ☐ Interfaz gráfica amigable (KDE o Gnome). ☐Otros (especifique):_____

<table>
<tr>
<td>Docentes</td>
<td>8.- Especificar los programas ESTRICTAMENTENECESARIOS para impartir sus clases de laboratorio.

▪ En la columna “Paquete” indique el nombre del paquete de software que usted requiere.
▪ En la columna de “Uso” indique si el paquete va a ser utilizado sólo durante una clase específica (alrededor de una semana), o a lo largo de todo el semestre.
▪ En la columna “Materia” indique el nombre de la materia en la que se va a utilizar el paquete (llene la columna en caso de que usted imparta más de una materia)

<table>
<tr><th>Paquete</th><th>Uso</th><th>Materia</th></tr>
<tr><td></td><td>1 clase ☐ Todo el semestre ☐</td><td></td></tr>
</table>
</td>
</tr>
</table>

Tabla 1.3 Estructura de la encuesta

Para ver la encuesta aplicada diríjase al Anexo B.

1.5 ANÁLISIS DE RESULTADOS DE LOS CUESTIONARIOS

Una vez realizadas las entrevistas y encuestas se procede a analizar las respuestas de las personas que colaboraron con las mismas a continuación:

1.5.1 ANÁLISIS DE RESULTADOS DE LA ENTREVISTA DE LA GESTIÓN DE SERVICIOS DE TI EN LOS LABORATORIOS

En la sección 1.5.1.1 se analizan los resultados obtenidos, divididos por cada módulo de ITIL. Para esto, primero se expone y analiza la forma en que se gestiona cada módulo que describe ITIL por parte de los administradores de los laboratorios.

Seguidamente, se obtiene una conclusión, o recomendación, a partir de la descripción y análisis previos.

Después de analizar los módulos de ITIL, se exponen los resultados de información necesaria para la realización del proyecto. Esta información se la obtiene de la misma entrevista, debido a que debe ser proporcionada por los mismos administradores. La información se refiere a la configuración de hardware actual de las computadoras de los laboratorios.

La encuesta se basó en la guía diseñada en la sección anterior y se encuentra en el Anexo A. A continuación se presentan los resultados de las entrevistas:

1.5.1.1 MÓDULOS DE LA METODOLOGÍA ITIL [9]

1.5.1.1.1 Soporte al servicio

.En esta sección se analizan los aspectos que garantizan: continuidad, disponibilidad y calidad de servicio que se presta a los usuarios del sistema Linux en los laboratorios.

En la figura 1.2 se observan los módulos de ITIL que son parte del soporte al servicio.

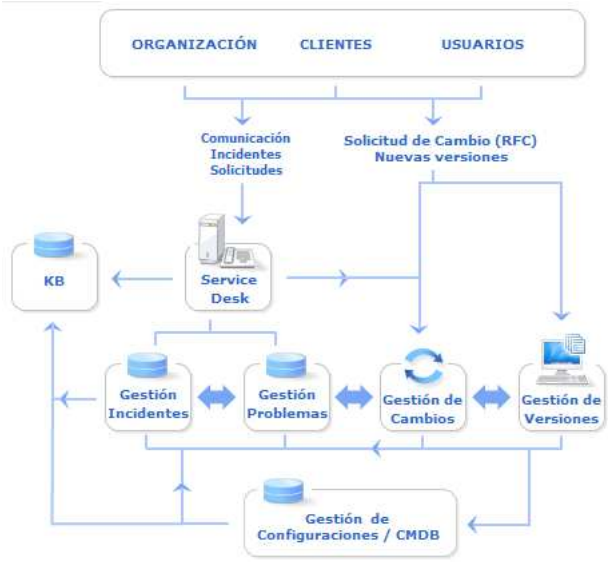


Figura 1.2 Metodología de soporte al servicio.[5]

Los ayudantes que se encargan de los laboratorios actúan como la Mesa de Servicios (Service Desk). Ellos recogen todos los pedidos, brindan soporte y proveen de los servicios requeridos. Estos pedidos no son almacenados en una base de conocimientos (Knowledge Base) que permitiría brindar un soporte más ágil a los usuarios.

- **Gestión de incidentes**

En los laboratorios una vez que se notifica un incidente con el sistema operativo Linux las personas encargadas del Help Desk no pueden resolverlo inmediatamente. Los ayudantes deben esperar que termine la hora de clases. En muchas ocasiones deben esperar a que terminen todas las sesiones para poder resolver el incidente.

Los incidentes se solucionan generalmente en menos de un día laboral. En casos excepcionales son resueltos en el lapso de una semana.

[5] Fuente: http://itil.osiatis.es

Es imposible resolver un incidente inmediatamente. No se puede interrumpir una sesión de laboratorio. Tampoco se puede dejar con menos computadoras al mismo. Por estas razones, el tiempo empleado en resolver eficazmente los problemas se considera el apropiado.

- **Gestión de problemas**

En caso que se haya o no resuelto el problema, los administradores investigan lo más rápido posible las causas que generan el problema. No se tiene un registro formal de los problemas suscitados. Tampoco se cuenta con algún procedimiento de prevención de problemas.

Una vez solventado el problema se monitorea el cambio realizado. Para ello se realiza la comprobación del correcto funcionamiento del servicio brindado. Existen ocasiones en que lo único que se hace es realizar los cambios para solventar el problema sin comprobaciones posteriores, a menos que los administradores reciban una queja por parte de los usuarios.

La administración de los laboratorios debería llevar un registro formal de los problemas suscitados. También se deberían registrar los procedimientos realizados para resolver dichos problemas. A futuro de presentarse el mismo incidente se lo resolvería de una manera más eficaz. También se debería monitorear el correcto funcionamiento de todos y cada uno de los cambios realizados.

- **Gestión de cambios**

Actualmente no se posee un procedimiento formal para la petición de nuevos servicios y/o paquetes de software. Estos son solicitados de forma verbal por parte de los profesores hacia los administradores o por medio de un correo electrónico al responsable del laboratorio.

Los cambios realizados dentro del sistema Linux no son registrados ni tampoco son siempre evaluados. Debido a que es una ambiente de laboratorio cualquier usuario del mismo puede realizar los cambios que desee sirvan o no para sus clases al poseer la contraseña de administrador del sistema.

La petición de nuevos servicios y/o paquetes debería llevar un registro formal. Con esto se controlaría de mejor manera el sistema. Además se mediría el tiempo exacto en el que se responden a las peticiones para una mayor satisfacción de los usuarios. Finalmente los administradores y profesores tendrían un respaldo en el cual sustentarse en caso de existir reclamos.

- **Gestión de configuraciones**

En los laboratorios se lleva un registro semestral de la configuración de hardware. Esa información es provista por el responsable de los mismos a los administradores. A nivel de software una vez instalado el sistema operativo no se lleva ningún registro formal de las configuraciones.

No se realizan auditorias periódicas de hardware ni de software. A nivel de hardware lo único que se hace es revisar que todos los componentes externos se encuentren presentes al finalizar cada sesión de laboratorio. No existe un cronograma para la realización de auditorías.

Convendría llevar un registro formal de las configuraciones de software de la máquina. Esto serviría para controlar que todas las máquinas tengan la misma configuración dentro del laboratorio y que no se desperdicien recursos de hardware en programas innecesarios. Se necesitaría tener un cronograma para auditar hardware y software. El fin es verificar todo lo que conste en los inventarios, y

eliminar software instalado no requerido. También se podría determinar si existen irregularidades en el manejo de los recursos de la infraestructura de TI.

- **Gestión de versiones**

No se respaldan archivos de configuración del sistema operativo. Tampoco se respaldan archivos de configuración de las herramientas instaladas. En caso de presentarse un problema crítico con el sistema operativo se reinstala todo el sistema Linux.

Actualmente los administradores manejan como política, el no responsabilizarse por documentos o archivos de configuración de las herramientas instaladas. Toda la responsabilidad se la deja al usuario.

En los casos en que no se encuentra instalado el sistema nativamente se realizan respaldos de las máquinas virtuales. De presentarse una falla crítica se recuperan los respaldos de las imágenes de las máquinas virtuales.

No es posible volver a versiones anteriores (configuraciones anteriores).No se crean imágenes de las particiones de Linux periódicamente. La responsabilidad recae en el usuario.

Resulta complicado regresar a versiones anteriores. Por ejemplo si se vuelve a una versión solicitada por un docente, esta podría no ser la apropiada para otro. En los laboratorios se debería hacer un análisis de requerimientos de todos los profesores que utilizan el sistema Linux desde el inicio de clases. Esto, para tener todas las herramientas necesarias instaladas para las materias antes de iniciar clases. De este modo se tendría una versión para cada semestre de clases.

Posteriormente con la realización de una imagen de las particiones y en caso de que se presente un error crítico se cargaría dicha imagen. Se evitaría la pérdida de tiempo usada en la reinstalación del sistema operativo y posteriormente en la instalación de cada herramienta solicitada.

1.5.1.1.2 Provisión del servicio

En esta sección se investiga la gestión de: disponibilidad, continuidad, capacidad de infraestructura de TI, financiera etc que existen en los laboratorios

La figura 1.3 ilustra los módulos de ITIL que forman parte de la provisión del servicio.

Figura 1.3 Metodología de provisión del servicio[6]

En los laboratorios no se cuenta con una base de datos de configuraciones completa (Configuration Management Data Base). No existe información de la configuración lógica de las computadoras.

[6] Fuente: http://itil.osiatis.es

- **Gestión de nivel de servicio**

A nivel de laboratorio no se pueden definir o formalizar los servicios de TI prestados por el sistema Linux, en términos de SLAs[7] y OLAs[8]. No se pueden definir niveles de servicios, rendimiento o tiempos de respuesta entre el proveedor de servicios (El laboratorio) y el cliente (estudiantes y profesores). Esto se debe a que por ejemplo el proveedor no está obligado a compensar al cliente en caso de no proveer por alguna circunstancia un servicio que debería brindar. No se pueden crear planes o emitir informes sobre la calidad de servicio brindada.

- **Gestión de capacidad**

No se tiene definido un procedimiento formal para planificar la capacidad de hardware requerida por el sistema. Antes de instalar el sistema operativo se asigna el espacio al mismo dependiendo del disco duro del computador.

El espacio asignado para el sistema Linux es actualmente de 70GB (en las computadoras con mejores características de hardware).Este valor es arbitrario y es tomado en base al espacio que ocupa el sistema en sí y a la suposición de cuánto espacio necesitan los usuarios para instalar herramientas. Especialmente para maquinas virtuales. En cuanto a recursos de memoria RAM y procesamiento.

No se controla periódicamente el rendimiento del hardware. La única planificación que existe es al finalizar el semestre. Aquí se evalúa el espacio en disco que se ha utilizado. Con esto se tiene una mejor apreciación de la que se va a necesitar para el siguiente.

[7] SLA: Acuerdos de Nivel de Servicio.
[8] OLA: Acuerdos de Nivel de Operaciones.

La planificación debería realizarse de manera más formal llevando registros de las capacidades requeridas. Resulta complicado planificar con exactitud el espacio necesitado debido a que siempre se debe dejar holgura para aplicaciones o archivos no contemplados inicialmente.

- **Gestión de continuidad**

Actualmente no se evalúan los posibles riesgos que atenten contra la continuidad del servicio del sistema. Ni siquiera se tienen identificados factores que podrían afectar la continuidad del mismo

En primer lugar se debería identificar los riesgos que podrían afectar la continuidad. Posteriormente se podría evaluar cada uno de ellos y elaborar planes de contingencia.

- **Gestión de disponibilidad**

No se tienen procedimientos formales para brindar disponibilidad del servicio en caso de presentarse fallos no previstos, sólo en caso de un fallo crítico se reinstala todo el sistema. Tampoco se tiene un cronograma de revisión y mantenimiento preventivo de la infraestructura de TI, todo funciona en base a solicitudes y quejas de los usuarios a las cuales se trata de dar solución lo más pronto posible.

Es muy importante que se tenga un cronograma para revisión y mantenimiento preventivo de la infraestructura. Con esto se podría garantizar una mayor disponibilidad de los servicios brindados.

- **Gestión financiera**

A este tipo de información resulta muy complicado acceder. Resulta desconocida para los estudiantes que administran los laboratorios. Por esta razón No se pueden medir las relaciones calidad/costo. Esto debido a que no se pueden evaluar los costes asociados a los servicios de TI de los laboratorios.

Los administradores de los laboratorios deberían conocer los costos asociados a los servicios de TI de los laboratorios. En base a esto se podrían establecer planes consistentes de inversión tecnológica y evitar desperdiciar recursos económicos y tecnológicos.

- **Gestión de seguridad**

No existen políticas de seguridad a nivel de software; y a nivel de hardware no se han definido políticas claras.

Se tiene definida una política de seguridad en contra del acceso de personal no autorizado. El laboratorio solamente se abre en el momento en que llega el profesor y sólo ingresan los alumnos pertenecientes a esa clase. El administrador es el encargado de cerrar el laboratorio. Se realiza en ocasiones, una revisión de los componentes externos de las computadoras.

No existe confidencialidad de la información del sistema, Linux es abierto. La información del sistema y todo el código es accesible a todos los usuarios. La integridad del sistema está en manos de los usuarios y de los administradores. De ellos depende que los archivos de configuración de los paquetes tengan información completa y correcta. Existe un problema de disponibilidad, al no tener instaladas y probadas las herramientas al momento de querer usarlas. Con el presente proyecto se pretende resolver este problema.

Es necesario implementar políticas de seguridad a nivel de hardware para evitar inconvenientes posteriores. Por ejemplo: para identificar al posible responsable de algún daño o pérdida.

1.5.1.2 INFORMACIÓN DE CONFIGURACIÓN DE HARDWARE DE LOS LABORATORIOS.

En los laboratorios todos los docentes utilizan CentOS como distribución de Linux, salvo en una materia que se utiliza Red Hat.

El motivo por el cual se utiliza CentOS es principalmente por la disponibilidad de paquetes requeridos en la materia de Gestión de Redes. Esta es la materia que más utiliza el sistema operativo Linux. Además, se la usa por facilidad en el manejo. Los administradores utilizaron esta distribución en cursos que recibieron anteriormente. Por esta razón están más familiarizados con la misma.

Las características de los componentes de hardware de las computadoras de los laboratorios son las siguientes:

Elemento	Marcas	Modelos (Características)
Procesadores	INTEL	Core 2 Quad CPU Q8200 2.33GHz Core 2 Duo 2,20 GHZ Intel Pentium 4 2.8 GHZ
Discos duros	SAMSUNG	120 GB, 160 GB , 240 GB, 320 GB (SATA)
Monitores	AOC, SAMSUNG	AOC-CRT 1024x768 SAMSUNG LCD SyncMaster 933
Tarjetas de red	Realtek, Intel	RTL8168D, PRO/100

Tarjetas de video	Intel	94GZ, G33/G31/G41
Unidades de CD/DVD	Toshiba, Samsung Storage Technology Corporation	CD/DVD ATAPI
Teclado		Idioma español

Tabla 1.4 Hardware del laboratorio de redes

En el desarrollo del presente proyecto se proveerá compatibilidad de hardware de acuerdo al hardware existente en el laboratorio.

1.5.2 ANÁLISIS DE RESULTADOS DE LA ENCUESTA DE LA SITUACIÓN ACTUAL DE LOS LABORATORIOS Y DE REQUERIMIENTOS FUNCIONALES.

En esta sección se presentan y analizan los datos obtenidos de la encuesta aplicada. En la parte de información general se listan las materias que hacen uso del sistema Linux. En la parte de percepción del nivel de servicio se ilustra la proporción de los resultados obtenidos en las encuestas mediante el uso de gráficas del tipo pastel. Posteriormente, en la parte de requerimientos funcionales hace uso de gráficas del tipo barras para exponer los resultados. Luego de cada gráfico se analizan los resultados expuestos.

La encuesta aplicada se encuentra en el Anexo B. A continuación se detalla por cada sección los resultados obtenidos en las encuestas realizadas:

Sección: Información general

1. Las siguientes materias utilizan el sistema basado en Linux para sus clases prácticas:

- Sistemas Operativos
- Gestión de Redes
- Aplicaciones Distribuidas
- Seguridad de Redes
- Redes WLAN
- Hardware de Conectividad
- Redes e Intranets

2. Número de horas a la semana promedio en que se usa el sistema Linux.

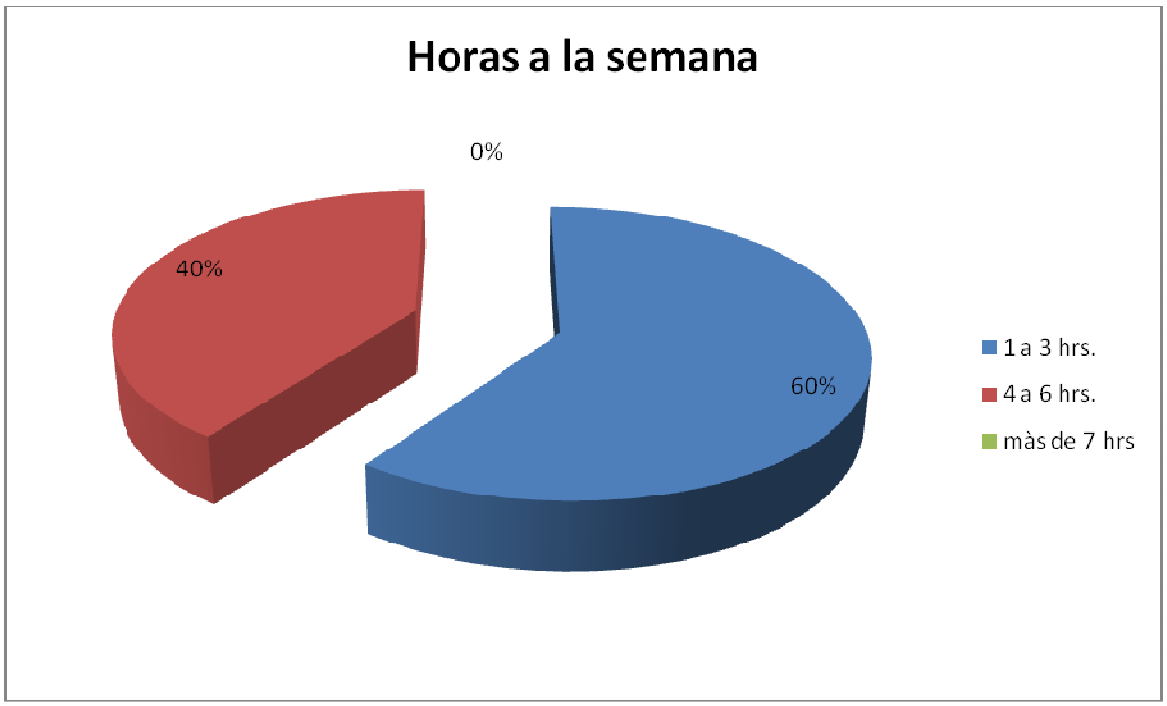


Figura 1.4 Horas a la semana promedio de uso del sistema Linux.

Un 60% de los docentes utiliza Linux entre 1 y 3 horas semanales, el otro 40% de los docentes lo utiliza entre 4 y 6 horas a la semana. La gráfica demuestra que el tiempo de uso del sistema Linux en los laboratorios durante la semana no es alto. Pese a esto el sistema se sigue y seguirá usando debido a que el perfil profesional de los estudiantes de Electrónica y Redes de Información lo demanda.

Sección: Percepción del nivel de servicio

3.- Tiempo de respuesta a solicitudes de instalación de nuevos paquetes.

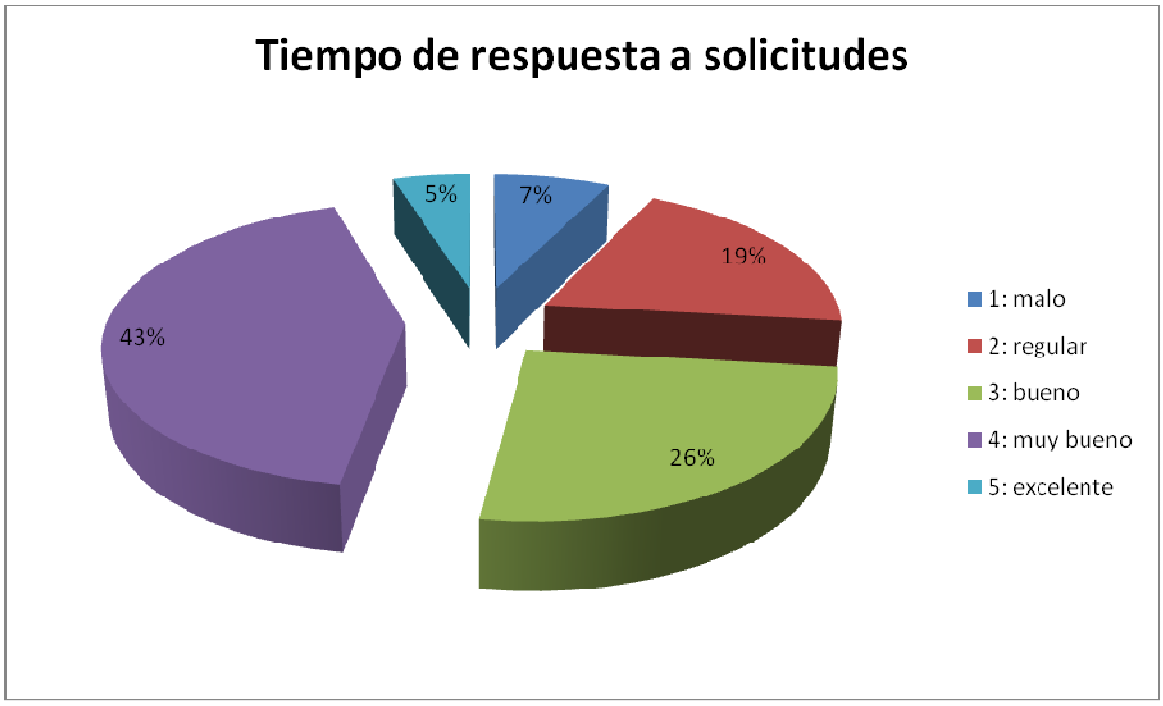


Figura 1.5 Tiempo de respuesta a solicitudes de instalación de nuevos paquetes.

Siendo 1 la calificación más baja y 5 la más alta en cuanto al tiempo de respuesta a solicitudes de nuevos paquetes, se observa que la mayoría, es decir el 43% de los encuestados califica el tiempo de respuesta con 4 (muy bueno). Seguido por un 26% que lo califica con 3 (bueno), en menor proporción se tienen calificaciones de 5 (excelente), 2 (regular) y 1 (malo).

La percepción de los usuarios está dividida en cuanto a la conformidad con el tiempo de respuesta a solicitudes. Esto indica que hay casos en que se pierden horas clase debido a retrasos en la respuesta a solicitudes.

4.- Disponibilidad de herramientas al momento de querer usarlas.

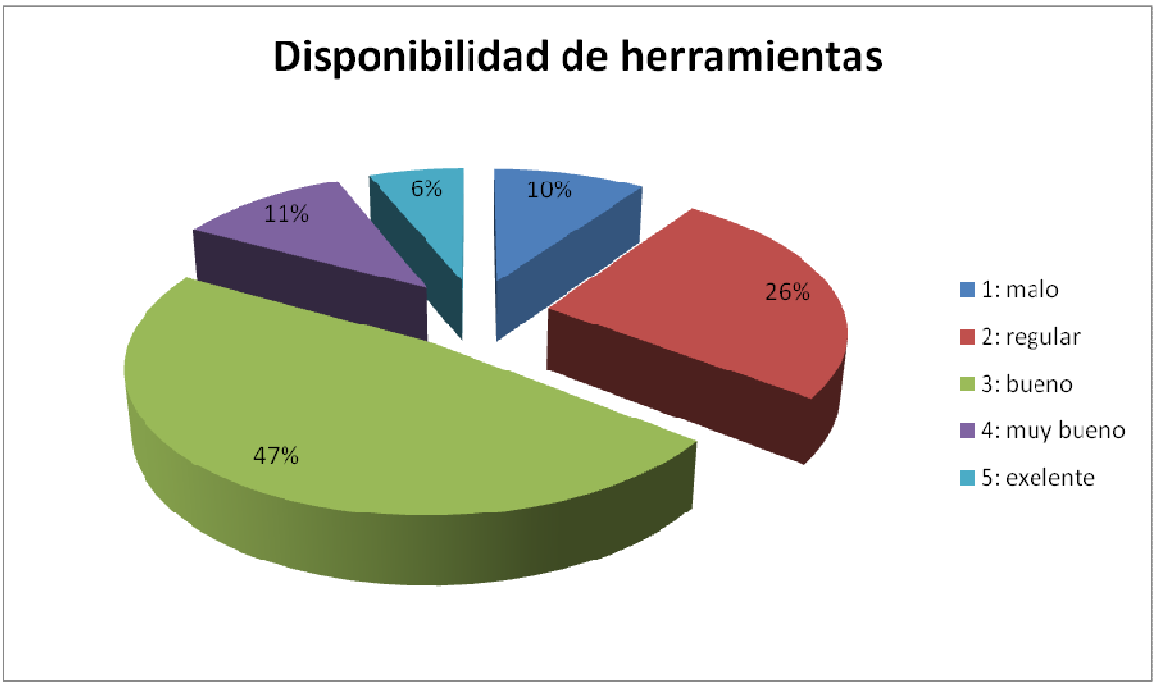


Figura 1.6 Disponibilidad de herramientas al momento de querer utilizarlas.

Siendo 1 la calificación más baja y 5 la más alta en cuanto a la disponibilidad de herramientas en el instante de querer utilizarlas, el 47% de encuestados le otorga una calificación de 3(bueno). Le sigue un 26% que le dan una calificación de 2(regular).

Se concluye que hay casos en que no se encuentran disponibles (instaladas y probadas) las herramientas necesarias en el momento de la clase pertinente.

5.- Solución eficaz por parte de los administradores a problemas suscitados en los laboratorios.

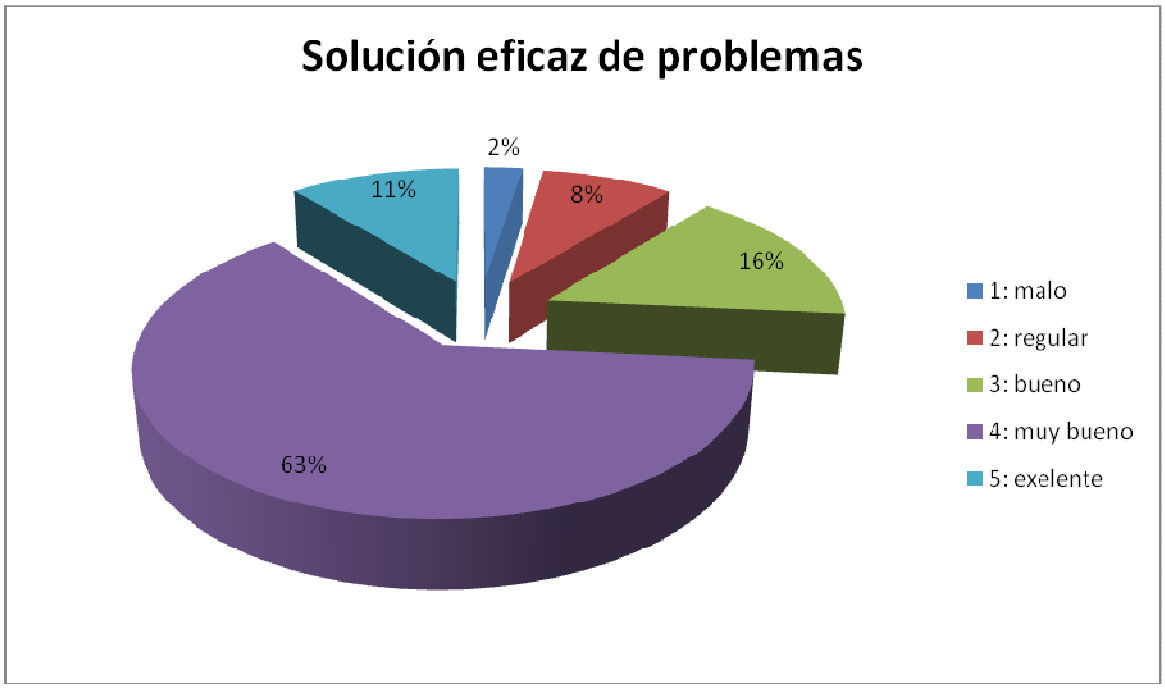


Figura 1.7 Solución eficaz a problemas.

Los resultados en cuanto a la solución eficaz de problemas que se brinda en los laboratorios, en una escala del 1 al 5, donde 5 es la calificación más alta muestran que: el 68% de encuestador la califican con 4 (muy bueno), lo que indica que en casi todos los casos se solucionan eficazmente los problemas. Le sigue la calificación de 3(bueno) con un 17% de los encuestados, los otros porcentajes son mucho menores.

Por parte de los usuarios existe conformidad en cuanto a soluciones eficaces brindadas por los administradores.

6.- Nivel de soporte técnico ofrecido por parte de la administración del laboratorio.

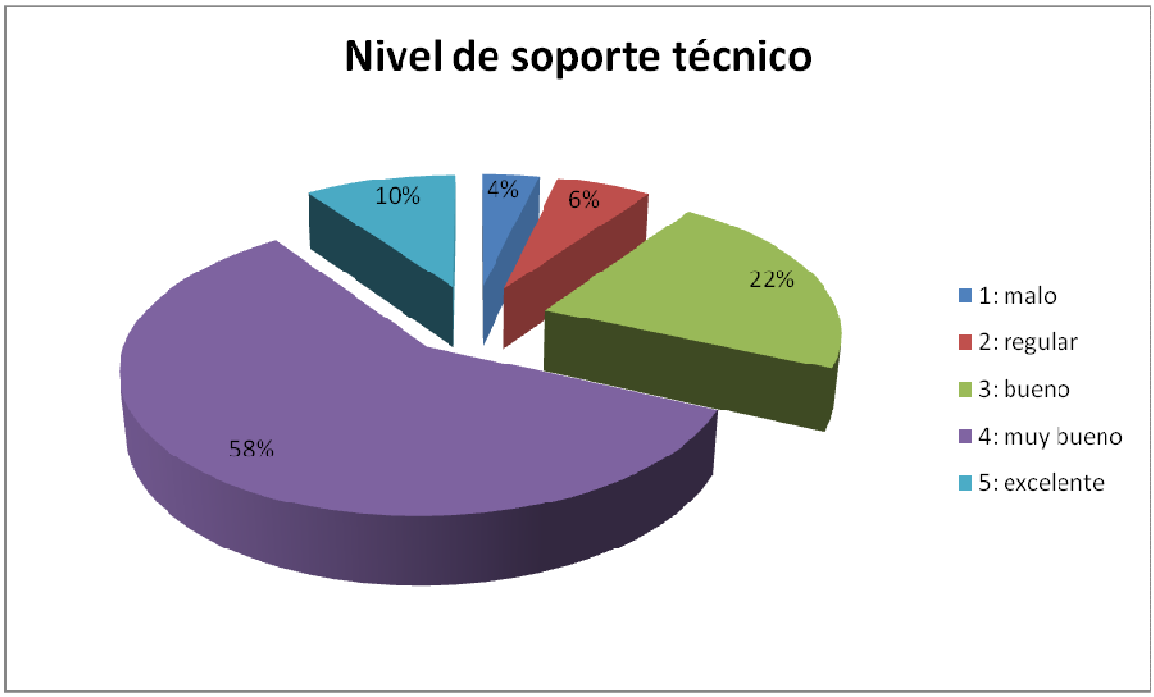


Figura 1.8 Nivel de soporte técnico de los administradores de los laboratorios.

En el nivel de soporte técnico percibido por parte de los usuarios siendo 5 el más optimo. Se establece con una calificación de 4(muy bueno) por parte del 58% de los encuestados. Un 22% lo califican con 3(bueno).

Los administradores de laboratorio al ser también estudiantes de la carrera, poseen conocimiento de las herramientas que se usan y del sistema Linux. Por esta razón, el nivel de soporte técnico recibido por los usuarios tiene una percepción de servicio muy buena.

Sección: Requerimientos

En esta sección se muestran los datos de los paquetes básicos que tendrá la distribución basada en Linux. A la vez se indican los programas necesarios para impartir las clases prácticas.

7.- Componentes básicos del sistema operativo basado en Linux

En la siguiente figura se muestran los datos recopilados en las encuestas.

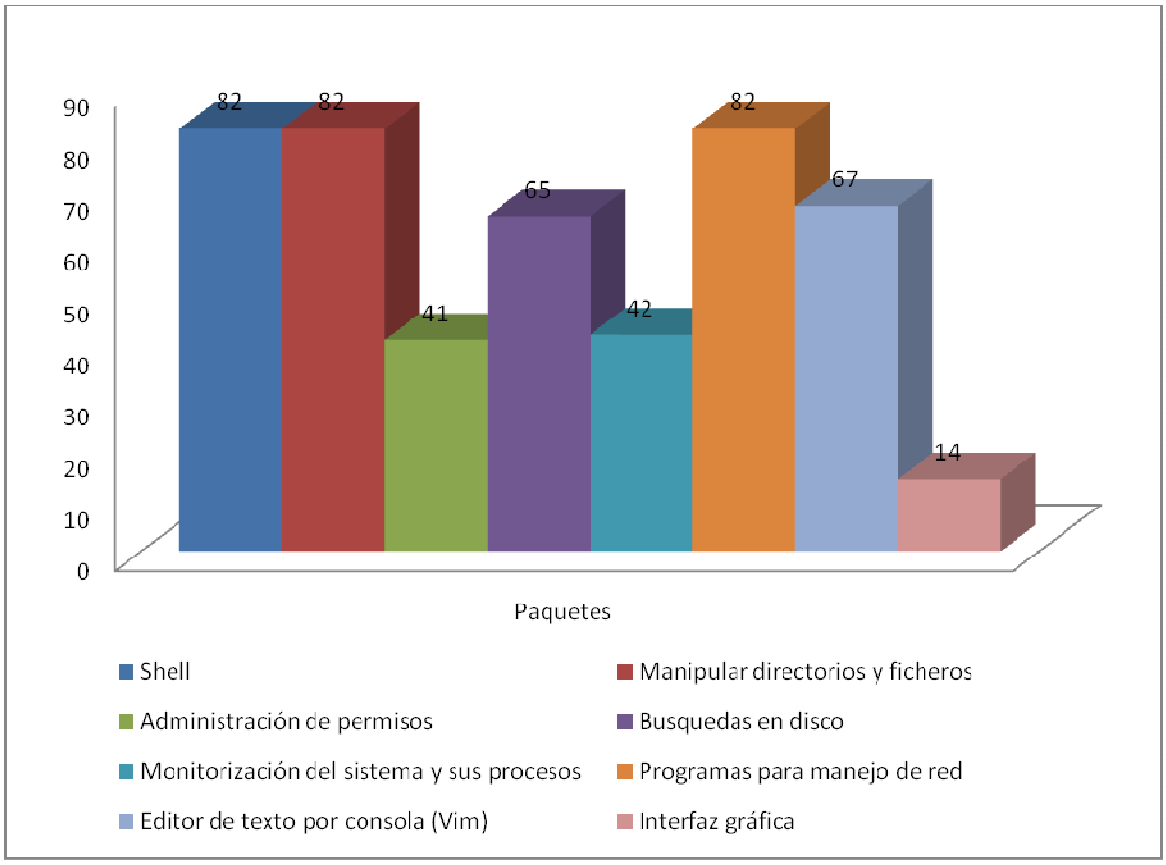


Figura 1.9 Componentes básicos del sistema (1)

Como se puede observar todos los paquetes considerados ¨básicos¨ en la encuesta tuvieron un alto número de aceptación, a excepción de la interfaz gráfica. Comenzando por: Shell, programas para manipular ficheros y directorios, y programas para el manejo de red que fueron considerados como básicos por todos los encuestados. Todos los demás paquetes sugeridos que fueron tomados en cuenta por más del 50% de encuestados.

Debido al alto porcentaje de aceptación de los paquetes mostrados en el gráfico se considerará a todo el grupo dentro de la nueva distribución a excepción de la interfaz

gráfica. Esta interfaz es utilizada necesariamente sólo por un paquete del análisis de requerimientos por lo que no será considerada en el sistema final.

8.- Programas necesarios para impartir las clases prácticas en los laboratorios.

La siguiente es la lista recopilada de programas que los profesores consideran necesarios para sus clases prácticas. Todos estos programas serán incluidos en la nueva distribución como parte de los requerimientos funcionales de la misma.

Paquete	Uso	Materia
NetSnmp	Todo el semestre	Gestión de Redes
Wireshark	Todo el semestre	Gestión de Redes
Traceroute	Todo el semestre	Gestión de Redes
Fping	Todo el semestre	Gestión de Redes
Nmap	Todo el semestre	Gestión de Redes
Java	1 clase	Sistemas operativos
NS-2	Todo el semestre	WLAN
Wireshark	Todo el semestre	Hardware de Conectividad
Apache	Todo el semestre	Hardware de Conectividad
FTP	Todo el semestre	Hardware de Conectividad
DNS	Todo el semestre	Hardware de Conectividad
Minicom	Todo el semestre	Hardware de Conectividad
OpenSSL	Todo el semestre	Seguridad de Redes
Kerberos	Todo el semestre	Seguridad de Redes
IPSec	Todo el semestre	Seguridad de Redes
OpenLDAP	Todo el semestre	Aplicaciones Distribuidas
Heartbeat	Todo el semestre	Aplicaciones Distribuidas
J2ee	Todo el semestre	Aplicaciones Distribuidas

Tabla 1.5 Software pedido por materia

No se indican paquetes para las clases de la materia de Redes e Intranets. Debido a que los contenidos de la materia y de las clases prácticas de laboratorio consisten en aprender a instalar y configurar herramientas y software en sistemas basados en Linux.

CAPÍTULO 2

DISEÑO Y CONSTRUCCIÓN DEL SISTEMA

En este capítulo se detalla el diseño del sistema basado en GNU/Linux. Así como el proceso de construcción del sistema, y la posterior creación del LiveCD, que es el entregable del proyecto.

Para el diseño del sistema se utilizarán los resultados provenientes del análisis de requerimiento y de situación actual del capítulo anterior. Para la construcción del sistema se utilizan los procedimientos explicados en la publicación de la metodología LFS (Linux From Scratch) en su versión 6.3, en conjunto con código fuente distribuido bajo licencia GPL de GNU. Luego, para cumplir con las peticiones obtenidas del análisis de resultados, el sistema básico creado a partir de LFS se complementará con procedimientos estipulados en BLFS, igual en su versión 6.3. Para finalizar este capítulo, se describen los pasos para la creación del LiveCD del sistema GNU/Linux.

2.1 DISEÑO DEL SISTEMA A IMPLEMENTAR

El diseño del sistema debe responder al análisis de requerimientos. Aquí se describen las capas que conforman el sistema. De manera seguida se describe el kernel que se implanta como parte del sistema, el cual se distribuye bajo licencia GPL y es el recomendado por la metodología LFS. También se describe la estructura de directorios con sus respectivos permisos. Se detallan los módulos o partes que complementan el núcleo. Así como sus librerías, compiladores, y utilitarios del sistema y del usuario.

2.1.1 CAPAS DEL SISTEMA [1] [12]

El diseño del proyecto contempla una distribución en base a tres capas. Al tener distribuido el sistema por medio de capas se logra jerarquizar los componentes que conforman el sistema a ser creado.

En la figura 2.1 se tiene como base al núcleo del sistema Linux. Esta capa se encarga de gestionar el manejo de hardware como son: gestión de memoria, manejo de periféricos, dispositivos de entrada/salida, etc. La segunda capa contempla las bibliotecas del sistema, que se utilizan como interfaz de comunicación entre la tercera capa (la que interactúa con el usuario) y la primera capa (núcleo del sistema). En la tercera capa se contemplan todos los utilitarios del sistema y del usuario, además de contener los compiladores del sistema.

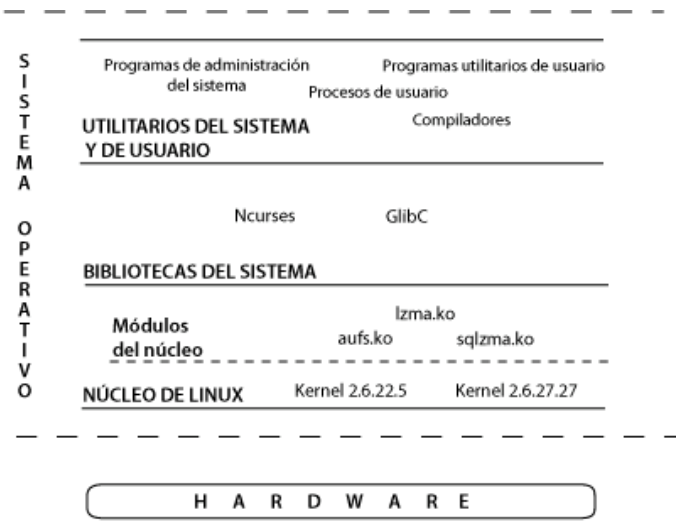


Figura 2.1 Capas del sistema a Implementar

2.1.2 KERNEL A IMPLANTAR [2] [16]

El kernel Linux a implantar dentro del sistema GNU/Linux se caracteriza por tener un arquitectura monolítica. Este tipo de arquitectura concentra funcionalidades básicas de un sistema operativo (gestión de memoria, controladores de hardware, gestión de

archivos, sistema de archivos, redes, etc) dentro de un solo programa llamado kernel. Por esto, para realizar cambios o añadir nuevas funcionalidades dentro del kernel es necesario recompilarlo, y volverlo a cargar. El kernel no tiene la flexibilidad de realizar cambios en caliente (Hot Swap). Al poseer esta estructura monolítica, las funciones y componentes del núcleo tienen acceso directo a sus rutinas y estructuras de datos internas

El kernel Linux provee la capacidad de que el sistema sea multitarea. Esto implica que el sistema presenta las tareas de forma intercalada. Se ejecutan varias simultáneamente. De esta manera se ejecutan varios programas a la vez. Esto sin necesidad de tener que parar la ejecución de cada aplicación.

El kernel Linux permite que el sistema sea multiusuario. El sistema operativo permitirá a varios usuarios acceder al mismo tiempo a través de terminales. Distribuye los recursos disponibles entre todos los usuarios.

2.1.3 MÓDULOS DEL KERNEL [8][12]

Los módulos del kernel permiten brindar funcionalidad modular al sistema operativo GNU/Linux. A diferencia del kernel, el cual es de estructura monolítica, el sistema operativo en conjunto brinda funcionalidad modular al usuario. Esto implica que se va a tener la facultad de cargar y descargar secciones arbitrarias de código (módulos) cuando sea necesario hacerlo. Los controladores externos y las extensiones del núcleo (archivos de extensión *.ko*) se pueden cargar y descargar fácilmente como módulos. El sistema continuará funcionando sin interrupciones al cargar un módulo. La funcionalidad de los módulos permite incluir de forma dinámica nuevas funcionalidades al núcleo del sistema operativo. Esto sin necesidad de recompilar el núcleo.

Debido a problemas de compatibilidad, durante el desarrollo del proyecto se hace uso de los módulos de kernel. Los módulos serán cargados en el desarrollo del LiveCD. Estos módulos son los correspondientes a: Aufs, Squashfs y Lzma. Éstos serán explicados en la sección 2.3.

2.1.4 COMPILADORES Y LIBRERÍAS [4]

Los compiladores son programas que permiten traducir el código fuente de un programa a lenguaje de máquina. Todos los programas, incluidos los del sistema y los programas pedidos en el análisis de requerimientos, deben ser compilados antes de ser instalados. Con base en el lenguaje de programación en el cual se distribuyen los programas, para el proyecto se implementa el paquete GCC, el cual incluye un compilador de lenguaje C y C++.

GCC se distribuye bajo licencia GPL de GNU. GPL está orientada principalmente a proteger la libre distribución, modificación y uso de software. Esta licencia permite construir el compilador desde el código fuente.

Como requisito para poder compilar el código fuente con GCC es necesario tener disponible las librerías de C y C++. En este caso se compila e instala GlibC la cual contiene la librería principal de C.

2.1.5 ESTRUCTURA DE DIRECTORIOS [4] [8]

Dentro del diseño del sistema GNU/Linux se sigue el estándar FHS (Filesystem Hierarchy Estándar). Esta norma define los directorios principales y sus contenidos en todo sistema operativo GNU/Linux. Por esta razón el sistema no saldrá del estándar. FHS determina en qué directorios deben existir y en cuáles se deben guardar los archivos de configuración, bibliotecas, binarios del sistema y archivos de datos.

En la figura 2.2 se detalla la estructura de directorios a implementar. Esta estructura también tiene su base en los programas que se instalan siguiendo la metodología de LFS. Cabe recalcar que existen programas que instalan sus propios directorios. Estos subdirectorios no se incluyen en la figura.

Todos los directorios serán de lectura y escritura (rw) en el LiveCD. Esto debido a que con la ayuda de Aufs se crea una rama o directorio que permite realizar modificaciones virtuales a cualquiera de ellos dentro del LiveCD. Físicamente ningún directorio se puede modificar debido a que el entregable al estar dentro de un CD es de sólo lectura.

Para un mayor detalle del contenido de cada directorio refiérase al Anexo C.

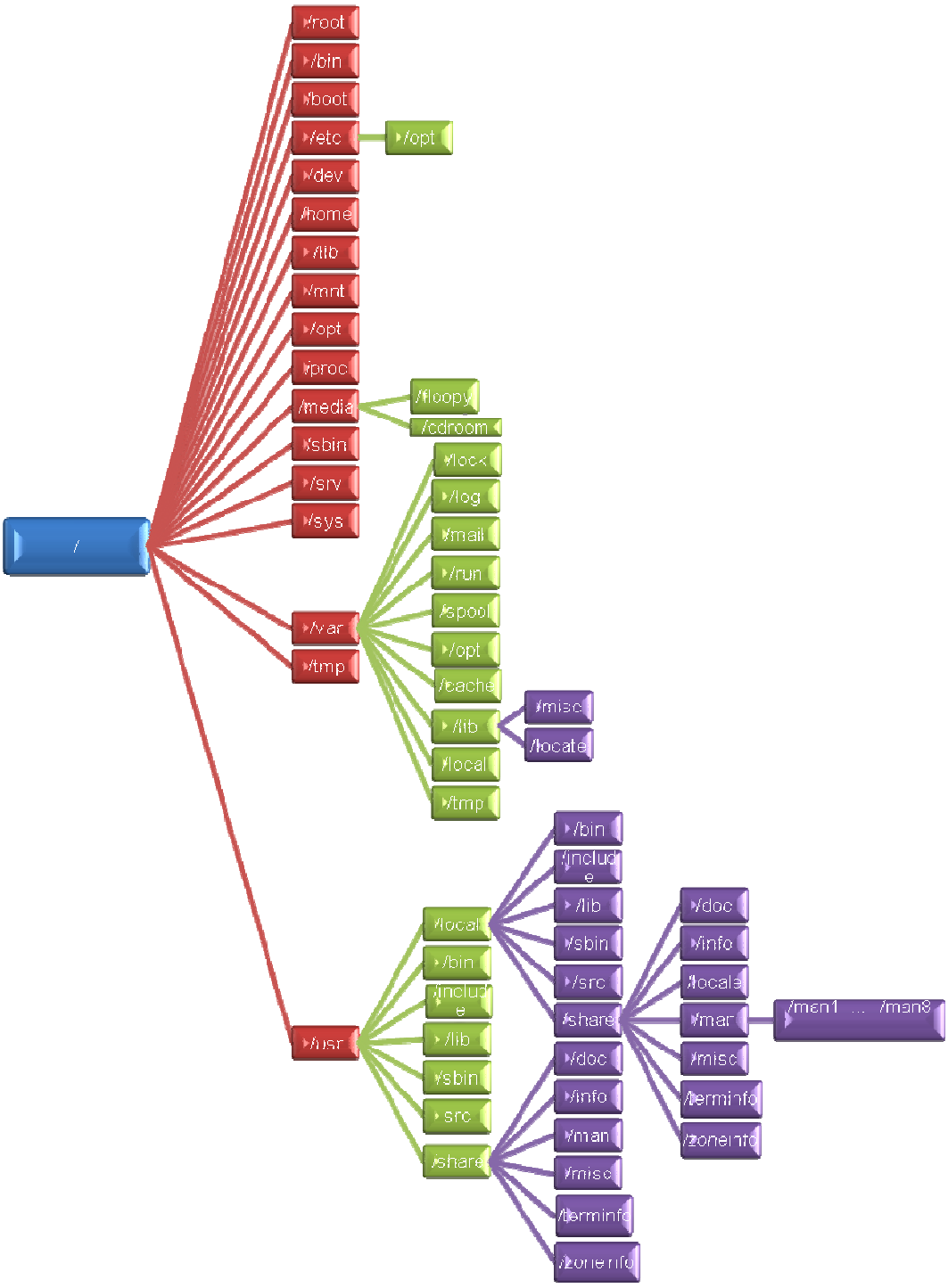


Figura 2.2 Estructura de directorios a implementar.

2.1.6 PAQUETES DE SOFTWARE

En el diseño se sigue la metodología LFS para construir un sistema GNU/Linux base. El orden de instalación y los paquetes a instalarse tienen como referencia la metodología LFS. En mayor cantidad el orden será determinado por el nivel de dependencias de cada paquete.

Adicional a los paquetes necesarios se instalarán paquetes que complementen la funcionalidad del sistema. Un ejemplo de esto es un cliente DHCP. No está incluido en el análisis de requerimientos ni en el sistema básico, pero necesario a nivel de laboratorio.

Para finalizar la construcción del sistema se instalarán los paquetes pedidos en el análisis de requerimientos. La instalación de estos paquetes y sus dependencias tiene su justificación en las encuestas y en el análisis realizado en el capítulo anterior.

2.2 CONSTRUCCIÓN DEL SISTEMA [4] [5]

En esta sección se procederá a construir el sistema. Se seguirá el diseño determinado en la sección 2.1. Se describirán todos los pasos que involucran el desarrollo del sistema. Los pasos se agrupan dentro de procesos. Como parte de los anexos del proyecto se presenta una guía de implementación.

Como se mencionó antes, para construir el sistema se utilizará la metodología LFS en su versión 6.3 (Linux from Scratch). Esta metodología permite instalar y configurar un sistema Linux a partir de código fuente.

La metodología divide a la construcción del sistema en 2 etapas. En primer lugar se creará un sistema Linux base (LFS). El sistema base contendrá: kernel, compilador

de C, y terminal de consola. Permitirá la ejecución de comandos como por ejemplo: comandos para manipular directorios y ficheros, para monitorizar el sistema y sus procesos, para manejo de red, etc. Sobre este sistema base van a correr todas las herramientas requeridas por los profesores. Esta etapa de la metodología se llevará a cabo con la realización de los procesos de la figura 2.3.

En la segunda etapa de la metodología se añadirá el software complementario. Se entiende así por ejemplo: los paquetes requeridos para brindar la funcionalidad deseada y scripts de configuración de inicio del sistema. Esta segunda parte tendrá como referencia la metodología BLFS 6.3 (Beyond Linux from Scratch). Ésta fue desarrollada por la comunidad BLFS. Se conseguirá un sistema funcional y a la medida de la mayoría de los requerimientos del laboratorio. La segunda etapa se llevará a cabo a través de los procesos mencionados en la figura 2.8.

Finalmente se creará el LiveCD de la distribución construida. Para la creación del LiveCD se sigue la metodología descrita en el foro de la página web: *www.linux-live.org.* Esta página detalla los pasos a seguir para la creación del LiveCD a partir de una distribución ya instalada. Se hace uso de los scripts publicados por Linux-Live.

2.2.1 CONSTRUCCIÓN DEL SISTEMA BASE [4]

Para construir un distribución de Linux, se necesita disponer de software base. Éste lo constituyen: un kernel Linux, un Shell de entrada de comandos, programas para editar, mover, borrar archivos, etc., bibliotecas del sistema, compiladores, etc. Todas estas herramientas son proporcionadas por una distribución anfitriona.

La metodología LFS recomienda utilizar cualquier distribución que cumpla con ciertos requisitos de software que contengan versiones mínimas de cada herramienta como prerrequisito. LFS recomienda el uso del LFS LiveCD. Este LiveCD es una distribución propia del proyecto sin interfaz gráfica. Esta distribución provee las

herramientas necesarias para completar la creación de la distribución, así como su código fuente.

Apegándose a la recomendación antes descrita, para el desarrollo del presente proyecto en su etapa inicial se utilizará el LFS-LiveCD como distribución anfitriona. Esta distribución correrá sobre un entorno de máquina virtual.

Para la construcción de la primera parte del sistema se incluyen los procesos que se observan en la figura 2.3.[9]

Figura 2.3 Procesos involucrados en la construcción del sistema básico

En las siguientes secciones se detallan todos los subprocesos involucrados.

2.2.1.1 Preparativos para la construcción

Las tareas llevadas a cabo dentro del subproceso de preparar la construcción se pueden observar en la siguiente figura:

[9] Para empezar la construcción, el sistema anfitrión debe ya estar corriendo en un entorno virtual.

Figura 2.4 Subprocesos involucrados en los preparativos para la construcción.

A continuación se explica y se justifica cada subproceso. El código fuente para la implementación de estos procesos se encuentra en el Anexo D, script 1.

- **Particionar el disco**

La partición albergará al nuevo sistema. La metodología recomienda crear la distribución en una partición diferente a la del sistema anfitrión. El LiveCD no requiere instalación nativa. Se crearán una partición de intercambio[10] y otra partición para el sistema a crear.

Para la partición del sistema LFS se va a utilizar un espacio de 8GB. Este espacio incluye la partición de intercambio. LFS recomienda hasta 3GB. Debido a que no se contempla la instalación de otras herramientas requeridas dentro de este espacio

[10] Partición de intercambio: partición en la que se cargan imágenes de procesos que no están en memoria física.

recomendado se amplía a 8GB para proveer suficiente espacio para los códigos fuente del resto de herramientas y su posterior compilación. La unidad de disco utilizado en el entorno de máquina virtual es del tipo SCSI.

- **Creación el sistema de archivos**

Se debe formatear la nueva partición. Esto se hará con el sistema de archivos de tipo ext3.[11]También se debe iniciar y activar la partición de intercambio, partición en la que se cargan imágenes de procesos que no están en memoria física.

- **Montar la partición**

Se debe montar la parición. Con esto se consigue hacerla accesible. Para esto se debe escoger un punto de montaje. Esto significa un directorio en particular para montarlo.

- **Creación de directorios para almacenar fuentes e instalaciones**

A lo largo de todo el desarrollo de la primera parte se utilizarán dos directorios en especial. Un directorio será para la instalación de todas las herramientas temporales (/mnt/lfs/tools). El otro directorio servirá para guardar todas las fuentes del proyecto (/mnt/lfs/sources).

- **Creación de un grupo y un usuario**

Por cuestiones de seguridad y para manejo de credenciales de seguridad y permisos se deben crear al menos un usuario y un grupo diferente al del *root*. Un error como usuario *root* podría destruir el sistema.

[11] ext3: tercer sistema de archivos extendido.

- **Configuración del entorno**

En esta tarea se crea un entorno de trabajo. Se crean dos ficheros de inicio para el intérprete de comandos. El fin es tener un entorno completamente vacío. Sin poseer variables de entorno propias del sistema anfitrión. Por medio de esto, el intérprete de comandos detecta los programas instalados en el directorio creado en /mnt/lfs/tools.

2.2.1.2 Construcción del sistema temporal

En esta parte se instalan y configuran las herramientas que conforman un entorno de desarrollo. Este entorno será independiente de la distribución anfitriona. Sólo habrá dependencia del núcleo en ejecución perteneciente al sistema anfitrión. Se proporcionará un entorno temporal al que se pueda entrar con *chroot*[12]. En la sección 2.2.1.3 se construirá el sistema básico definitivo.

En este sistema temporal se construye primero un conjunto de herramientas independientes del sistema anfitrión (compilador, librerías, etc.). Después se utilizan estas herramientas para construir el resto de paquetes correspondientes al sistema temporal.

Todos los ficheros compilados en esta sección se instalan solamente en el directorio de instalación creado anteriormente. No generan problema alguno con la distribución anfitriona o con la construcción del sistema definitivo. Debido a esto, resulta más fácil la eliminación posterior del las herramientas temporales.

[12] Chroot: cambia el directorio raíz. Un programa que se ejecuta con directorio raíz cambiado no puede acceder a ficheros fuera de este directorio.

Los paquetes de esta sección se instalan en orden de importancia y de dependencia. El orden de instalación es el provisto por la metodología LFS. Los paquetes instalados aquí no forman parte de la distribución final.

A continuación se describen los subprocesos implicados en construir el sistema temporal:

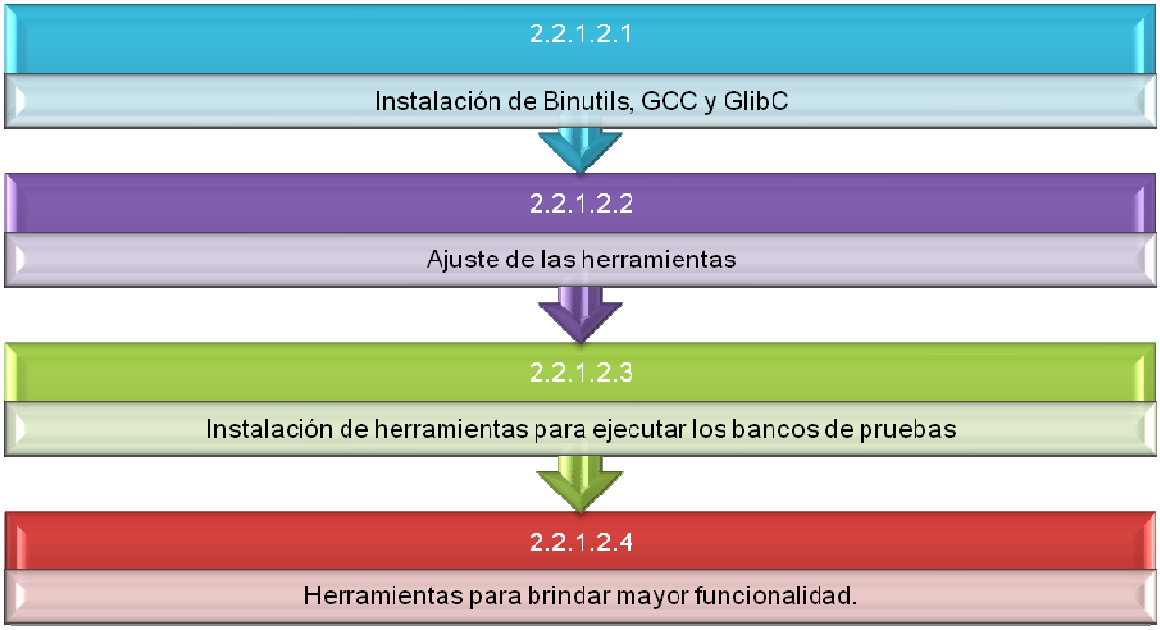


Figura 2.5 Subprocesos involucrados en la construcción del sistema temporal.

De manera seguida se explica, se justifica y se describen las tareas de cada subproceso. Para presentar la información se hace uso de tablas en las que se indica el nombre del paquete y su utilidad. El código fuente para la implementación de estos procesos se encuentra en el Anexo D, script 2 y 3.

2.2.1.2.1 Instalación de Binutils, GCC y Glibc

Binutils, GCC y Glibc son las herramientas principales a instalar. Éstas conforman: ensamblador, enlazador, compiladores y librerías de C. Juntas permiten instalar cualquier otro paquete.

- **Instalación de Binutils**

Binutils es un paquete que tiene una colección de herramientas binarias. Se destacan *ld*[13], el enlazador de GNU y *as*[14], el ensamblador de GNU.

"Se debe instalar primero Binutils debido a que, tanto en GCC como en Glibc, la ejecución de *configure* realiza varias pruebas sobre el ensamblador y el enlazador para determinar qué características del software deben activarse o desactivarse."[15]

- **Instalación de GCC**

El GCC es el paquete que contiene los compiladores de GNU. Se incluyen los compiladores de C y C++. Los compiladores mencionados son necesarios debido a que los siguientes paquetes serán instalados después de compilar sus códigos fuente. Estos paquetes vienen escritos en C o C++ y se distribuyen libremente bajo licencia GPL.

- **Instalación de Glibc**

Glibc contiene la librería principal de C. Esta librería proporciona rutinas para actividades como: búsqueda de directorios, creación, eliminación, lectura de ficheros, manejo de patrones, de operaciones aritméticas, entre otras.

[13] Enlazador dinámico: encuentra y carga las librerías compartidas necesarias para un programa.
[14] as: ensambla la salida del compilador gcc para ser usada por el linker ld, el ensamblador traduce de un fichero fuente a un fichero objeto.
[15] Fuente: LFS 6.3, Gerard Beekmans

2.2.1.2.2 Ajustar las herramientas

Es necesario realizar ajustes para asegurar que la búsqueda y el enlazado tengan lugar solamente dentro del directorio /tools. Así mismo, se instala un ld ajustado que limita su búsqueda interna al directorio /tools/lib.

Se debe modificar el fichero Specs de GCC para que apunte al nuevo enlazador dinámico en /tools/lib. Así se tendrá la seguridad de que todo binario utilizará el nuevo enlazador dinámico en /tools/lib Se hace esto ya que dentro de cada ejecutable ELF[16] se fija la ruta del enlazador dinámico. Con todo esto se asegura que cualquier paquete compilado hasta el final utilice el enlazador dinámico[17] en /tools/lib. De no realizar estas operaciones, los propios programas de GCC apuntarían al enlazador dinámico del sistema anfitrión en /lib. Esto generaría que no se tenga independencia del sistema anfitrión.

2.2.1.2.3 Herramientas para ejecutar los bancos de pruebas.

Se deben ejecutar los bancos de pruebas que vienen con las herramientas. Esto sirve para asegurar que las herramientas están instaladas correctamente. El propósito de la instalación de Tcl, Expect y DejaGNU es poder ejecutar los bancos de pruebas de GCC y Binutis.

La tabla 2.1 muestra las herramientas necesarias instaladas para ejecutar los bancos de pruebas.

[16] ELF: Executable and Linkable Format, es un format de archivo para ejecutables.
[17] Enlazador dinámico: encuentra y carga las librerías compartidas necesarias para un programa.

Herramienta	Descripción
▪ Tcl *Lenguaje de Herramientas de Comandos*	Es un lenguaje utilizado en scripts. Es un lenguaje multiplataforma y sirve para comprobar que las herramientas más importantes funcionan correctamente.
▪ Expect	Herramienta para probar aplicaciones. Está basado en *tcl*. Se comunica con otros programas interactivos según un guión.
▪ DejaGNU	Contiene un entorno de trabajo para comprobar otros programas. Está basado en *Expect*. Instala *runtest*. Este programa encuentra el intérprete de comandos de *expect* y ejecuta DejaGNU.

Tabla 2.1 Herramientas para ejecutar bancos de pruebas.

2.2.1.2.4 Instalación y configuración de herramientas para brindar mayor funcionalidad.

Luego de instalar y probar GCC y Glibc, es necesario instalar paquetes adicionales para aumentar la funcionalidad del sistema. Estos paquetes proporcionan todas las herramientas necesarias para construir el sistema básico. El orden de instalación y los paquetes a instalarse son especificados por LFS. La tabla 2.2 muestra las herramientas instaladas y una descripción de la utilidad de las mismas.

Herramienta	Descripción
▪ Ncurses	Contiene librerías para la programación en modo de texto. Su utilización se observa por ejemplo en el movimiento de cursores, colores, refresco de pantalla, etc. Instala programas como por ejemplo: *clear:* utilizado para limpiar la pantalla. También instala programas para manipular terminales.

▪ Bash	Es el paquete que contiene al intérprete de comandos "Bourne-Again SHell".
▪ Bzip2	Contiene programas para comprimir y descomprimir ficheros. Proporciona mayor porcentaje de compresión que *gzip.*
▪ Coreutils	Contiene programas para mostrar y establecer las características básicas del sistema. Instala programas como por ejemplo: *cat, chgrp, chmod, chown, chroot, date, df, dir, du, echo, install, hostname, mkdir, nice, printf, pwd, rm, touch,* etc.
▪ Diffutils	Contiene programas que muestran diferencias entre archivos o directorios. Instala programas como por ejemplo: *diff.*
▪ Findutils	Este paquete contiene utilidades para encontrar ficheros, hacer búsquedas recursivas en el árbol de directorios, etc. Instala programas como por ejemplo: *locate, find, updatedb.*
▪ Gawk	Es un editor de texto, que sirve para procesar patrones, es un programa de análisis semántico
▪ Gettext	Contiene utilidades para la internacionalización y localización. Permite a los programas compilarse con Soporte de Lenguaje Nativo (NLS). Esto permite que los mensajes mostrados estén en el mismo idioma del usuario.
▪ Grep	Contiene programas que ayudan a realizar búsquedas dentro de archivos. Por ejemplo instala el programa *grep*.
▪ Gzip	Este paquete contiene herramientas para comprimir y descomprimir archivos. Para este propósito instala *gzip, gunzip.*
▪ Make	Make contiene el programa para compilar paquetes.
▪ Patch	Contiene un programa para modificar o crear ficheros. Esto se realiza mediante la aplicación de un fichero que funciona como un parche. El parche se crea generalmente con el programa *diff.*

▪ Perl	Perl contiene el Lenguaje Práctico de Extracción e Informe.
▪ Sed	Sed contiene un editor de flujos orientado a líneas de texto.
▪ Tar	Contiene un programa de archivado. Instala el programa *tar*.
▪ Texinfo	Contiene programas para leer, escribir y convertir páginas info[18].
▪ Util-Linux	Este paquete contiene varias utilidades esenciales para un sistema Linux. Sirven por ejemplo para manejar sistemas de ficheros, mensajes, particiones, consolas, etc. Instala los programas: *cfdisk, fdisk, mkfs, mkswap, more, mount, umount, swapon, swapoff, whereis,* etc.

Tabla 2.2 Herramientas para brindar mayor funcionalidad.

Debido a que no se contempla una etapa de depuración ni en el desarrollo ni en el uso posterior del sistema, y con el objetivo de recuperar espacio de la partición en donde se trabaja en la construcción del sistema GNU/Linux es necesario eliminar los símbolos de depuración innecesarios generados al compilar binarios y librerías. Este espacio es liberado con la ayuda del comando *strip.*

2.2.1.3 Construcción del sistema base.

En esta sección se comienza la construcción del sistema definitivo. El orden de instalación sigue el establecido por la metodología. Esto asegura que ningún programa inserte su código fuera de la carpeta de instalación creada en /tools. Se explicará cada subproceso envuelto. Posteriormente se describen las tareas involucradas en cada subproceso. El código fuente para la implementación de estos procesos se encuentra en el Anexo D, script 4.

[18] Páginas info: similares a las de manual. Tienden a ser más profundas que una simple explicación

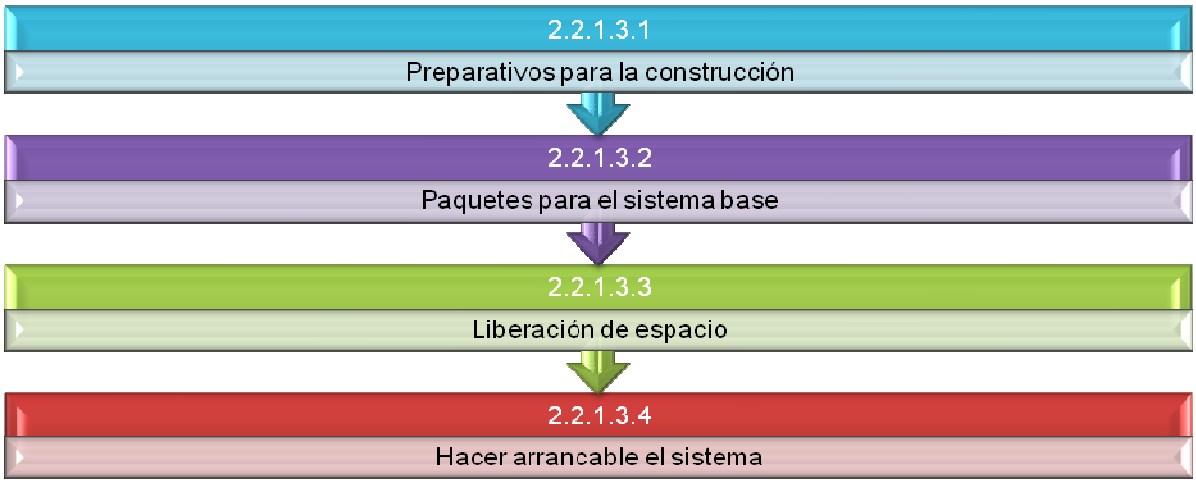


Figura 2.6 Subprocesos involucrados en la construcción del sistema básico

2.2.1.1 Preparativos para la construcción

- **Preparar los sistemas de ficheros virtuales del kernel**

Se exportan ficheros para la comunicación desde y hacia el núcleo. Estos ficheros son virtuales. Esto quiere decir que no usan espacio en el disco. El contenido de estos ficheros reside en la memoria.

VFS (Virtual File System) permite que cada proceso maneje información de sus ficheros. Esto sin necesidad de saber donde se encuentran. Tampoco sin necesidad de saber a qué tipo de sistema de archivos pertenecen. VFS permite a Linux soportar sistemas de archivos, como por ejemplo: FAT16, FAT32, NTFS (de windows), EXT2, EXT3, etc.

Estos ficheros se montarán sobre:

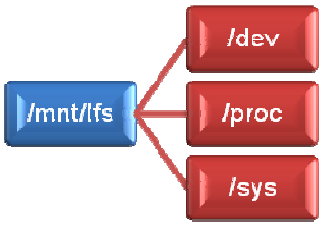


Figura 2.7 Ficheros en donde se montan los ficheros virtuales

/proc como sistema de ficheros no existe en realidad. Cuando VFS realiza llamadas a /proc, este crea los ficheros pedidos con información del kernel. El directorio /mnt/lfs actúa como directorio raíz.

- **Poblar /dev**

Linux accede a los dispositivos del sistema (hardware) a través de los nodos de dispositivos. Estos nodos se encuentran en el directorio /dev. Cuando arranca el sistema operativo necesita la presencia de estos nodos. En particular necesita los nodos *console* y *null.* Estos deben ser creados para que el sistema los encuentre en su arranque.

Para poblar /dev se monta un sistema de ficheros virtual como *tmpfs* en este directorio. Esto permite se creen los dispositivos según sean detectados. Esto lo hace *udev*[19] cuando arranca el sistema. Como el nuevo sistema aún no tiene *udev* instado se lo debe poblar manualmente. Esto se hace creando enlaces al directorio /dev del sistema anfitrión (Montaje enlazado). Así se crea una réplica del directorio.

- **Ingreso al entorno *chroot***

En este entorno se comienza la instalación del sistema final. En este entorno solamente se encuentran las herramientas temporales. Toda la construcción del

[19] Udev: es el gestor de dispositivos que usa el kernel de Linux.

sistema básico se realizará en este entorno. Esto se hace para usar las herramientas temporales creadas anteriormente.

- **Creación de los directorios.**

El árbol de directorios está basado en FHS (Filesystem Hierarchy Standard). Sólo se crearán los directorios necesitados por el sistema.

Los directorios se crean por defecto con permisos 755. El propietario tiene permisos para leer, escribir y ejecutar; el grupo y otros usuarios sólo de leer y ejecutar. Otros como el directorio /root tienen permisos 1777, es decir el propietario, el grupo y todos los usuarios pueden leer, escribir y ejecutar. El 1 al inicio es para que un usuario no pueda borrar o renombrar ficheros de otros usuarios.

La jerarquía de directorios construida es ilustrada en la figura 2.2 de la sección 2.1.5 que corresponde al diseño del sistema.

- **Creación de ficheros necesarios**

Linux mantiene una lista de ficheros montados en /etc/mtab. No se montará ningún sistema de ficheros. Se crea este fichero para las utilidades que esperan que se encuentre presente.

Para que se reconozca el nombre “root” se debe incluir entradas en los ficheros */etc/passwd* y */etc/group*. */etc/passwd* determina quién puede acceder al sistema y qué puede hacer dentro del sistema. */etc/group* contiene atributos de grupos. Para la creación de grupos se siguen las recomendaciones de: LSB (Linux Standard Base).
Se debe iniciar un nuevo intérprete de comandos. Se debe hacer que *bash* no ejecute rutas de binarios que ya ha ejecutado.

Se deben también crear los ficheros de registro. Estos ficheros abarcan información de los usuarios que están y estaban en el sistema. Por ejemplo *login, getty* e *init*son programas que mantienen ficheros de registro.

Ejemplo de ficheros de registro:

"El fichero /var/run/utmp lista los usuarios que están actualmente dentro del sistema, /var/log/wtmp registratodos los ingresos y salidas. El fichero /var/log/ lastlog muestra, para cada usuario, cuando fue la última vez que ingresó, y el fichero /var/log/btmp lista los intentos de ingreso fallidos."[20]

2.2.1.2 Paquetes para el sistema base

En esta sección se describen paquetes que forman parte del sistema definitivo. La información de cada paquete instalado para el sistema definitivo será presentada a modo de una página de manual, esto es: una tabla en la que se resume: la utilidad, justificación, dependencias, programas instalados, librerías instaladas, y archivos instalados de ser el caso. Estas tablas responden a la estructura de la tabla 2.3.

[20] Fuente: LFS 6.3, Gerard Beekmans

- ***Nombre del paquete*.**

Utilidad:	***Se define la utilidad del paquete.***	
Justificación:	*Indica el o los motivos que justifican el uso de este paquete*	
Dependencias para su instalación:	**Programas Instalados:**	**Librerías y directorios instalados:**
Se listan los programas necesarios que deben estar instalados antes que se proceda a instalar el paquete.	*Se listan los programas que se instalan luego de implantar el paquete.*	*Se listan las librerías necesarias que se instalan en conjunto con el paquete. Además se listan los directorios que se crean con el paquete, de ser el caso.*

Tabla 2.3 Estructura de las tablas para mostrar la información de los paquetes

Todos los paquetes de los que se depende fueron construidos en el sistema temporal. La lista de los paquetes de las dependencias se la obtuvo de LFS-6.3. Los comandos utilizados para instalar y configurar los paquetes, se encuentran en la guía de implementación ubicada en el anexo D.

A continuación se presentan los paquetes requeridos que necesita el sistema. Se incluyen paquetes pedidos en el análisis previo y programas necesarios debido a relaciones de dependencia. También se listan enlaces en los que se puede obtener mayor información. Esto debido a que no es el propósito del proyecto explicar a profundidad cada paquete que conforma el sistema.

- **Man-pages**

Utilidad:	Este paquete contiene páginas de manual.
Justificación:	Estas páginas son muy útiles para consultar funciones del lenguaje C. Describen ficheros de configuración.
Archivos instalados	
Páginas de manual	

Tabla 2.4 Características del paquete Man-pages

- **Glibc**

Utilidad:	El paquete Glibc contiene la principal librería de C.
Justificación:	La librería de C proporciona rutinas para actividades como: búsqueda de directorios, creación, eliminación, lectura de ficheros, manejo de patrones, de operaciones aritméticas, entre otras.
Dependencias para su instalación:	**Librerías instaladas**
Cabeceras del API de Linux, Bash, Binutils, Coreutils, Diffutils, Gawk, GCC, Gettext, Grep, Gzip, Make, Perl, Sed, y Texinfo.	Librería de C

Tabla 2.5 Características del paquete GlibC

Después de instalar GCC se debe reajustar las herramientas como en la sección 2.2.1.2.2. Cualquier nuevo programa compilado se debe enlazar contra la nueva Glibc.

- **Binutils (Binary Utilities)**

Utilidad:	Posee utilidades para trabajar con ficheros objeto. Contiene el enlazador, así como el ensamblador.	
Justificación:	Para la instalación de cualquier programa se debe trabajar con ficheros objeto. Principalmente se utilizan el ensamblador y el enlazador.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, Diffutils, GCC, Gettext, Glibc, Grep, Make, Perl, Sed y Texinfo	**as**.- ensamblador **ld**.- enlazador **readelf**.- muestra información sobre binarios de tipo ELF. **strip**.- elimina símbolos de ficheros objeto. **objcopy.-** traduce de un tipo de fichero objeto a otro.	**libiberty.-** contiene rutinas usadas por programas GNU. **libbfd**.- la librería del Descriptor de Fichero Binario. **libopcodes**.- una librería para manejar versiones en "texto legible" de las instrucciones del procesador.

Tabla 2.6 Características del paquete Binutils

- **GCC**

Utilidad:	Contiene la colección de compiladores de GNU. Incluye los compiladores de C y C++.	
Justificación:	Los compiladores permiten la traducción de un lenguaje de alto nivel a uno de máquina. Todo sistema operativo debe poseer un compilador.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, Diffutils, Findutils, Gawk, GCC, Gettext, Glibc, Grep, Make, Patch, Perl, Sed, Tar y Texinfo	**c++, g++**.- el compilador de C++ **cc, gcc.**- el compilador de C. **cpp**.- es el preprocesador de C.[21] **gccbug**.- ayuda a crear notificaciones de errores. **gccbug.-** analiza programas para encontrar optimizaciones.	**libgcc**.- soporte en tiempo de ejecución para gcc. **libmudflap**.- contiene rutinas para la comprobación de límites de funcionalidad de GCC. **libstdc**++.- es la librería estándar de C++. **libsupc**++.- proporciona rutinas de soporte para el lenguaje de programación c++..

Tabla 2.7 Características del paquete GCC

[21] Preprocesador de C: utilizado para expandir sentencias como #include y #define en ficheros objeto

- **Berkeley DB**

Utilidad:	Contiene utilidades y funciones utilizadas para aplicaciones que trabajan con bases de datos
Justificación:	Requerido para la instalación de OpenLDAP.
Dependencias para su instalación:	
Bash, Binutils, Coreutils, Diffutils, Gawk, GCC, Glibc, Grep, Make y Sed	

Tabla 2.8 Características del paquete Berkeley DB

- **Sed**

Utilidad:	Contiene un programa para editar flujos de texto.
Justificación:	Este programa es muy utilizado durante la construcción de paquetes. Es utilizado principalmente para filtrar texto[22].
Dependencias para su instalación:	**Programas Instalados**
Bash, Binutils, Coreutils, GCC, Gettext, Glibc, Grep, Make, Sed y Texinfo	**sed**

Tabla 2.9 Características del paquete Sed

- **Coreutils (GNU Core Utilities)**

Utilidad:	Contiene herramientas para utilidades de ficheros (fileutils), utilidades de intérpretes de comandos (shellutils) y utilidades de proceso de textos (textutils),
Justificación:	Este tipo de programas fueron requeridos en el análisis de requerimientos. Son utilidades esperadas en cualquier sistema

[22] Fuente: http://www.gnu.org/software/sed/

	operativo.
Dependencias para su instalación:	**Programas Instalados**
Bash, Binutils, Coreutils, GCC, Gettext, Glibc, Grep, Make, Patch, Perl, Sed y Texinfo	**cat.-** concatena ficheros en la salida estándar. **chgrp.-** cambia el grupo propietario de ficheros y directorios. **chmod.-** cambia los permisos de cada fichero dado a los nuevos permisos. **chown.-** cambia el usuario y/o el grupo propietario de ficheros y directorios. **chroot.-** ejecuta un comando usando el directorio especificado como directorio. **cp.-** copia ficheros. **df**.- muestra la cantidad de espacio disponible (y usado) en todos los sistemas de ficheros montados o seleccionados. **echo**.- muestra la cadena indicada. **install.-** copia ficheros mientras establece sus permisos, su propietario y grupo. **ln.-** crea enlaces simbólicos duros o blandos (simbólicos) entre ficheros. **mkdir.-** crea directorios con los nombres indicados. **mv.-** mueve o renombra ficheros o directorios. **nice.-** ejecuta un programa con una prioridad distinta. **printf.-** muestra los argumentos de acuerdo al formato indicado. **pwd.-** muestra el nombre del directorio de trabajo actual. **rm.-** elimina ficheros o directorios. **touch.-** cambia las fechas de modificación o acceso de un fichero especificado, poniéndole la fecha actual.Si un fichero no existe crea uno vacío.

Tabla 2.10 Características del paquete Coreutils

- **M4**

Utilidad:	Contiene un procesador de macros. Copia ficheros expandiendo las macros que estos contengan. Tiene funciones internas para incluir ficheros, lanzar comandos UNIX, hacer aritmética entera.[23]
Justificación:	Bash depende para su instalación de Bison que a su vez depende de M4. Flex también necesita a M4 para su instalación

Dependencias para su instalación:	**Programas Instalados**
Bash, Binutils, Coreutils, GCC, Glibc, Grep, Make y Sed	**M4**

Tabla 2.11 Características del paquete M4

- **Bison**

Utilidad:	Es un generador de *parsers* (analizador sintáctico) de propósito general[24]. Puede usarse para crear parsers de lenguajes.
Justificación:	Bash depende para su instalación de Bison.

Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, GCC, Gettext, Glibc, Grep, M4, Make y Sed	**bison.-** genera, a partir de reglas, un programa para analizar la estructura de ficheros de texto. **yacc.-** utilizado por programas que todavía llaman a yacc en lugar de a bison.	**liby.a.-** es la librería de Yacc

Tabla 2.12 Características del paquete Bison

[23] Fuente: LFS 6.3, Gerard Beekmans.
[24] Fuente: http://www.gnu.org/software/bison/

- **Ncurses (New Cures)**

Utilidad:	Es una biblioteca de programación. Provee un API que permite escribir interfaces basadas en texto. También optimiza el refresco de la pantalla.[25]	
Justificación:	Vim y util_linux, dependen para su instalación de Ncurses. Vim fue pedida en el análisis de requerimientos.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, Diffutils, Gawk, GCC, Glibc, Grep, Make, Patch y Sed	**clear**.- limpia la pantalla **tset**.- sirve para inicializar terminales Etc.	**libncurses**.- contienen funciones para mostrar texto en la pantalla de un terminal. Por ejemplo el uso de estas funciones es el menú que se muestra en el proceso *make menuconfig* del núcleo. **Libform.-** contiene funciones para implementar formularios. **libmenu**.- contiene funciones para implementar menús. **libpanel.-** contiene funciones para implementar paneles.

Tabla 2.13 Características del paquete Ncurses

[25] Fuente: http://es.wikipedia.org/wiki/Ncurses

- **Procps**

Utilidad:	Contiene programas para monitorizar procesos.	
Justificación:	Este tipo de programas fueron pedidos en las encuestas de análisis de requerimientos.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, GCC, Glibc, Make y Ncurses	**free**.- muestra la cantidad total de memoria libre. **kill.- e**nvía señales a los procesos. **pgrep.-** visualiza procesos basándose en su nombre u otros atributos **pkill.-** envía señales a procesos basándose en su nombre u otros atributos. **ps.-** muestra los procesos actuales. **top.-** muestra los procesos más activos en uso de CPU. **uptime.-** muestra cuánto tiempo hace que el sistema está en ejecución, cuántos usuarios están conectados y la carga media del sistema. **vmstat.-** muestra estadísticas de la memoria virtual.	**libproc**.- contiene funciones usadas por los programas de este paquete.

Tabla 2.14 Características del paquete Procps

- **Libtool**

Utilidad:	Provee un guión de GNU para dar soporte generalizado para la compilación de librerías.	
Justificación:	Este programa brinda soporte para poder instalar todas las librerías necesarias.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, Diffutils, Gawk, GCC, Glibc, Grep, Make, Sed y Texinfo	**libtool**	**libltdl.-** oculta las dificultades para abrir la carga dinámica de las librerías.

Tabla 2.15 Características del paquete Libtool

- **Perl**

Utilidad:	Este paquete contiene al lenguaje de programación PERL (Lenguaje Práctico de extracción e Informe).	
Justificación:	Es requerimiento para los principales paquetes como GCC, Glibc y Binutils. Además es prerrequisito para la instalación de heartbeat que fue pedido en el análisis de requerimientos.	
Dependencias para su instalación:	**Programas y librerías Instaladas.**	
Bash, Berkeley DB, Binutils, Coreutils, Gawk, GCC, Glibc, Grep, Groff, Make y Sed	Instala una gran cantidad de paquetes y librerías. Para mayor información refiérase a: http://www.perl.org/	

Tabla 2.16 Características del paquete Perl

- **Readline**

Utilidad:	Este paquete contiene librerías con funciones para edición en línea de comandos y capacidades de historial.[26]
Justificación:	Es utilizado por Vim que fue pedido en el análisis de requerimientos
Dependencias para su instalación:	**Librerías Instaladas**
Bash, Binutils, Coreutils, GCC, Glibc, Grep, Make, Ncurses, Patch, Sed y Texinfo	Para mayor información refiérase a: http://tiswww.case.edu/php/chet/readline/readline.html.

Tabla 2.17 Características del paquete Readline

- **Zlib**

Utilidad:	Este paquete contiene librerías con funciones para compresión y descompresión.
Justificación:	Estas funciones fueron pedidas en el análisis de requerimientos
Dependencias para su instalación:	**Librerías Instaladas**
Bash, Binutils, Coreutils, GCC, Glibc, Grep, Make y Sed	**libz**

Tabla 2.18 Características del paquete Zlib

[26] Fuente: http://tiswww.case.edu/php/chet/readline/rltop.html

- **Autoconf**

Utilidad:	Es un paquete extensible de M4. Produce shellscripts. Estos automatizan la configuración de paquetes desde código fuente.[27]
Justificación:	Ayuda a la configuración de paquetes de código fuente. Todos los paquetes del sistema se crean desde el código fuente.
Dependencias para su instalación:	**Programas Instalados**
Bash, Coreutils, Grep, M4, Make, Perl, Sed y Texinfo	Para mayor información refiérase a: http://www.gnu.org/software/autoconf/

Tabla 2.19 Características del paquete Autoconf

- **Automake**

Utilidad:	Es una herramienta para generar automáticamente archivos *makefile*.[28] Estos *makefiles* se utilizan con *autoconf*.
Justificación:	Ayuda a la creación de archivos makefile de paquetes de código fuente.
Dependencias para su instalación:	**Programas Instalados**
Autoconf, Bash, Coreutils, Gettext, Grep, M4, Make, Perl, Sed y Texinfo	Para mayor información refiérase a: http://www.gnu.org/software/automake/

Tabla 2.20 Características del paquete Automake

[27] Fuente: http://www.gnu.org/software/autoconf/
[28] Makefile: es un archivo que establece las versiones a seguir para crear una versión ejecutable de un sistema.

- **Bash**

Utilidad:	Es el intérprete de comandos. Contiene el Bourne-Again Shell.
Justificación:	Todos los sistemas operativos tienen su intérprete de comandos. Este paquete también fue pedido en el análisis de requerimientos.

Dependencias para su instalación:	**Programas Instalados**
Bash, Bison, Coreutils, Diffutils, GCC, Glibc, Grep, Make, Ncurses, Patch, Readline,Sed y Texinfo	**bash.-** el intérprete de comandos. **bashbug.-** un guión que ayuda al usuario en la composición y envío de informes de errores relacionados con bash. **sh.-** enlace simbólico al programa **bash**.

Tabla 2.21 Características del paquete Bash

- **Bzip2**

Utilidad:	Contiene programas para comprimir y descomprimir ficheros.
Justificación:	Paquetes con este tipo de utilidad fueron pedidos en el análisis de requerimientos.

Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, Diffutils, GCC, Glibc, Make y Patch	**bunzip2**.- descomprime ficheros que han sido comprimidos con bzip2. **bzip2**.- comprime ficheros usando el algoritmo: Burrows-Wheeler con codificación Huffman. La compresión es superior en relación a otros compresores basados en el algoritmo “Lempel-Ziv”, como gzip.	**libbz2**.- la librería que implementa que utiliza bzip2.

Tabla 2.22 Características del paquete Bzip2

- **Diffutils**

Utilidad:	Es un paquete que tiene programas para encontrar diferencias entre archivos.
Justificación:	El kernel de Linux requiere este paquete para su instalación.
Dependencias para su instalación:	**Programas Instalados:**
Bash, Binutils, Coreutils, Diffutils, GCC, Gettext, Glibc, Grep, Make, Patch, Sed y Texinfo	**cmp.-** compara dos ficheros e informa en dónde o en qué bytes difieren. **diff.-** compara dos ficheros o directorios e informa qué líneas de los ficheros difieren. **diff3.-** compara tres ficheros línea a línea.

Tabla 2.23 Características del paquete Diffutils

- **File**

Utilidad:	Contiene una utilidad para determinar tipos de ficheros	
Justificación:	Es muy útil cuando se desea saber qué tipo de archivo se tiene. Esto cuando no se tiene una terminación que indica el tipo de archivo.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
	file.- intenta clasificar los ficheros indicados. Lo hace realizando 3 pruebas: pruebas de sistemas de	**libmagic**.-contiene rutinas para reconocimiento de

	ficheros, pruebas de números mágicos[29] y pruebas de lenguajes.	números mágicos.

Tabla 2.24 Características del paquete File

- **Findutils**

Utilidad:	Este paquete tiene programas que permiten encontrar ficheros. Permiten realizar búsquedas recursivas en el árbol de directorios.
Justificación:	Este tipo de programas fue considerado como un requisito básico en el análisis de requerimientos.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, GCC, Gettext, Glibc, Grep, Make, Sed y Texinfo	**find.-** busca en los árboles de directorios indicados los ficheros que cumplan el criterio especificado. **locate**.- busca en una base de datos de nombres de ficheros y muestra los nombres que contienen la cadena indicada o cumplen un patrón dado. **updatedb.-** actualiza la base de datos de locate.

Tabla 2.25 Características del paquete Findutils

[29] Números mágicos: firma que dejan los formatos de texto en los archivos.

- **Flex**

Utilidad:	Contiene un programa que reconoce patrones de texto.	
Justificación:	Este programa es utilizado por ejemplo para realizar búsquedas. Se usan patrones de texto para la creación de scripts	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Bash, Binutils, Coreutils, GCC, Gettext, Glibc, Grep, M4, Make, Patch, Sed y Texinfo	**Flex**	**libfl.a**.- la librería flex.

Tabla 2.26 Características del paquete Flex

- **GRUB**

Utilidad:	Es el gestor de arranque del sistema. Este paquete contiene el Grand Unified Bootloader.
Justificación:	El sistema debe tener un gestor de arranque. Prepara todo lo que necesita el sistema operativo para iniciar.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, Diffutils, GCC, Glibc, Grep, Make, Ncurses, Sed y Texinfo	**grub.-** el intérprete de comandos del GRand Unified Bootloader (Gran Gestor de ArranqueUnificado). **grub-install.-** instala GRUB en el dispositivo indicado. **grub-md5-crypt**.- encripta una contraseña en formato MD5. **grub-set-default.-** establece la entrada de arranque por defecto para GRUB

Tabla 2.27 Características del paquete GRUB

- **Gawk**

Utilidad:	Este paquete contiene programas para manipular ficheros de texto.
Justificación:	Gettex depende para su instalación de Gawk. Es útil para modificar archivos, buscar y transformar bases de datos, generar informes, etc. Esto también fue pedido en el análisis de requerimientos.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, GCC, Gettext, Glibc, Grep, Make, Patch, Sed y Texinfo	**gawk.-** un programa para manipular ficheros de texto.

Tabla 2.28 Características del paquete Gawk

- **Gettext**

Utilidad:	Contiene utilidades para la internacionalización y localización. Permite mostrar los mensajes de compilación en el idioma nativo del usuario.
Justificación:	Permite mostrar los mensajes en idioma español.
Dependencias para su instalación:	**Programas y librerías Instaladas.**
Bash, Binutils, Coreutils, Gawk, GCC, Glibc, Grep, Make, Sed y Texinfo	Para mayor información refiérase a: http://www.gnu.org/software/gettext/

Tabla 2.29 Características del paquete Gettext

- **Grep**

Utilidad:	Contiene programas para realizar búsquedas dentro de ficheros.
Justificación:	Fue requerido en el análisis de requerimientos. Muy utilizado en la creación de scripts en el laboratorio.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, Diffutils, GCC, Gettext, Glibc, Make, Patch, Sed y Texinfo	**egrep.-** muestra las líneas que coincidan con una expresión regular extendida. **fgrep.-** muestra las líneas que coincidan con una lista de cadenas fijas. **grep.-** muestra las líneas que coincidan con una expresión regular básica.

Tabla 2.30 Características del paquete Grep

- **Groff**

Utilidad:	Contiene programas para procesar y formatear texto.
Justificación:	Es utilizado para la instalación de las páginas de manual de los paquetes.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Bison, Coreutils, Gawk, GCC, Glibc, Grep, Make, Patch, Sed y Texinfo	**groff**

Tabla 2.31 Características del paquete Groff

- **Gzip**

Utilidad:	Contiene programas para comprimir y descomprimir ficheros**.**
Justificación:	Este paquete fue pedido en el análisis de requerimientos
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, Diffutils, GCC, Gettext, Glibc, Make, Patch, Sed y Texinfo	**gunzip.-** descomprime ficheros que hayan sido comprimidos con *gzip.* **gzexe.-** crea ficheros ejecutables autodescomprimibles. **gzip.-** comprime los ficheros indicados usando codificación Lempel-Ziv (LZ77). **uncompress.-** descomprime ficheros comprimidos.

Tabla 2.32 Características del paquete Gzip

- **Inetutils**

Utilidad:	Contiene utilidades para trabajo en la red.
Justificación:	Paquete pedido en el análisis de requerimientos. Necesario para uso en el laboratorio de redes.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, GCC, Glibc, Grep, Make, Ncurses, Patch, Sed y Texinfo.	**ftp.-** el programa para transferencia de ficheros. **ping.-** envía paquetes de petición de eco e informa cuánto tardan las respuestas. **ping6.-** versión de ping para redes IPv6. **rcp.-** copia ficheros de forma remota. **rlogin.-** realiza entradas remotas a un sistema. **rsh.-** ejecuta un intérprete de comandos remoto.

	talk.- permite hablar con otro usuario. **telnet.-** una interfaz de usuario para el protocolo TELNET. **tftp.-** un programa para la transferencia trivial de ficheros

Tabla 2.33 Características del paquete Inetutils

- **Kbd**

Utilidad:	Este paquete contiene ficheros para mapas de teclado y utilidades para el teclado.
Justificación:	Este es un programa que servirá para brindar flexibilidad en cuanto al uso de teclados.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Bison, Coreutils, Flex, GCC, Gettext, Glibc, Gzip, Make, Patch y Sed	Para mayor información refiérase a: http://www.linuxfoundation.org/collaborate/workgroups/accessibility/kbd

Tabla 2.34 Características del paquete kbd

- **Make**

Utilidad:	Contiene un programa para compilar paquetes.
Justificación:	Al instalar todos los paquetes desde código fuente se requiere siempre este programa.
Dependencias para su instalación:	**Programa Instalados.**
Bash, Binutils, Coreutils, Diffutils, GCC, Gettext, Glibc, Grep, Make, Sed y Texinfo	**make.-** determina automáticamente qué partes de un paquete necesitan ser (re)compiladas y lanza los comandos para hacerlo

Tabla 2.35 Características del paquete Make

- **Man-DB**

Utilidad:	Contiene programas para encontrar y visualizar páginas de manual.
Justificación:	Permitirá a los usuarios leer las páginas de manual en caso de que las necesite.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Berkeley DB, Binutils, Bzip2, Coreutils, Flex, GCC, Gettext, Glibc, Grep, Groff,Gzip, Less, Make y Sed.	**apropos.-** búsca en la base de datos de *whatis* y muestra las descripciones cortas de los comandos del sistema que contienen la cadena dada. **catman.-**crea o actualiza las páginas de manual preformateadas **convert-mans.-** reformatea páginas de manual para que Man-DB pueda leerlas. **lexgrog.-** Muestra información sumarizada en una línea sobre la página de manual dada. **man.-** formatea y muestra la página de manual solicitada. **mandb.- c**rea o actualiza la base de datos de *whatis*. **whatis.-** busca en la base de datos de *whatis* y muestra las descripciones cortas de los comandos del sistema.

Tabla 2.36 Características del paquete Man-DB

- **Module-Init-Tools**

Utilidad:	Contiene programas para manejar módulos del kernel.
Justificación:	A lo largo del desarrollo se implementan módulos en el kernel. Principalmente para la creación del LiveCD
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, GCC, Glibc, Grep, Make, Sed y Zlib	**depmod.-** crea un fichero de dependencias basándose en los símbolos que encuentra en el conjunto existente de módulos del núcleo. A este fichero lo usa *modprobe* para cargar automáticamente los módulos necesarios. **generate-modprobe.conf.-** crea un fichero *modprobe.conf* a partir de una configuración de módulos. **Insmod.-** instala un módulo dentro del núcleo en ejecución. **lsmod.-** muestra todos los módulos cargados. **modinfo.-** examina un fichero objeto asociado con un módulo del núcleo y muestra la información que pueda encontrar. **modprobe.-** carga automáticamente los módulos necesarios. **rmmod.-** descarga módulos del núcleo en ejecución

Tabla 2.37 Características del paquete Module-Init-Tools

- **Patch**

Utilidad:	Contiene un programa que permite crear o modificar ficheros mediante la aplicación de un archivo parche.
Justificación:	A lo largo de toda la construcción se debe aplicar parches a paquetes. Estos parches corrigen problemas causados como por ejemplo de seguridad o incompatibilidad. Un parche es una lista de diferencias creadas por el programa *diff.*
Dependencias para su instalación:	**Programa Instalados.**
Bash, Binutils, Coreutils, GCC, Glibc, Grep, Make y Sed.	**patch.-** modifica ficheros según lo indicado en un fichero parche.

Tabla 2.38 Características del paquete Patch

- **Psmisc**

Utilidad:	Los programas contenidos en este paquete muestran información de procesos que se encuentran en ejecución.
Justificación:	Este tipo de programas fueron pedidos en el análisis de requerimientos.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Coreutils, GCC, Gettext, Glibc, Grep, Make, Ncurses y Sed	**fuser.-** muestra los números de identificación (PID) de los procesos que usan los ficheros o sistemas deficheros especificados. **killall.-** mata procesos por su nombre. **pstree.-** muestra los procesos en ejecución en forma de árbol.

Tabla 2.39 Características del paquete Psmisc

- **Shadow**

Utilidad:	El paquete *shadow* contiene programas para manejar contraseñas seguras. El paquete incluye utilidades para añadir, modificar, eliminar usuarios y grupos. Establece y cambia contraseñas.	
Justificación:	Todas las utilidades antes mencionadas deben ser parte de cualquier sistema operativo.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías instaladas**
Bash, Binutils, Coreutils, Diffutils, Findutils, Gawk, GCC, Gettext, Glibc, Grep, Make y Sed	**groupadd.-** crea un nuevo grupo con el nombre especificado. **groupdel.-** borra el grupo con el nombre especificado. **groupmems.-** permite a un usuario administrar la lista de miembros de su propio grupo sin necesidad de privilegios de superusuario. **groupmod.-** modifica el nombre o el identificador (GID) de un grupo especificado. **login.-** lo utiliza el sistema para permitir el ingreso de un usuario. **passwd.-** se utiliza para cambiar la contraseña de la cuenta de un usuario o grupo. **su.-** ejecuta un intérprete de comandos sustituyendo los identificadores de usuario y grupo. **useradd.-** crea un nuevo usuario con el	**libshadow**.- contiene funciones usadas por la mayoría de los programas de este paquete.

	nombre especificado o actualiza la información por defecto de un nuevo usuario. **userdel.-** borra la cuenta de usuario indicada. **usermod.-** modifica el usuario indicado.	

Tabla 2.40 Características del paquete Shadow

- **Sysklogd**

Utilidad:	Sus programas registran mensajes del sistema. Por ejemplo los generados por el núcleo si sucede algo inusual.
Justificación:	El usuario de cualquier sistema operativo necesita conocer los mensajes del sistema. Esto para corregir posibles errores por ejemplo de software mal instalado o problemas con un modulo del núcleo.
Dependencias para su instalación:	**Programas Instalados.**
Binutils, Coreutils, GCC, Glibc, Make y Patch.	**klogd**.- es un demonio del sistema que intercepta y registra los mensajes del núcleo. **syslogd**.- registra los mensajes que los programas del sistema ofrecen. Cada mensaje registrado contiene al menos una marca de tiempo y un nombre de máquina y, normalmente, también el nombre del programa.

Tabla 2.41 Características del paquete Sysklogd

- **Sysvinit**

Utilidad:	Este paquete contiene programas para controlar el arranque, ejecución y cierre del sistema. Se lo configura con el fichero */etc/inittab*
Justificación:	Son paquetes para proporcionar funciones como por ejemplo para encender , apagar, reiniciar el equipo, etc.

Dependencias para su instalación:	**Programas Instalados:**
Binutils, Coreutils, GCC, Glibc, Make y Sed	**halt.-** usado para apagar o reiniciar el sistema junto con la opción *shutdown*. **init.-** primer proceso iniciado cuando el kernel inicia el hardware. **last.-** muestra los últimos usuarios conectados y desconectados a través del fichero */var/log/wtmp.* **lastb**.- muestra los intentos fallidos de acceso al sistema. **mountpoint.-** comprueba si el directorio es un punto de montaje. **poweroff.-** le pide al núcleo apagar el sistema y apagar la máquina. **reboot.-** le indica al núcleo que reinicie el sistema. **runlevel.-** muestra los niveles de ejecución anterior y actual a través del archivo */var/run/utmp.* **shutdown.-** provoca el cierre del sistema de forma segura.

Tabla 2.42 Características del paquete Sysvinit

- **Tar**

Utilidad:	Contiene un programa de archivado.
Justificación:	Pedido en el análisis de requerimientos

Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Bison, Coreutils, GCC, Gettext, Glibc, Grep, Inetutils, Make, Patch, Sed y Texinfo	**tar**

Tabla 2.43 Características del paquete Tar

- **Texinfo**

Utilidad:	Este paquete contiene programas para leer, escribir y convertir páginas *info*. Estas páginas son páginas de ayuda. Están organizadas en una estructura de árbol y referencias cruzadas.[30]
Justificación:	Permite visualizar ayudas que el usuario puede necesitar. Es el formato de documentación del proyecto GNU

Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, GCC, Gettext, Glibc, Grep, Make, Ncurses, Patch y Sed	**info**.- lee páginas *info*, son más profundas que una simple explicación de las opciones de un programa **install-info**.- se usa para instalar páginas *info*. **makeinfo**.- convierte documentos fuente Texinfo a páginas info, texto plano, o HTML.

Tabla 2.44 Características del paquete Texinfo

[30] Fuente: http://www.estrellateyarde.org/so/ayuda-en-linux

- **Udev**

Utilidad:	Contiene programas para la creación dinámica de nodos de dispositivo.	
Justificación:	Linux accede a los dispositivos del sistema (hardware) a través de los nodos de dispositivos.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías y ficheros Instaladas**
Binutils, Coreutils, GCC, Glibc y Make	**ata_id.-** proporciona a Udev una cadena única e información adicional paraun controlador ATA. **cdrom_id.-** proporciona a Udev las capacidades de un cotrolador CD-ROM o DVD-ROM. **create_floppy_devices.-** crea todos los tipos posibles de dispositivos de disquete. **edd_id.-** proporciona a Udev el ID EDD de un controlador BIOS de disco. **firmware.sh.-** carga firmware en dispositivos. **path_id.-** proporciona la ruta hardware más corta posible a un dispositivo. **scsi_id.-** proporciona a Udev un identificador SCSI. **udevcontrol.-**configura una serie de opciones para el demonio udevden ejecución. **udevd.-** un demonio que escucha	**libvolume_id** **/etc/udev**.- contiene ficheros de configuración de Udev. Permisos de dispositivos y reglas para la denominación de dispositivos.

	uevents en un conector de red. **udevinfo.-** permite a los usuarios consultar en la base de datos de Udev información sobre cualquier dispositivo actualmente presente en el sistema. **udevmonitor.-** muestra el evento recibido del núcleo y el entorno que Udev envía tras el proceso de reglas.	

Tabla 2.45 Características del paquete Udev

- **Util-linux**

Utilidad:	Contiene utilidades del sistema. Estas utilidades permiten manejar sistemas de ficheros, consolas, particiones, etc.
Justificación:	Estas utilidades fueron pedidas en el análisis de requerimientos. Son parte básica de un sistema operativo basado en Linux.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, E2fprogs, GCC, Gettext, Glibc, Grep, Make, Ncurses, Patch,Sed y Zlib	**arch.-** muestra la arquitectura de la máquina. **cal.- m**uestra un calendario simple. **cfdisk.-** se usa para manipular la tabla de particiones del dispositivo indicado. **fdisk.-** se usa para manipular la tabla de particiones del dispositivo indicado. **look.-** muestra líneas que comienzan con una cadena dada. **mkfs.-** construye un sistema de ficheros en un dispositivo (normalmente una partición del disco duro). **mkswap.- i**nicializa el dispositivo o fichero indicado para usarlo como área de intercambio (swap). **more.-** un filtro para paginar texto pantalla a pantalla.

	mount.- monta el sistema de ficheros de un dispositivo dado en el directorio indicado del árbol de ficheros del sistema. **rename.-** renombra ficheros, sustituyendo la cadena indicada con otra. **renice.-** altera la prioridad de los procesos en ejecución. **sfdisk.-** un manipulador de la tabla de particiones del disco. **swapoff.-** desactiva los dispositivos y ficheros de paginación e intercambio. **swapon.-** activa los dispositivos y ficheros de paginación e intercambio y lista los dispositivos y ficheros en uso. **umount.-** desmonta un sistema de ficheros del árbol de ficheros del sistema. **whereis.-** localiza el binario, las fuentes y la página del manual de un comando

Tabla 2.46 Características del paquete Util-linux

- **Vim**

Utilidad:	Contiene un editor de texto vía consola.
Justificación:	Este paquete fue pedido en el análisis de requerimientos. Vi es muy utilizado para la creación y edición de scripts.
Dependencias para su instalación:	**Programas Instalados.**
Bash, Binutils, Coreutils, Diffutils, GCC, Glibc, Grep, Make, Ncurses y Sed	**vi**

Tabla 2.47 Características del paquete Vim

2.2.1.3 Liberación de espacio

Se eliminarán símbolos de depuración. Esos símbolos aparecen al compilar librerías y programas. Los símbolos hacen más grandes a los programas y librerías. La diferencia en tamaño entre programas con y sin símbolos de depuración es generalmente de 5 a 2. La depuración se hace con la ayuda del comando *strip.*

Seguido a la eliminación de los símbolos de depuración, se procede a eliminar el directorio */tools* y sus subdirectorios debido a que los programas que se utilizaron para formar el sistema temporal ya no son necesarios.

2.2.1.4 Hacer arrancable el sistema [4]

En esta sección se procederá a explicar cómo instalar y configurar el paquete LFS-Bootscripts. Este paquete contiene los scripts necesarios que provee la metodología LFS para hacer arrancable el sistema. Aunque esta metodología provee de extensos guiones que funcionan según lo que LFS especifica, es necesario modificar algunos de ellos según el hardware que se posee en los laboratorios de la carrera.

2.2.1.4.1 Sysvinit

Siguiendo los estándares de implementación de un sistema GNU/Linux, se implementa un sistema de inicio basado en SysVinit. Este posee un concepto de niveles de ejecución. En *SysVinit* (o simplemente *init*) existen siete niveles de ejecución (numerados del 0 al 6). Estos niveles indican al sistema qué debe hacer el momento de arrancar. Mientras no se especifique lo contrario, el nivel por defecto en ejecutarse es el nivel 3.

A continuación se describen los niveles de ejecución:

0	Parada del sistema
1	Modo monousuario
2	Modo multiusuario sin red
3	Modo multiusuario con red
4	Reservado para personalizar, caso contrario, es igual que el modo 3
5	Se utiliza para iniciar el entorno gráfico mediante *xdm*
6	Reinicio del sistema

Tabla 2.48 Niveles de ejecución de SysVinit

Es importante destacar que en la estructura de ficheros de GNU/Linux dentro del directorio */etc/rc.d* existen directorios *rc?.d* donde *?* es un número que indica el nivel de ejecución. Además existe paralelamente el directorio *rcsysinit.d,* el cual contiene un grupo de enlaces simbólicos que en su nombre poseen un número de 2 cifras los cuales indican el orden de ejecución. Los enlaces simbólicos mencionados anteriormente no son otra cosa que una representación por niveles de ejecución de los guiones ubicados en /etc/rc.d/init.d/

2.2.1.4.2 Manejo de disposivos

De los paquetes instalados en la sección 2.2.1.3.2 se destaca al paquete *Udev*, el cual provee un método dinámico de creación de dispositivos (o nodos). Normalmente un sistema GNU/Linux utiliza un método estático de creación de dispositivos, los cuales son creados en el directorio */dev/* sin tomar en cuenta si existe o no físicamente el dispositivo creado, ya sea un lector de cd's, un disco duro, una memoria flash, un dispositivo de red, etc.

La creación estática de dispositivos se la hace mediante el guión *makedev*, el cual contiene llamadas al comando *mknod.* Con el uso de *Udev* se crean solamente nodos detectados por el kernel del sistema operativo. Los nodos detectados solamente se crean cada vez que se inicia el sistema y se almacenan en un sistema de archivos temporal *tmpfs*, el cual existe solamente en memoria RAM. Los nodos creados son de espacio reducido, por lo tanto no hace uso de mucha memoria.

Esto hace un uso óptimo de memoria. La cual es uno de los puntos críticos al momento de diseñar el sistema, ya que dentro del análisis de hardware disponible se corrobora que no todas las computadoras de los laboratorios poseen la misma capacidad de memoria RAM.

Udev además provee manejo dinámico de eventos para creación de nodos no solamente al instante de iniciar el sistema, sino también eventos posteriores al inicio del sistema como: inserción de memoria flash, conexión (o desconexión) del cable de red, etc.

Para una referencia más completa de *Udev* se puede consultar:

http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev-FAQ

2.2.1.4.3 Configuración de guiones necesarios

- ***Consola de Linux***

Entre los guiones necesarios de arranque de Linux se encuentran: CONSOLE y CONSOLELOG. Estos establecen el mapa del teclado, la fuente con la cual se presenta el texto en consola, y el nivel de registro de la consola del kernel. Como fichero de configuración se utiliza */etc/sysconfig/console* para *console* y *consolelog*.

En el caso de esta distribución GNU/Linux creada se usa el siguiente esquema:

```
KEYMAP="es"
FONT="lat1-16 –m 8859-1"
```

Estas configuraciones representan la disposición de teclas de los teclados usados comúnmente en los laboratorios de redes.

- ***Ficheros de inicio para el intérprete de comandos BASH***

Los ficheros del directorio */etc* proporcionan los ajustes básicos y necesarios que ayudan a crear y configurar el entorno global de trabajo. Los ficheros leídos al momento de iniciar el sistema básico son */etc/profile*, *~/.bashrc* y *~/.bash_profile*. En el archivo */etc/profile* se encuentran variables de entorno necesarios para el soporte de idioma nativo.

- ***Configuración del guión localnet***

Se procede a asignar un nombre a la máquina, para ello se procede a crear el archivo */etc/sysconfig/network*. En el archivo se especifica el nombre del host. Quedando el contenido del archivo como:

```
HOSTNAME=tesis
```

En este archivo no es necesario especificar el nombre completo del host ni incluir su dominio. Para ello se tiene el archivo */etc/hosts*

- ***Configuración del archivo /etc/hosts.***

Siguiendo los requerimientos de la construcción de la distribución GNU/Linux, es necesario crear el archivo */etc/hosts*; se deberá proveer una configuración que siga el siguiente esquema:

```
dirección_IP mi_pc.mi_dominio.org alias
```

Para el caso de la distribución anfitriona al no poseer un dominio, se puede obviarlo y quedaría:

```
127.0.0.1 localhost
192.168.0.200 tesis
```

Esto asigna una dirección IP fija a la distribución de GNU/Linux, y le provee de un *alias* para ser invocado posteriormente ya sea por su *IP* o por su *alias.* Esta dirección *IP* se utiliza únicamente en esta fase de la construcción. En la sección 2.2.2.1 se incluye el cliente *dhcp* que sirve para obtener una IP automáticamente dentro de una red.

- ***Configuración del guión network***

Luego de asignar una *IP*, es necesario configurar el dispositivo físico de red que se usará inicialmente.

Una de las desventajas de Udev y los controladores de red modulares es la numeración de las interfaces de red no persistente. Con cada reinicio los controladores se cargan en paralelo, por tal motivo se generaría incongruencias al momento de iniciar Linux en una máquina que posea dos o más interfaces de red. Para evitar este problema, Udev contempla un guión y algunas reglas para poder

asignar nombres estables a los dispositivos de red, basándose en las direcciones MAC de los dispositivos.

Primero se pregenera las reglas para asegurar que en cada arranque de Linux se asignen los mismos nombres a los mismos dispositivos. Para ellos se ejecuta:
/lib/udev/write_net/rules all_interfaces

Este comando genera el archivo: */etc/udev/rules.d/70-persistent-net.rules.* Para conocer qué nombre se le asignó a qué dispositivo de red, se puede ingresar al archivo y verificar.

Luego, se crean los archivos de configuración de interfaz de red. Para lograr activar los dispositivos o interfaces de red se debe crear dentro del directorio */etc/sysconfig/network/network-devices* un subdirectorio por cada interfaz a configurar, del tipo *ifconfig.xyz*, donde '*xyz*' es el nombre de la interfaz; por ejemplo *'eth0'.*

Para el caso de la construcción del sistema GNU/Linux se crea el subdirectorio: */etc/sysconfig/network/network-devices/ifconfig.eth0*. Aquí se crea el archivo *ipv4* que contendrá la siguiente configuración:

```
ONBOOT=yes
SERVICE=ipv4-static
IP=172.16.164.130
GATEWAY=172.16.237.1
PREFIX=24
BROADCAST=172.16.237.255
```

Los parámetros descritos fueron los necesarios para poder configurar el host conectado en la Poli-Red. Las direcciones IP podrán ser cualquiera siempre y cuando cumplan con los requerimientos de la red a la cual se conecta inicialmente el host el momento de construir el sistema.

Adicionalmente es necesario crear el archivo */etc/resolv.conf* en el cual consta la o las direcciones IP del o los servidores DNS. Quedando así:

```
search localdomain
nameserver 172.16.164.2
```

Todas estas configuraciones fueron necesarias únicamente en esta fase del proyecto. Esto para poder probar la funcionalidad de la interfaz de red.

2.2.1.4.4 Sistema LFS arrancable

En esta sección se explican los pasos para hacer arrancable el sistema GNU/Linux. Para ello es necesario crear el archivo *fstab*, construir el kernel del nuevo sistema. Y la instalación de un gestor de arranque, el cual permitirá arrancar el sistema.

- ***Creación del fichero fstab***

El archivo */etc/fstab* es la tabla de sistemas de archivos. Es utilizado por la mayoría de programas del sistema Linux para conocer dónde deben montarse los sistemas de archivos. También en *fstab* se especifica el orden en el que deben montarse dichos volúmenes. El archivo de configuración *fstab* queda de la siguiente manera:

```
#file-system   mount-point    type       options           dump     fsck order
/dev/sda2      /              ext3       defaults          1        1
/dev/sda1      swap           swap       pri=1             0        0
proc           /proc          proc       defaults          0        0
sysfs          /sys           sysfs      defaults          0        0
devpts         /dev/pts       devpts     gid=4,mode=620    0        0
shm            /dev/shm       tmpfs      defaults          0        0
```

2.2.1.4.5 Compilación del núcleo [4] [6] [7]

Luego de preparar la tabla de sistemas de archivos (*fstab*) se procede con la compilación del kernel o núcleo del sistema. Debido a la metodología seguida, se utiliza el kernel Linux-2.6.22.5

Para preparar la compilación del kernel se realizan los siguientes pasos:

```
make mkproper
```

Para limpiar el árbol del núcleo. Recomendación por parte de los desarrolladores del kernel

```
make menuconfig
```

Permite ingresar a un menú gráfico más amigable para la configuración de compilación del kernel.

```
make
```

Compila la imagen del núcleo, así como sus módulos.

```
make modules_install
```

Permite instalar los módulos creados en el paso anterior.

Luego de completar la compilación es necesario copiar los nuevos archivos a la partición del nuevo GNU/Linux. Esto se lo realiza con los siguientes comandos:

```
cp arch/i386/boot/bzImage /boot/kernel-2.6.22.5
cp System.map /boot/System.map-2.6.22.5
```

```
cp .config /boot/config-2.6.22.5
```

Donde *bzImage* es la nueva imagen del kernel compilado. Se lo debe copiar al directorio /boot, de preferencia asignándole un nombre que identifique la versión del kernel.

System.map es un archivo de símbolos para el kernel. Permite realizar la traslación de las funciones en el API del kernel, así como las direcciones de las estructuras de datos del kernel para el kernel en ejecución.

Por último, el archivo *.config* contiene la configuración del núcleo creado. También se recomienda guardar el archivo debido a que más adelante es necesario recompilar el kernel.

2.2.1.4.6 Gestor de arranque

Finalmente para que el nuevo sistema sea arrancable se debe contar con un gestor de arranque. Apegándose a la metodología LFS se toma el gestor GRUB.

Se inicia el intérprete de comandos de GRUB por medio del comando *grub*, cabe aclarar que GRUB utiliza una estructura de nombres distinta para discos duros de la forma *(hdn,m)*, donde n es el número de disco duro y m es el número de partición, comenzando ambos desde 0. Para instalar el GRUB en el MBR (Master Boot Record) se procede con el comando:

```
setup (hd0)
```

Luego, se sale del intérprete de comandos del GRUB con el comando *quit*. Se debe crear un archivo que brinde la configuración del menú de arranque del GRUB. El archivo se ubica en el directorio */boot/grub/menu.lst* con el siguiente contenido:

```
default 0
timeout 30
color green/black light-green/black
title TESIS
root (hd0,1)
kernel /boot/kernel-2.6.22.5 root=/dev/sda2
```

Se crea un enlace simbólico del archivo *menu.lst* en */etc/grub*. Esto se lo hace apegándose al estándar FHS. Como pasos finales se ejecuta los siguientes comandos:

Se sale del entorno *chroot*

```
logout
```

Se desmonta los sistemas de archivos virtuales:

```
umount $LFS/dev/pts
umount $LFS/dev/shm
umount $LFS/dev
umount $LFS/dev/proc
umount $LFS/dev/sys
```

Se desmonta el sistema de archivos del LFS:

```
umount $LFS
```

Y se procede con el primer reinicio del sistema:

```
shutdown –r now
```

El flamante sistema básico GNU/Linux arranca por primera vez y con independencia del sistema anfitrión utilizado.

Una vez que se comprueba que el sistema básico GNU/Linux arranca, entonces se procede a realizar la implementación de herramientas y paquetes complementarios para cumplir con los requerimientos de software obtenidos de las encuestas.

La configuración presentada es la que se usó el momento de la compilación del kernel. Se incluye principalmente la compilación para el hardware de los laboratorios de la carrera de Electrónica y Redes.

2.2.2 INSTALACIÓN DE HERRAMIENTAS REQUERIDAS Y COMPLEMENTARIAS DEL SISTEMA

En esta sección se describirán todos los procesos a seguir después de que el sistema básico arrancó por primera vez. Esto corresponde a la segunda etapa de la metodología LFS. Se detallará la utilidad de los programas pedidos en el análisis de requerimientos. También se explicará la utilidad de paquetes complementarios del sistema. Estos paquetes no formaron parte del sistema Linux básico creado. A continuación se pueden observar los procesos involucrados

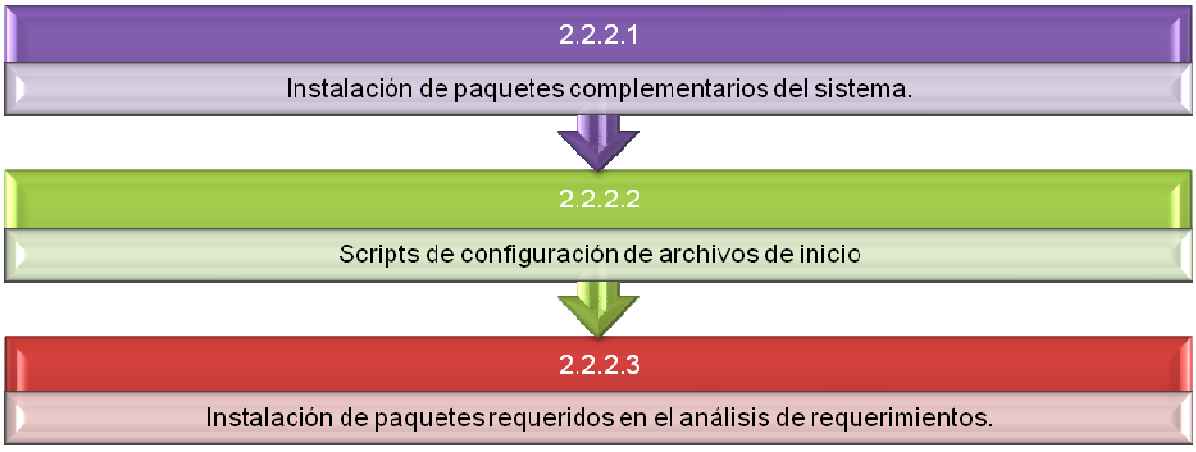


Figura 2.8 Procesos involucrados en la instalación de herramientas requeridas y complementarias

2.2.2.1 Instalación de paquetes complementarios del sistema [5]

En esta sección se describen herramientas que complementan la funcionalidad del sistema. Las herramientas instaladas brindan transparencia al usuario para el manejo de periféricos. Además existen otras herramientas básicas para el uso de la red e internet como lo son: dhcpd y wget.

A continuación se describen los paquetes instalados.

- **Autofs**

Utilidad:	Es una utilidad que automonta sistemas de archivos. Trabaja directamente con el kernel. Automonta y desmonta sistemas de archivos removibles.
Justificación:	Muy útil para el auto montaje y desmotaje de memorias flash. Estos dispositivos de almacenamiento son utilizados por todos los estudiantes.

Programas Instalados	**Librerías y ficheros Instalados**
automount.- este demonio se encarga del auto montaje.	Instala librerías que sirven para el auto montaje y desmontaje de sistemas de ficheros. Para mayor información refiérase a: http://www.autofs.org/ Fichero instalado.- /lib/autofs y var/run/autofs

Tabla 2.49 Características del paquete Autofs

- **PCI Utilities**

Utilidad:	Contiene utilidades para listar, inspeccionar el estado y establecer los registros de configuración de dispositivos PCI.
Justificación:	Es utilizado ya que alrededor de 10 PCs del laboratorio manejan dispositivos periféricos PCI.

Programas Instalados	**Librerías y ficheros Instalados**
lspci.- despliega información de dispositivos PCI **setpci**.- consulta y configura dispositivos PCI.	**libpci.a**

Tabla 2.50 Características del paquete PCI Utilities

- **Usbutils**

Utilidad:	Este paquete contiene utilidades para desplegar información y configurar dispositivos usb conectados al sistema.
Justificación:	Es una utilidad básica del sistema. En el laboratorio la mayor parte de las PCs utilizan dispositivos usb.

Dependencias para su instalación:	**Programas Instalados.**
libusb	**lsusb**

Tabla 2.51 Características del paquete Usbutils

- **Dhcpd**

Utilidad:	Este paquete contiene una implementación de cliente DHCP**.**
Justificación:	Permite al usuario obtener una dirección IP automáticamente del servidor DHCP. En el laboratorio se encuentra implementado este servidor. Evita que el usuario configure siempre la dirección IP
Programas Instalados	
Dhcpd	

Tabla 2.52 Características del paquete Dhcpd

- **Wget**

Utilidad:	Este paquete contiene una utilidad para descargar archivos de internet.
Justificación:	Es utilizado para descargar los archivos que contienen el código fuente de los programas. Simplifica el proceso de descarga al tener la dirección de los archivos fuente.
Programas Instalados	
Wget	

Tabla 2.53 Características del paquete Wget

2.2.2.2 Scripts de configuración de archivos de inicio [5]

Luego de haber seguido la metodología LFS y haber obtenido como resultado un sistema básico e independiente de la distribución anfitriona, se procede a realizar implementación de herramientas requeridas y aspectos complementarios del sistema. Para ello se empieza por configurar el entorno *bash* de trabajo.

Como primera tarea la metodología BLFS recomienda crear un dispositivo de recuperación de arranque (boot recovery). Debido a que se trabaja en un ambiente de máquina virtual, la técnica utilizada de recuperación es volver a copias de seguridad anteriores de la máquina virtual. Utilizando esta técnica no se justifica realizar la creación del dispositivo de recuperación de arranque.

La implementación del Shell de comandos usa una colección de archivos de inicio que ayudan a crear un ambiente de trabajo funcional.

El proceso de inicio del Shell es de la siguiente manera: Primero se inicia el proceso de *login* de un usuario, usando */bin/login*, leyendo el archivo */etc/passwd*. Durante este proceso el Shell lee el archivo */etc/profile* y sus equivalentes archivos privados *~/.bash_profile* y *~/.bashrc.*

Los archivos de configuración se presentan a continuación:

- ***/etc/profile***

En este archivo de configuración se especifica funciones de configuración para el manejo del PATH de trabajo para los usuarios del sistema operativo. Adicionalmente se configura el PATH inicial de trabajo. Se configura la longitud del registro del historial de comandos del *bash*. Todo lo explicado se muestra en la captura de la figura 2.9.

```
# Begin /etc/profile
# Written by James Robertson
# Modifications by Alexander Santamaria & Cristian Giler

# System wide environment variables and startup programs.

# System wide aliases and functions should go in /etc/bashrc. Personal
# environment variables and startup programs should go into
# ~/.bash_profile. Personal aliases and functions should go into
# ~/.bashrc.

# Functions to help us manage paths. Second argument is the name of the
# path variable to be modified (default: PATH)
pathremove () {
        local IFS=':'
        local NEWPATH
        local DIR
        local PATHVARIABLE=${2:-PATH}
        for DIR in ${!PATHVARIABLE} ; do
                if [ "$DIR" != "$1" ] ; then
                  NEWPATH=${NEWPATH:+$NEWPATH:}$DIR
                fi
        done
        export $PATHVARIABLE="$NEWPATH"
}

pathprepend () {
        pathremove $1 $2
        local PATHVARIABLE=${2:-PATH}
        export $PATHVARIABLE="$1${!PATHVARIABLE:+:${!PATHVARIABLE}}"
}

pathappend () {
        pathremove $1 $2
        local PATHVARIABLE=${2:-PATH}
        export $PATHVARIABLE="${!PATHVARIABLE:+${!PATHVARIABLE}:}$1"
}

# Set the initial path
export PATH=/bin:/usr/bin

if [ $EUID -eq 0 ] ; then
        pathappend /sbin:/usr/sbin
        unset HISTFILE
fi

# Setup some environment variables.
export HISTSIZE=1000
--Más--
```

```
export HISTIGNORE="&:[bf]g:exit"

# Setup a red prompt for root and a green one for users.
NORMAL="\[\e[0m\]"
RED="\[\e[1;31m\]"
GREEN="\[\e[1;32m\]"
if [[ $EUID == 0 ]] ; then
  PS1="$RED\u [ $NORMAL\w$RED ]# $NORMAL"
else
  PS1="$GREEN\u [ $NORMAL\w$GREEN ]\$ $NORMAL"
fi

for script in /etc/profile.d/*.sh ; do
        if [ -r $script ] ; then
                . $script
        fi
done

# Now to clean up
unset pathremove pathprepend pathappend

#export LANG=es_EC.iso88591

# export some configuration for GNOME,
# written by Alexander Santamaría & Cristian Giler
# for the implementation of LINUX for EPN-REDES
export XORG_PREFIX=/usr
export XORG_CONFIG="--prefix=$XORG_PREFIX --sysconfdir=/etc --mandir=$XORG_PREFI
X/share/man --localstatedir=/var"
export GNOME_PREFIX=/usr
export XDG_CONFIG_DIRS=/etc/gnome/2.18.3/xdg:/etc/xdg
export XDG_DATA_DIRS=$GNOME_PREFIX/share:/usr/local/share:/usr/share
export LIBGLADE_MODULE_PATH=$GNOME_PREFIX/lib/libglade/2.0

# End /etc/profile
root [ ~ ]# _
```

Figura 2.9 Captura de /etc/profile

- ***/etc/bashrc***

En este archivo se configura los colores del *prompt* del *bash*. Se asigna el color rojo cuando el usuario logeado es el *root*. Para cualquier otro usuario que se loguee se asigna al *prompt* el color verde. A continuación se muestra el contenido del archivo:

```
root [ ~ ]# cat /etc/bashrc
# Begin /etc/bashrc

alias ls='ls --color=auto'

NORMAL="\[\e[0m\]"
RED="\[\e[1;31m\]"
GREEN="\[\e[1;32m\]"
if [[ $EUID == 0 ]] ; then
  PS1="$RED\u [ $NORMAL\w$RED ]# $NORMAL"
else
  PS1="$GREE\u [ $NORMAL\w$GREEN ]\$ $NORMAL"
fi

# End /etc/bashrc
root [ ~ ]# _
```

Figura 2.10 Captura de /etc/bashrc

- ***~/.bashrc***

En este archivo se incluye las configuraciones globales del *bash* para ser aplicado a los usuarios actuales del sistema y a usuarios nuevos que posteriormente se creen en el sistema. Seguidamente se muestra el archivo implementado:

```
root [ ~ ]# cat ~/.bashrc
# Begin ~/.bashrc
# Written by James Robertson
# Modyfied by Alexander Santamaria & Cristian Giler

# Personal aliases and functions.

# Personal environment variables and startup programs should go in
# ~/.bash_profile. System wide environment variables and startup
# programs should go in /etc/profile. System wide aliases and functions are
# in /etc/bashrc.

if [ -f "/etc/bashrc" ] ; then
  source /etc/bashrc
fi

# End ~/.bashrc
root [ ~ ]# _
```

Figura 2.11 Captura de ~/.bashrc

- ***~/.bash_profile***

A diferencia del archivo anterior (*.bashrc*), éste contiene información de configuración del *bash* exclusiva para cada usuario. Existe un archivo .bash_profile por cada

usuario del sistema. En este archivo se incluyen las configuraciones propias o requeridas para cada usuario, creando independencia del resto de usuarios. Estas configuraciones sólo afectan al usuario del cual pertenece el archivo. Por ejemplo, para un usuario llamado *cgiler* se contará con el archivo */home/cgiler/.bash_profile*. Para el usuario llamado *asantamaria* se contará con el archivo */home/asantamaria/.bash_profile.* En la figura 2.12 se muestra la captura del archivo.

- **~/.bash_profile**

```
# Begin ~/.bash_profile
# Written by James Robertson
# Modifications by Alexander Santamaria & Cristian Giler

# Personal environment variables and startup programs

# Personal aliases and functions should go in ~/.bashrc. System wide
# environment variables and startup programs are in /etc/profile.
# System wide aliases and functions are in /etc/bashrc.

append () {
        # First remove the directory
        local IFS=':'
        local NEWPATH
        for DIR in $PATH ; do
                if [ "$DIR" != "$1" ] ; then
                  NEWPATH=${NEWPATH:+NEWPATH:}$DIR
                fi
        done
        # Then append the directory
        export $PATH=$NEWPATH:$1
}
if [ -f "$HOME/.bashrc" ] ; then
  source $HOME/.bashrc
fi

if [ -d "$HOME/bin" ] ; then
  append $HOME/bin
fi

unset append

# End ~/.bash_profile
root [ ~ ]#
```

Figura 2.12 Captura de ~/.bash_profile

Uno de los objetivos del desarrollo del presente proyecto es optimizar el uso de espacio en disco duro en la etapa de desarrollo. Esto se refleja al final como menos espacio ocupado al instante de crear el LiveCD.

En la metodología de BLFS se sugiere la compresión de los archivos correspondientes a los manuales de cada comando o aplicación, llamadas *Man Pages* e *Info Pages.* Los programas lectores de Man e Info Pages pueden procesar transparentemente archivos comprimidos en *gzip* o *bzip2*, esta funcionalidad es explotable para poder liberar espacio en disco duro.

Para realizar la compresión de las Man e Info Pages se presenta el script llamado *compressdoc*, ubicado en */usr/sbin/.* Este script fue creado por una gran cantidad de desarrolladores debido a su complejidad. Este script puede ser encontrado en el Anexo D.

2.2.2.3 Instalación de paquetes pedidos en el análisis de requerimientos

En esta sección se describe la utilidad y dependencias de todos los programas pedidos. Se encuentran separados por materia. La instalación de todos los programas de esta sección tiene su justificación en el análisis de requerimientos.
Se muestran como dependencias únicamente los programas adicionales a los instalados como parte del sistema básico.

2.2.2.3.1 Gestión de Redes

- **Netsnmp [17]**

Utilidad:	Provee una suite de aplicaciones para implementar SNMP v1, SNMP v2c y SNMP v3. Permite el uso de IPv4 e IPv6.
Programas Instalados[31]	
snmptranslate.- traduce los OIDs[32] de las MIBs[33] entre formatos numérico y de texto. **snmpget**.- este comando puede ser usado para obtener datos desde un host remoto, se tiene que dar el hostname, información de autenticación y el OID. **snmpgetnext**.- su uso es similar a snmpget, da la información del siguiente OID en el árbol MIB. **snmpwalk**.- està diseñado para ejecutar una cadena de GETNEXT. **snmpset**.- modifica información en un host remoto, se debe especificar el OID que se va a actualizar. **snmptrap**.- los traps son utilizados para notificar condiciones anormales a las estaciones de administración.	

Tabla 2.54 Características del paquete Netsnmp

[31] Fuente: http://net-snmp.sourceforge.net/tutorial/tutorial-5/commands/
[32] OID: es el identificador de objeto que identifica las variables de la MIB a recuperar.
[33] MIB: es la Management Information Base

- **Traceroute**

Utilidad:	Este paquete contiene un programa que es utilizado para desplegar la ruta de red que toman los paquetes para alcanzar un host especificado
Programas Instalados	
Traceroute: muestra todos los saltos que da un paquete para llegar al destino.	

Tabla 2.55 Características del paquete Traceroute

- **Wireshark [5]**

Utilidad:	Este paquete contiene un analizador de protocolos de red. Es conocido como un sniffer. Es útil para analizar el tráfico capturado. Provee una interfaz gráfica y vía consola para analizar paquetes.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías Instaladas**
Glib y libpcap[34]	**capinfos.-** lee un archivo capturado y retorna estadísticas acerca del archivo **dftest.-** es un programa de prueba que muestra filtra y compila. **dumpcap.-** es una herramienta de tráfico de red que permite capturar y guardar los paquetes a un archivo. **editcap.-** edita y/o traduce el formato de los archivos capturados	**libwireshark.so.-** contiene las funciones necesarias por las utilidades del paquete.

[34] libpcap: provee funciones para realizar capturas de paquetes.

	mergecap.- combina múltiples archivos guardados en una sola salida. **tshark.-** es el analizador de red en modo de consola. **wireshark.-** es el analizador de red en modo gráfico.	

Tabla 2.56 Características del paquete Wireshark

Este paquete requiere que el kernel este configurado antes de su instalación de la siguiente manera:

```
# Networking options
#
CONFIG_PACKET=m
```

- **Fping** [18]

Utilidad:	Este paquete instala únicamente el programa fping. Fping utiliza mensajes de echo ICMP para determinar si un host está arriba. Con esta herramienta se puede hacer pings a muchos hosts a la vez. Se puede especificar la lista de hosts de un fichero.
Programas Instalados:	
fping.- utilidad que permite realizar peticiones de eco (echo request) a un grupo de direcciones IP de una misma red.	

Tabla 2.57 Características del paquete Fping

- **Nmap** [5]

Utilidad:	Es una utilidad para exploración de la red. Se utiliza para auditar la seguridad de la misma. Soporta escaneo de ping, de puertos etc.
Programas Instalados	**Directorios Instalados**
nmap.- utilidad para exploración de red y auditar la seguridad de la misma.	/usr/share/nmap

Tabla 2.58 Características del paquete Nmap

2.2.2.3.2 Sistemas Operativos

- **Java JDK (Java Developer Kit)** [5]

Utilidad:	Contiene el entorno de desarrollo de Java. Es útil para el desarrollo de programas en Java. Provee el entorno para ejecutar programas en Java.
Programas Instalados	**Librerías y Directorios Instalados**
Instala programas de la suite JDK. Para mayor información refiérase a: https://jdk6.dev.java.net/	**Librerías**: $JAVA_HOME/lib/*, $JAVA_HOME/jre/lib/*, libjavaplugin_oji.so **Ficheros**: /opt/jdk-6u5

Tabla 2.59 Características del paquete Nmap

2.2.2.3.3 Hardware de Conectividad

- **Apache** [5]

Utilidad:	Contiene un servidor http Open Source.
Programas Instalados	**Librerías y Directorios Instalados**
Instala programas para manejo de contraseñas, el programa del servidor, etc. Para mayor información refiérase a: http://www.apache.org/	**Librerías**: libapr-1.{so,a}, libaprutil-1.{so,a}, /usr/lib/apache/*.so **Directorios**: /etc/apache /srv/www, /usr/include/apache /usr/lib/apache /var/log/apache

Tabla 2.60 Características del paquete Apache

- **FTP (ProFTPD)** [5]

Utilidad:	Contiene un servidor http Open Source.
Programas Instalados	**Directorios Instalados**
proftpd.- es el demonio FTP **ftpcount**.- muestra el número actual de conexiones **ftpshut**.- apaga todos los servidores en un tiempo determinado. **ftptop**.- despliega el estado de las conexiones. **ftpwho**.- despliega la información actual de cada sesión.	/var/run/proftpd

Tabla 2.61 Características del paquete ProFTPD

- **DNS (BIND)** [5]

Utilidad:	Provee utilidades de cliente y servidor DNS.
Programas Librerías y Directorios Instalados	
Instala utilidades por ejemplo para hacer consultas a un servidor DNS, nombres de dominio de internet, etc. Para mayor información refiérase a: http://www.escomposlinux.org/lfs-es/recetas/bind.html	

Tabla 2.62 Características del paquete Bind

- **Minicom**

Utilidad:	Es un programa para emulación de terminal. Se usa para establecer sesiones vía consola con equipos. Utiliza comunicación serial
Programas Instalados:	
Minicom	

Tabla 2.63 Características del paquete Minicom

2.2.2.3.4 Seguridad de redes

- **OpenSSL** [5]

Utilidad:	Contiene herramientas y librerías relacionadas con criptografía.
Programas Instalados	**Librerías y Directorios Instalados**
openssl.- Es una herramienta de línea de comandos. Sirve para utilizar herramientas relacionadas con criptografía	**Librerías**: **libcrypto.{so,a}.-** implementa un amplio rango de algoritmos criptográficos **libssl.{so,a}.-** implementa los protocolos SSL (Secure Sockets Layer)y TLS(Transport Layer Security) **Directorios**: /etc/ssl /usr/include/ssl /usr/lib/engines /usr/share/doc/openssl-0.9.8g

Tabla 2.64 Características del paquete OpenSSL

- **MIT Kerberos** [5]

Utilidad:	Kerberos es un middleware de autenticación. Es un protocolo de autenticación de red. Centraliza la autenticación.
Programas Librerías y Directorios Instalados	
Este paquete instala el servidor de autenticación. Para mayor información refiérase a: http://web.mit.edu/Kerberos/dist/index.html#krb5-1.8	

Tabla 2.65 Características del paquete MIT Kerberos

- **IPSec** [19]

Utilidad:	Es un protocolo que implementa seguridad sobre IP. Soporta IPv4 e IPv6.
Programas Librerías y Directorios Instalados	
Para mayor información refiérase a: http://www.ipsec-howto.org/	

Tabla 2.66 Características del paquete IPSec

2.2.2.3.5 Aplicaciones distribuidas

- **OpenLDAP** [5]

Utilidad:	Provee una implementación Open Source del Lightweight Directory Access Protocol. Este protocolo permite el acceso a un servicio de directorio.	
Dependencias para su instalación:	**Programas Instalados**	**Librerías y Directorios Instalados**
Berkeley DB, OpenSSL	Instala programas para crear, modificar, crear, comparar entradas de directorios, etc. Para mayor información refiérase a: http://www.openldap.org/	**Librerías**: IInstala las librerías que brindan toda la funcionalidad a LDAP. Estas son: liblber.{so,a} libldap.{so,a} y libldap_r.{so,a} **Directorios**: /etc/openldap /srv/ldap /usr/share/openldap

Tabla 2.67 Características del paquete OpenLDAP

- **Heartbeat** [11]

Utilidad:	Es un demonio que provee un servicio de infraestructura de cluster a sus clientes. Permite que el cliente conozca de procesos pares en otras maquinas.
Programas Instalados	
Para obtener mayor información acerca de los programas instalados refiérase a: http://linux-ha.org/wiki/Heartbeat	

Tabla 2.68 Características del paquete Heartbeat

2.3 CREACIÓN DEL LIVECD [12]

Para la construcción de LiveCDs personalizados existen diversas metodologías de creación. Apegándose a las recomendaciones de la comunidad de BLFS se usa los scripts y recomendaciones publicadas en www.linux-live.org. Los scripts publicados aquí se distribuyen bajo licencia GPL de GNU.

Se destacan los siguientes procesos:

Figura 2.13 Procesos involucrados en la creación del LiveCD

2.3.1 REEMPLAZO DEL KERNEL [4] [7]

Procediendo con la construcción del LiveCD es necesario tener (como prerrequisito) construidos los paquetes de sistemas de archivos *aufs (another union file system)* y *squashfs*. Ambos paquetes deben estar implementados como módulos del kernel.y no como parte del kernel. Adicionalmente el paquete squashfs debe estar parchado con el módulo de soporte para LZMA (Lemple-Ziv-Markov Algorithm) que es un algritmo de compresión de datos. El requerimiento de que ambos paquetes estén instalados como módulos del kernel se debe a que LZMA no es compatible con la integración del kernel.

Aunque el kernel usado para la implementación del LiveCD soporta el sistema de archivos squashfs, tanto como parte del núcleo como módulo, se encontró que es necesario cambiar el kernel del sistema operativo de su versión 2.6.22.5 a la 2.6.27.27. Esto se debe a que el kernel 2.6.22.5 no soporta squashfs + lzma cuando se lo implementa como un módulo del kernel.

2.3.2 CONFIGURACIÓN DEL KERNEL [12] [14] [15]

Luego de reemplazar el Kernel se lo debe compilar para que soporte las funciones y módulos necesarios para crear el LiveCD. La metodología indica que las siguientes funciones deben ser compiladas directamente en el kernel:

```
ext2 (CONFIG_EXT2_FS=y)
tmpfs (CONFIG_TMPFS=y)
ramdisk (CONFIG_BLK_DEV_RAM=y)
initrd (CONFIG_BLK_DEV_INITRD=y)
```

Los siguientes módulos deben ser implementados fuera del kernel:

```
loop.ko
isofs.ko
aufs.ko
squashfs.ko
sqlzma.ko
unlzma.ko
```

La documentación referente a la implementación de *squashfs* con *lzma* específica que se requiere al menos la versión 2.6.24.0 del kernel. Para el desarrollo se utiliza la versión 2.6.27.27.

La compilación del nuevo kernel se la realiza con la configuración del kernel anteriormente implementado. Esto se hace para no perder compatibilidad ni funcionalidad de los paquetes de software ya instalados anteriormente durante todo el desarrollo del proyecto. Los pasos para compilar el kernel son los mismos utilizados en la sección 2.2.4.1.5.

2.3.3 LIBERACIÓN DE ESPACIO [4]

Continuando con los pasos para la creación del LiveCD, se procede a liberar espacio de disco por medio de la ejecución del comando *compressdoc*. Esto hace que la distribución LiveCD sea lo suficientemente compacta para que quepa en un CD.

2.3.4 CONSTRUCCIÓN DE DATOS [12]

La construcción de los datos se la hace antes de la creación de la imagen .iso del LiveCD. Se procede a ejecutar el script de construcción y compresión de los directorios que conformarán el LiveCD.

Los lineamientos seguidos por el script son los siguientes:

- Pedir al usuario el nombre con el que se creará la distribución de Linux. Hay que tomar en cuenta que el script sólo podrá ser ejecutado por el usuario root.
- Ubicar la dirección en dónde se encuentra el kernel que se incluirá en la construcción de los archivos necesarios para la creación del LiveCD.
- Copiar los archivos necesarios para el booteo del nuevo sistema.
- Crear la imagen de booteo por medio del comando *initrd*
- Crear los módulos de los directorios del sistema GNU/Linux. Para ello se comprimen los directorios con el algoritmo *squashfs+lzma* generando archivos con terminación .lzma.

En el Anexo F se presenta el script.

2.3.5 CREACIÓN DEL ISO [12]

Un script crea el archivo *.iso* a partir de cuatro pasos, que se explican a continuación:

- Primer paso, solicita al usuario un nombre para crear el archivo *.iso* (*nombre_de_la_distribución.iso*).
- Segundo, solicita en qué dirección del disco duro se va a almacenar el archivo *.iso*.
- Tercero, con ayuda del comando *mkisofs* se crea el archivo de imagen *.iso* que será quemado posteriormente en un CD.
- Como último paso se procede a copiar el archivo *.iso* en un CD. Para ello se copió el archivo en una memoria flash y se procedió a utilizar otro computador con cualquier herramienta de grabado de CD's para hacerlo.

Para proceder con la creación del archivo *.iso* se ejecuta el script que se muestra en el Anexo G.

En las siguientes figuras se presentan una parte del proceso de la creación del LiveCD. Esto es la compresión de los directorios con lzm.

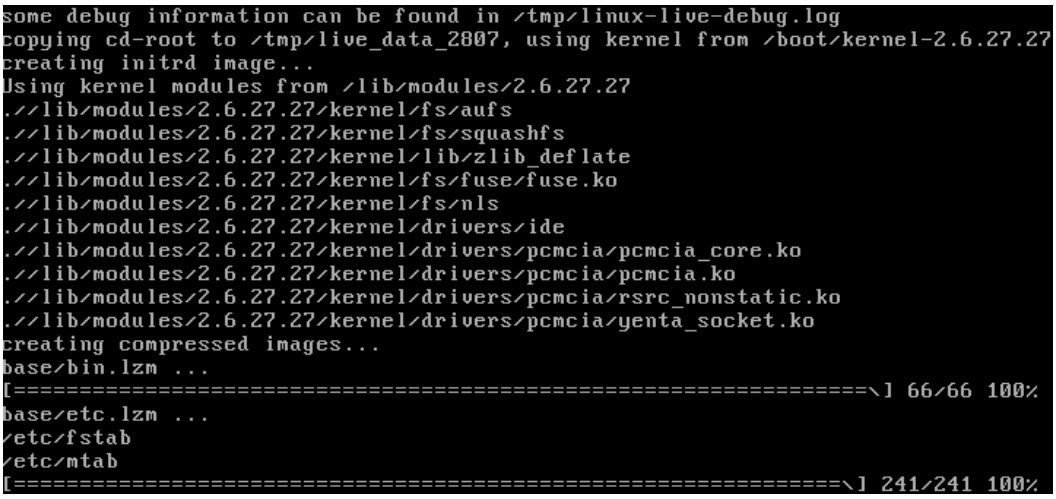


Figura 2.14 Captura de la creación del LiveCD

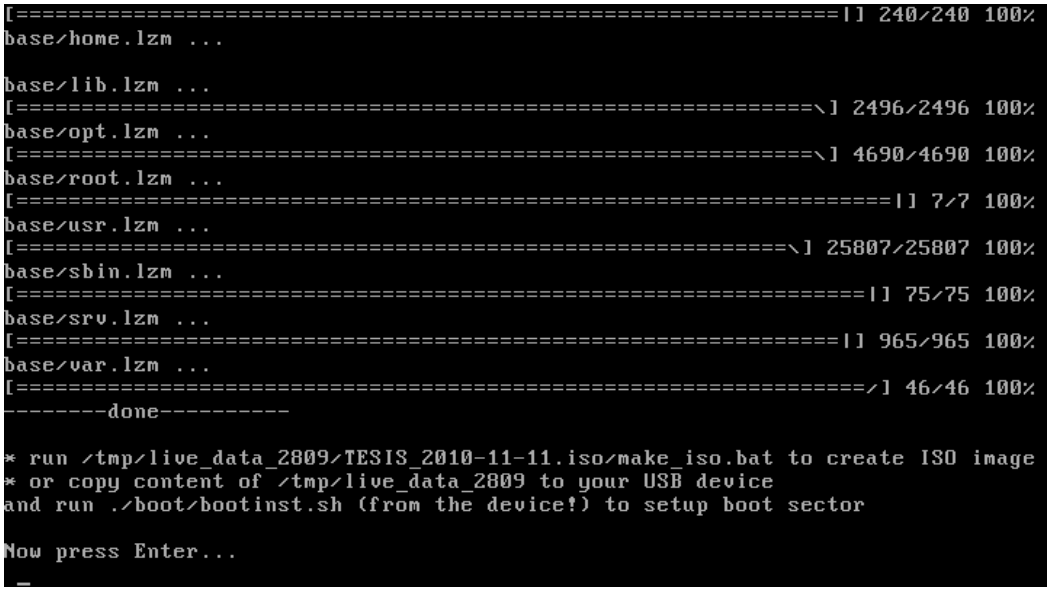


Figura 2.15 Captura de finalización del proceso de creación del live CD.

Finalmente se comprueba el arranque del LiveCD como se muestra en la figura 2.16.

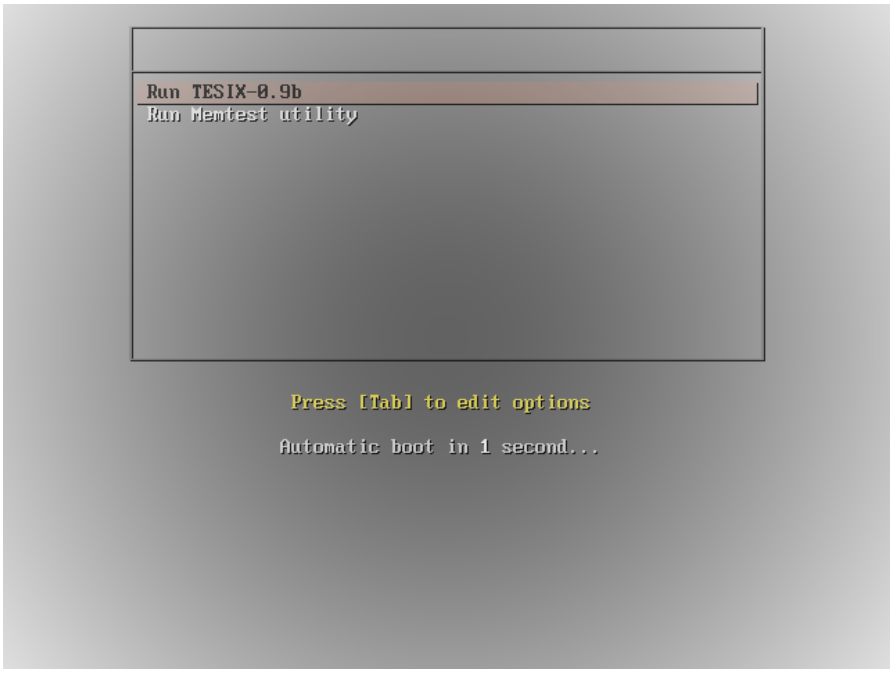


Figura 2.16 Captura del arranque del LiveCD

En la figura 2.17 se muestra el arranque de los módulos del kernel del LiveCD.

Figura 2.17 Captura del arranque de los módulos del kernel del LiveCd

CAPÍTULO 3

PRUEBAS DEL SISTEMA

Esta sección contempla las pruebas para verificar que se cumplan los requerimientos funcionales obtenidos de las encuestas de análisis de requerimientos. Esto es comprobar que cada una de las herramientas requeridas esté trabajando correctamente como se menciona en el alcance del proyecto.

También se verifica si se cumplieron los objetivos planteados por el proyecto, esto es que:

- La distribución cumpla con los requerimientos de espacio en CD.
- La distribución no demore en arrancar.

Se comprueba la compatibilidad con los dispositivos de hardware definidos en el alcance del proyecto: interfaces del tipo: USB, serial, y de red, además de monitores del tipo LCD y CRT, procesadores de 32-bits Intel y AMD.

Se dividirán a las pruebas en dos grupos: pruebas para verificar que se cumpla el alcance del proyecto y pruebas para verificar que las herramientas requeridas por cada materia funcionen correctamente. Para la realización de estas pruebas se define un protocolo sistemático. La información de cada prueba se la muestra en tablas que contienen la información que se indica en la tabla 3.1 que se aprecia a continuación:

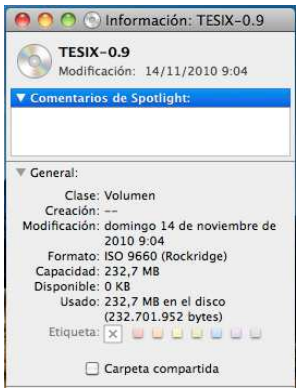
Nombre de la prueba	
Objetivo de la prueba	
Se explica el objetivo de la prueba	
Topologías/Equipos para la prueba	
Dependiendo de la prueba se incluye la topología o la configuración de hardware de los equipos utilizados.	
Procedimiento para realizar la prueba:	
Se numeran y explican los pasos para realizar la prueba.	
Resultados Esperados:	
Se describen los resultados que se esperan en la prueba.	
Resultados Obtenidos:	
Se describen los resultados que se obtienen en la prueba.	
Análisis de Resultados:	
Se analizan los resultados obtenidos.	

Tabla 3.1 Estructura de las tablas para mostrar el procedimiento de pruebas

3.1 PRUEBAS PARA VERIFICAR EL CUMPLIMIENTO DEL ALCANCE DEL PROYECTO

En esta sección se realizan pruebas para corroborar que se cumpla con el alcance del proyecto.

3.1.1 ESPACIO EN DISCO QUE OCUPA EL SISTEMA

Nombre de la Prueba	Espacio en disco que ocupa el sistema
Objetivo de la Prueba	
Determinar que el sistema final cumpla los requerimientos de espacio en CD.	
Equipo para la prueba	
 ▪ 1 x laptop o computadora personal ▪ 1 x copia del LiveCD del sistema	
Procedimiento para realizar la prueba:	
1.- Insertar el LiveCD mientras corre el sistema un sistema operativo anfitrión. 2.- Dentro de las propiedades del LiveCD medir el espacio que el sistema ocupa en el disco.	
Resultados Esperados:	
1.- El sistema creado debería utilizar un espacio menor o igual a la capacidad de un CD (700Megabytes).	
Resultados Obtenidos:	
1.- El sistema ocupa 232.7 MB como se muestra en la figura: 	

Análisis de Resultados:
El sistema cumple el objetivo de requerimiento de espacio en un CD. El producto final ocupa alrededor de la tercera parte del espacio disponible en un CD. En cuanto al uso de disco duro, el sistema no requiere espacio en el disco duro del computador usado, ya que no es necesario realizar una instalación del sistema operativo.

Tabla 3.2 Prueba, Espacio que ocupa el sistema

3.1.2 TIEMPO DE ARRANQUE DEL SISTEMA

Nombre de la prueba	Tiempo de arranque del sistema
Objetivo de la Prueba	
Medir el tiempo de arranque del sistema utilizando 4 computadoras con configuraciones de hardware diferentes.	
Topologías/Equipos para la prueba	
▪ 1 x copia del LiveCD del sistema. ▪ 4 x computadoras M1: 1 x laptop (procesador 1.8GHz con 2 núcleos, 3 GB de memoria RAM) M2: 1 x PC del laboratorio de Redes M3: 1 x maquina virtual (procesador 2.2 GHz con 1 núcleo, 256 MB de memoria RAM) M4: 1 x maquina virtual (procesador 2.2 GHz con 2 núcleos, 256 GB de memoria RAM) M5: 1 x laptop (procesador 1.6 GHz con 1 núcleo, 512 MB de memoria RAM) ▪ 1 x cronómetro	
Procedimiento para realizar la prueba:	
1.- Arrancar el sistema en la primera computadora.	

2.- Cronometrar el tiempo que demora en arrancar el sistema operativo

3.- Repetir el proceso con las 3 computadoras restantes.

4.- Comparar los resultados obtenidos y explicarlos.

Nota: El tiempo es medido desde que inicia la computadora hasta que el sistema pide el ingreso de usuario y contraseña. Sin contar el tiempo inherente del arranque del BIOS del propio del hardware

Resultados Esperados:

1.- El sistema creado debe arrancar en un tiempo menor a 5 minutos, que equivale aproximadamente al 8% del tiempo de una (1) hora clase.

Resultados Obtenidos:

Los resultados obtenidos se muestran en la siguiente tabla:

	M1	**M2**	**M3**	**M4**	**M5**
Tiempo de arranque en segundos (s).	48	48	48	50	52

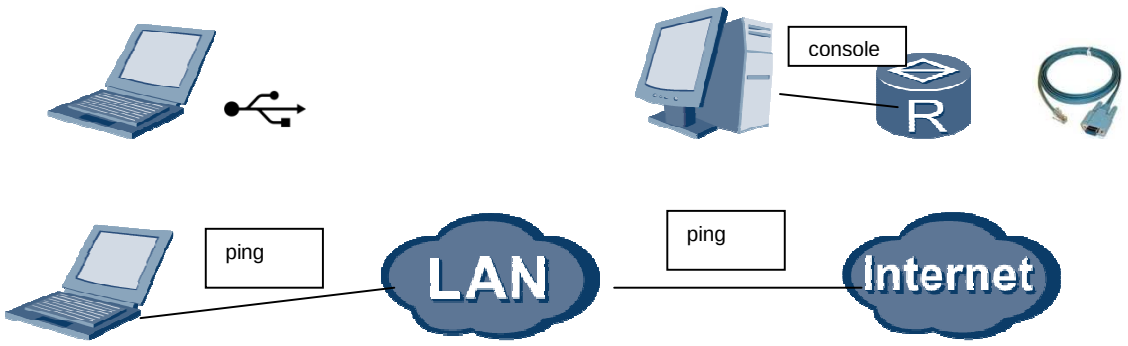
Análisis de Resultados:

El tiempo de arranque depende de las características de hardware de la máquina en la que se arranca el sistema. Se puede observar en la tabla 3.3 que el sistema tiene un tiempo máximo de arranque de 52 segundos. El ususario no espera demasiado tiempo en que el sistema arranque y se encuentre totalmente funcional. De los valores obtenidos se observa que se disminuye notablemente el tiempo esperado de arranque, superando en gran medida las expectativas del proyecto.

Tabla 3.4 Prueba, Tiempo de arranque del sistema

3.1.3 COMPATIBILIDAD CON INTERFACES

Nombre de la prueba	Compatibilidad con interfaces
Objetivo de la Prueba	
Comprobar que el sistema es compatible con interfaces de tipo usb, serial y de red.	
Topologías/Equipos para la prueba	



- 1 x laptop
- 1 x PC del laboratorio de Redes
- 1 x copias del LiveCD del sistema

Procedimiento para realizar la prueba:

1.- Arranque el sistema en la primera computadora.

2.- Probar que la interfaz usb funcione al insertar una memoria flash y verificar su contenido.

3.- Probar que la interfaz serial funcione correctamente al establecer exitosamente una conexión por consola con un router o switch del laboratorio de Redes. Esta conexión debe hacerse por medio del programa Minicom instalado como parte del análisis de requerimientos para la materia Hardware de Conectividad.

4.- Probar que la interfaz de red funcione conectando el computador a la red y verificando la conexión con la misma. Realizar las pruebas de conectividad con el comando *ping* y verficar la configuración de la red con el comando *ifconfig*.

5.- Repetir este proceso con la otra computadora.

Resultados Esperados:

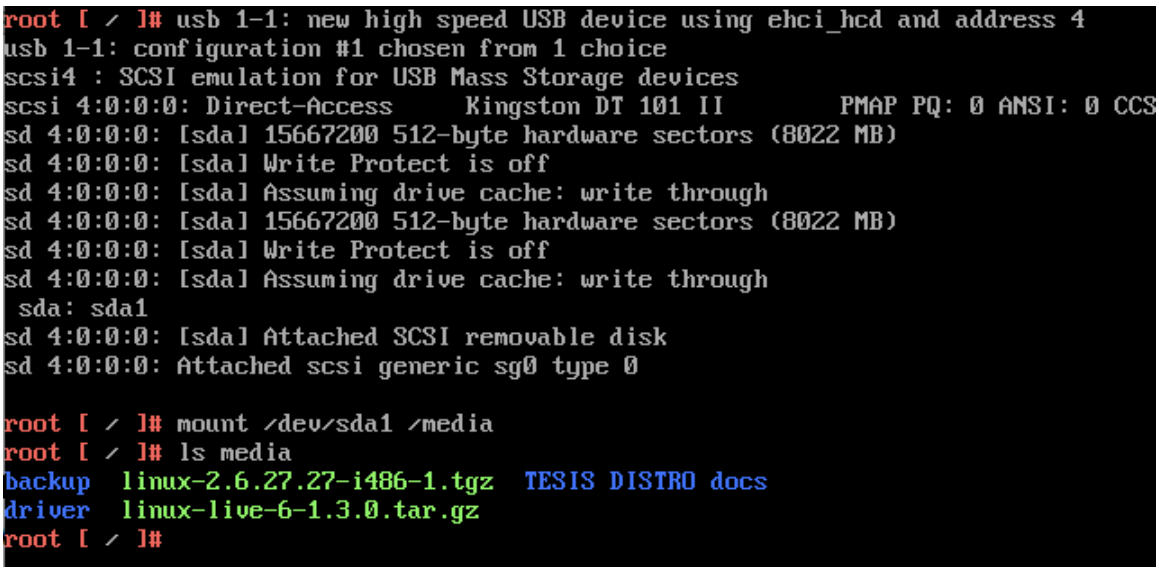
1.- La memoria flash puede ser detectada y se puede acceder a su contenido.

2.- Se establece correctamente la conexión de consola con el equipo de red.

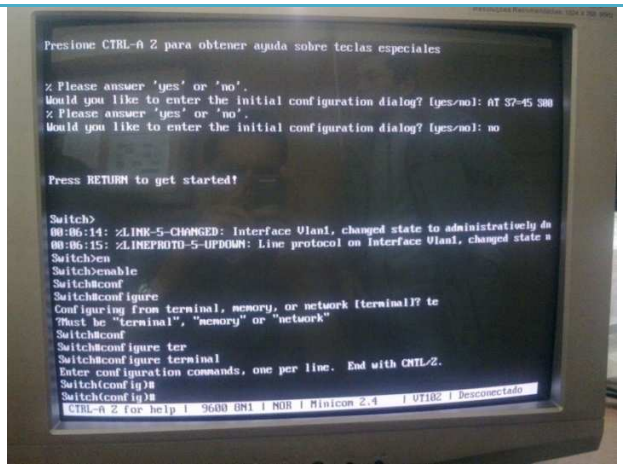
3.- Se tiene conectividad con la red.

Resultados Obtenidos:

1.- La memoria flash es detectada por el sistema. Se monta la memoria. El contenido de la flash es verificado. Estos resultados se muestran en la siguiente figura:



2.- Se establece exitosamente una sesión de consola con un equipo de red como se muestra en la siguiente captura:



3.- Se verifica la configuración de red y se tiene conectividad con la misma.

```
root [ / ]# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436
          txqueuelen:0

eth1      Link encap:Ethernet  HWaddr 00:0C:29:CD:F9:1B
          inet addr:172.16.164.131  Bcast:172.16.164.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500
          txqueuelen:1000

root [ / ]# ping www.google.com
PING www.l.google.com (74.125.229.17): 56 data bytes
64 bytes from 74.125.229.17: icmp_seq=0 ttl=128 time=122.832 ms
64 bytes from 74.125.229.17: icmp_seq=1 ttl=128 time=121.555 ms
64 bytes from 74.125.229.17: icmp_seq=2 ttl=128 time=121.563 ms
64 bytes from 74.125.229.17: icmp_seq=3 ttl=128 time=124.490 ms
64 bytes from 74.125.229.17: icmp_seq=4 ttl=128 time=123.454 ms
--- www.l.google.com ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 121.555/122.779/124.490/1.128 ms
root [ / ]#
```

Análisis de Resultados:

El sistema tiene compatibilidad con interfaces: usb, de red y de consola. Se muestra que la memoria flash es reconocida como un dispositivo scsi. Se verica la presencia de la interfaz loopback y la de Ethernet. Se prueba conectividad exitosamente con internet con el comando ping.

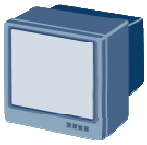
Tabla 3.5 Prueba, Compatibilidad con interfaces

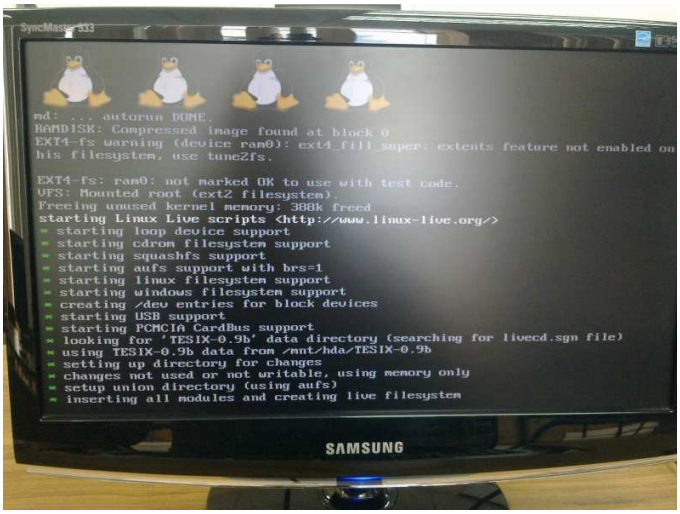
3.1.4 COMPATIBILIDAD CON PROCESADORES

Nombre de la prueba	Compatibilidad con procesadores
Objetivo de la Prueba	
Verificar que el sistema sea compatible con 2 marcas de procesadores de 32-bits bajo la arquitectura x86.	
Equipos para la prueba	
  ▪ 1 x computadora con un procesador Intel de 32-bits ▪ 1 x computadora con un procesador Intel de 64-bits ▪ 1 x computadora con un procesador AMD de 32-bits ▪ 1 x computadora con un procesador AMD de 64-bits ▪ 2 x copias del live CD del sistema	
Procedimiento para realizar la prueba:	
1.- Arrancar el sistema en la primera computadora. 2.- Realizar la misma prueba con todas las computadoras.	
Resultados Esperados:	
1.- El sistema debería arrancar sin ningún problema con las 2 marcas de procesador de 32 bits.	
Resultados Obtenidos:	
1.- El sistema arranca exitosamente con los dos tipos de procesador (32 y 64 bits) de las 2 marcas (Intel y AMD).	
Análisis de Resultados:	
El sistema arranca en 4 tipos de arquitectura. Se cumple con el alcance del proyecto. Se provee compatiilidad adicional con arquitecturas de procesador de 64 bits.	

Tabla 3.6 Prueba, Compatibilidad con procesadores

3.1.5 COMPATIBILIDAD CON MONITORES

Nombre de la prueba	Compatibilidad con monitores
Objetivo de la Prueba	
Verificar la compatibilidad del sistema con 2 tipos de monitores.	
Equipos para la prueba	
 ▪ 1 x computador con un monitor LCD ▪ 1 x computador con un monitor CRT ▪ 2 x copias del LiveCD del sistema	
Procedimiento para realizar la prueba:	
1.- Arrancar el sistema desde el LiveCD, verificar si se observa la imagen del mismo en el monitor. 2.- Realizar este proceso con cada tipo de monitor.	
Resultados Esperados:	
1.- En los 2 tipos de monitores se debería observar claramente la imagen que muestra el sistema.	
Resultados Obtenidos:	
1.- En los 2 monitores se puede observar la imagen que muestra el sistema. Esto se puede observar en las siguientes capturas:	



Análisis de Resultados:

Se tiene compatibilidad con los 2 tipos de monitores. Al tener compatibilidad con monitores CRT se puede reutilizar hardware que se encuentre obsoleto o que no se utiliza actualmente.

Tabla 3.7 Prueba, Compatibilidad con monitores

3.2 PRUEBAS PARA VERIFICAR EL FUNCIONAMIENTO DE LAS HERRAMIENTAS INSTALADAS

En esta sección se realizan pruebas para corroborar que las herramientas instaladas para cada materia funcionen correctamente. Para esto se incluirá una práctica con cada herramienta. Las pruebas serán agrupadas por materia.

3.2.1 GESTIÓN DE REDES

3.2.1.1 Net-snmp

Nombre de la prueba	Net-snmp
Objetivo de la Prueba	
Verificar el correcto funcionamiento de la herramienta *Net-snmp* incluida en el sistema GNU/Linux.	
Equipos para la prueba	
▪ 1 x copia del LiveCD del sistema ▪ 1 x laptop conectada a una red LAN y con acceso a Internet	
Procedimiento para realizar la prueba:	
1.- Se inicia el computador desde el sistema del LiveCD. 2.- Se ejecuta los comandos detallados a continuaciòn para verificar el correcto paquete Net-snmp. *Snmpget –v 2c –c demopublic test.net-snmp.org sysuptime.0* *Snmpget –v 2c –c demopublic test.net-snmp.org sys* Nota: Se utiliza un servidor de pruebas (test.net-snmp.org) que se encuentra dentro de la comunidad pública declarada en snmp.	

Resultados Esperados:

Observar que los comandos del paquete Net-snmp funcionen correctamente presentando en pantalla los datos requeridos del NMS consultado. En este caso se utiliza al host: *test.net-snmp.org*.

Resultados Obtenidos:

Se realizan pruebas con los comandos de Net-snmp: snmpget y snmpwalk para corroborar el correcto funcionamiento. Se confirma que el paquete con sus comandos funcionan correctamente como se puede apreciar en la siguiente captura.

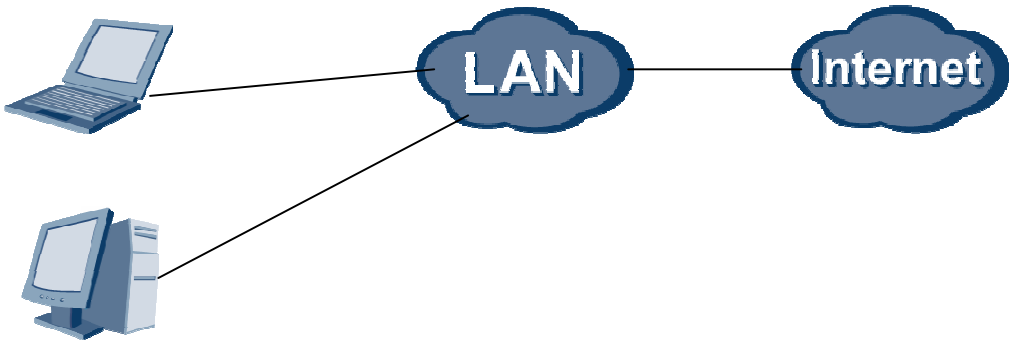
```
root [ ~ ]# snmpget -v 2c -c demopublic test.net-snmp.org sysUpTime.0
SNMPv2-MIB::sysUpTime.0 = Timeticks: (35235416) 4 days, 1:52:34.16
root [ ~ ]# snmpwalk -v 2c -c demopublic test.net-snmp.org system
process 'snmpwalk' is using obsolete setsockopt SO_BSDCOMPAT
SNMPv2-MIB::sysDescr.0 = STRING: Linux snmptest.netsec.tislabs.com 2.6.18-164.15
.1.el5 #1 SMP Wed Mar 17 11:30:06 EDT 2010 x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
SNMPv2-MIB::sysUpTime.0 = Timeticks: (35242535) 4 days, 1:53:45.35
SNMPv2-MIB::sysContact.0 = STRING: Net-SNMP Coders <net-snmp-coders@lists.source
forge.net>
SNMPv2-MIB::sysName.0 = STRING: snmptest.netsec.tislabs.com
SNMPv2-MIB::sysLocation.0 = STRING: Undisclosed
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (1) 0:00:00.01
SNMPv2-MIB::sysORID.1 = OID: SNMPv2-MIB::snmpMIB
SNMPv2-MIB::sysORID.2 = OID: SNMP-VIEW-BASED-ACM-MIB::vacmBasicGroup
SNMPv2-MIB::sysORID.3 = OID: SNMP-MPD-MIB::snmpMPDMIBObjects.3.1.1
SNMPv2-MIB::sysORID.4 = OID: SNMP-USER-BASED-SM-MIB::usmMIBCompliance
SNMPv2-MIB::sysORID.5 = OID: SNMP-FRAMEWORK-MIB::snmpFrameworkMIBCompliance
SNMPv2-MIB::sysORDescr.1 = STRING: The MIB module for SNMPv2 entities
SNMPv2-MIB::sysORDescr.2 = STRING: View-based Access Control Model for SNMP.
SNMPv2-MIB::sysORDescr.3 = STRING: The MIB for Message Processing and Dispatchin
g.
SNMPv2-MIB::sysORDescr.4 = STRING: The MIB for Message Processing and Dispatchin
g.
SNMPv2-MIB::sysORDescr.5 = STRING: The SNMP Management Architecture MIB.
SNMPv2-MIB::sysORUpTime.1 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.2 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.3 = Timeticks: (1) 0:00:00.01
SNMPv2-MIB::sysORUpTime.4 = Timeticks: (1) 0:00:00.01
SNMPv2-MIB::sysORUpTime.5 = Timeticks: (1) 0:00:00.01
SNMPv2-MIB::sysORUpTime.5 = No more variables left in this MIB View (It is past
the end of the MIB tree)
root [ ~ ]#
```

Análisis de Resultados:

De los resultados obtenidos de la prueba realizada se confirma el correcto funcionamiento del Net-snmp. El proyecto cumple con este requerimiento pedido.

Tabla 3.8 Prueba, Net-snmp

3.2.1.2 Traceroute

Nombre de la prueba	Traceroute
Objetivo de la Prueba	
Verificar el correcto funcionamiento del programa traceroute.	
Equipos para la prueba	
 ▪ 1 x copia del live CD del sistema ▪ 1 x laptop o pc con interfaz de red	
Procedimiento para realizar la prueba:	
1.- Verificar que se tenga una dirección IP asignada. 2.- Verificar que se tenga conectividad con internet. 3.- En la consola del sistema ejecutar el comando *traceroute www.google.com*	
Resultados Esperados:	
1.- Se debe poder observar las direcciones IP de todos los saltos que se dan antes de llegar al destino.	
Resultados Obtenidos:	
1.- Se observan todos los saltos como se muestra en la siguiente imagen:	

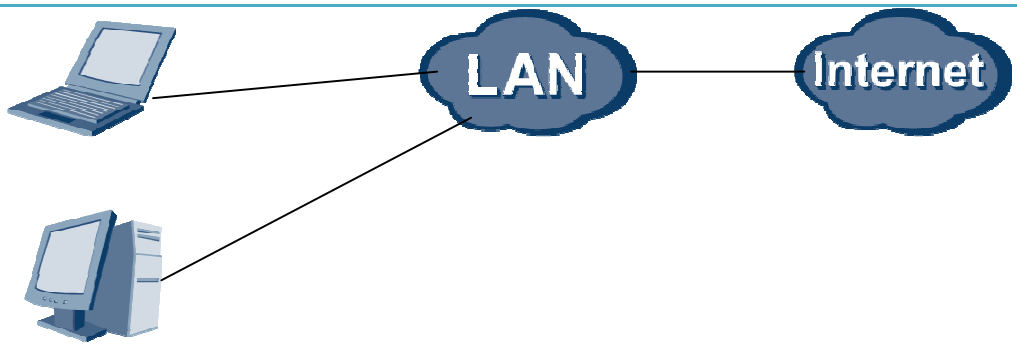
```
root [ ~ ]# traceroute www.google.com.ec
traceroute: Warning: www.google.com.ec has multiple addresses; using 74.125.229.
20
traceroute to www.l.google.com (74.125.229.20), 30 hops max, 40 byte packets
 1  FRIDA.mshome.net (192.168.0.1)  0.355 ms  0.288 ms  0.273 ms
 2  192.168.1.1 (192.168.1.1)  3.428 ms  4.460 ms  2.650 ms
 3  * * *
 4  190.152.127.238 (190.152.127.238)  22.643 ms 190.152.127.242 (190.152.127.24
2)  198.957 ms  121.525 ms
 5  190.152.127.241 (190.152.127.241)  66.171 ms  68.969 ms  46.673 ms
 6  186.46.4.25 (186.46.4.25)  58.969 ms  63.269 ms  72.235 ms
 7  186.46.4.2 (186.46.4.2)  71.158 ms  60.340 ms  65.108 ms
 8  186.42.168.1 (186.42.168.1)  27.819 ms  20.964 ms  46.365 ms
 9  190.152.254.138 (190.152.254.138)  31.737 ms *  46.907 ms
10  190.152.254.133 (190.152.254.133)  59.516 ms  88.336 ms  72.353 ms
11  pos5-3.miami2.mia.seabone.net (195.22.199.153)  140.920 ms  172.128 ms  138.
261 ms
12  te8-1.miami7.mia.seabone.net (195.22.199.99)  415.860 ms  438.527 ms  399.79
5 ms
13  google.miami7.mia.seabone.net (89.221.41.74)  416.099 ms google.miami7.mia.s
eabone.net (89.221.41.18)  401.524 ms  439.707 ms
14  209.85.253.116 (209.85.253.116)  226.274 ms  236.916 ms  247.851 ms
15  209.85.254.252 (209.85.254.252)  248.531 ms  254.340 ms  323.918 ms
16  64.233.175.34 (64.233.175.34)  267.320 ms  242.743 ms  250.039 ms
17  74.125.229.20 (74.125.229.20)  253.306 ms  262.183 ms  251.879 ms
root [ ~ ]#
```

Análisis de Resultados:

El programa funciona correctamente. Se pueden apreciar las direcciones IP de los saltos y tiempos que toma a un paquete llegar al destino.

Tabla 3.9 Prueba, Traceroute

3.2.1.3 Wireshark

Nombre de la prueba	Wireshark
Objetivo de la Prueba	
Verificar que el paquete que sirve para capturar tráfico funcione correctamente.	
Equipos para la prueba	
 ▪ 1 x copia del live CD del sistema ▪ 1 x laptop o pc con interfaz de red	
Procedimiento para realizar la prueba:	
1.- Verificar que se tenga una dirección IP asignada. 2.- Verificar que se tenga conectividad con internet. 3.- En la consola del sistema ejecutar el comando *tshark.* 4.- Realizar la misma prueba con el comando *tcpdump.*	
Resultados Esperados:	
1.- Con los comandos *tshark* y *tcpdump* se puede analizar el tráfico entrante y saliente en la interfaz de red.	
Resultados Obtenidos:	
Con los 2 programas se pueden capturar paquetes en la interfaz de red.como se muestra en las respectivas capturas.	

- Captura con tshark.

```
root [ ~ ]# tshark
device eth1 entered promiscuous mode
Capturing on eth1
1290358767.868951  192.168.0.9 -> 17.151.16.23 NTP NTP client
1290358768.176403 17.151.16.23 -> 192.168.0.9  NTP NTP server
1290358792.065045 192.168.0.218 -> 255.255.255.255 DHCP DHCP Request  - Transact
ion ID 0xa4af9217
1290358792.067140 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.218?
Tell 192.168.0.1
1290358792.067248 Vmware_cd:f9:1b -> QuantaCo_d3:8a:78 ARP 192.168.0.218 is at 0
0:0c:29:cd:f9:1b
1290358792.068641  192.168.0.1 -> 255.255.255.255 DHCP DHCP ACK      - Transacti
on ID 0xa4af9217
1290358792.389159  192.168.0.9 -> 192.168.0.255 NBNS Name query NB WORKGROUP<1d>
1290358792.389170  192.168.0.9 -> 192.168.0.1  DNS Standard query A anna.mshome.
net
1290358792.389172  192.168.0.1 -> 192.168.0.9  DNS Standard query response A 192
.168.0.9
1290358792.389179  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358792.389181  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358792.389186  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358796.649452 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358802.025054 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358803.672816  192.168.0.1 -> 192.168.0.255 BROWSER Domain/Workgroup Announc
ement INICIOMS, NT Workstation, Domain Enum
1290358807.525280 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358768.176403 17.151.16.23 -> 192.168.0.9  NTP NTP server
1290358792.065045 192.168.0.218 -> 255.255.255.255 DHCP DHCP Request  - Transact
ion ID 0xa4af9217
1290358792.067140 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.218?
Tell 192.168.0.1
1290358792.067248 Vmware_cd:f9:1b -> QuantaCo_d3:8a:78 ARP 192.168.0.218 is at 0
0:0c:29:cd:f9:1b
1290358792.068641  192.168.0.1 -> 255.255.255.255 DHCP DHCP ACK      - Transacti
on ID 0xa4af9217
1290358792.389159  192.168.0.9 -> 192.168.0.255 NBNS Name query NB WORKGROUP<1d>
1290358792.389170  192.168.0.9 -> 192.168.0.1  DNS Standard query A anna.mshome.
net
1290358792.389172  192.168.0.1 -> 192.168.0.9  DNS Standard query response A 192
.168.0.9
1290358792.389179  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358792.389181  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358792.389186  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358796.649452 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358802.025054 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358803.672816  192.168.0.1 -> 192.168.0.255 BROWSER Domain/Workgroup Announc
ement INICIOMS, NT Workstation, Domain Enum
1290358807.525280 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358813.025165 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
device eth1 left promiscuous mode
17 packets captured
root [ ~ ]#
```

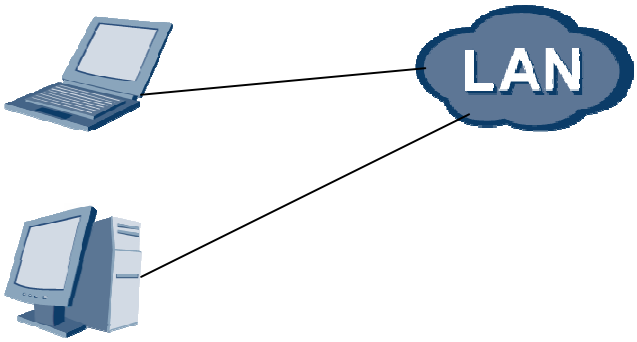
- Captura con tcpdump

```
root [ ~ ]# tcpdump
device eth1 entered promiscuous mode
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
12:01:12.902641 ARP, Request who-has tesis.mshome.net tell FRIDA.mshome.net, len
gth 46
12:01:12.909778 ARP, Reply tesis.mshome.net is-at 00:0c:29:cd:f9:1b (oui Unknown
), length 28
12:01:12.909890 IP FRIDA.mshome.net > tesis.mshome.net: ICMP echo request, id 10
24, seq 1280, length 40
12:01:12.909996 IP tesis.mshome.net > FRIDA.mshome.net: ICMP echo reply, id 1024
, seq 1280, length 40
12:01:12.904547 IP tesis.mshome.net.46330 > FRIDA.mshome.net.domain: 58306+ PTR?
 218.0.168.192.in-addr.arpa. (44)
12:01:12.908433 IP FRIDA.mshome.net.domain > tesis.mshome.net.46330: 58306- 1/0/
0 PTR tesis.mshome.net. (100)
12:01:12.908694 IP tesis.mshome.net.49928 > FRIDA.mshome.net.domain: 23321+ PTR?
 1.0.168.192.in-addr.arpa. (42)
12:01:12.909606 IP FRIDA.mshome.net.domain > tesis.mshome.net.49928: 23321- 1/0/
0 PTR FRIDA.mshome.net. (96)
12:01:13.901816 IP FRIDA.mshome.net > tesis.mshome.net: ICMP echo request, id 10
24, seq 1536, length 40
12:01:13.901837 IP tesis.mshome.net > FRIDA.mshome.net: ICMP echo reply, id 1024
, seq 1536, length 40
12:01:14.901711 IP FRIDA.mshome.net > tesis.mshome.net: ICMP echo request, id 10
24, seq 1792, length 40
12:01:14.901737 IP tesis.mshome.net > FRIDA.mshome.net: ICMP echo reply, id 1024
, seq 1792, length 40
1290358768.176403 17.151.16.23 -> 192.168.0.9  NTP NTP server
1290358792.065045 192.168.0.218 -> 255.255.255.255 DHCP DHCP Request  - Transact
ion ID 0xa4af9217
1290358792.067140 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.218?
Tell 192.168.0.1
1290358792.067248 Vmware_cd:f9:1b -> QuantaCo_d3:8a:78 ARP 192.168.0.218 is at 0
0:0c:29:cd:f9:1b
1290358792.068641  192.168.0.1 -> 255.255.255.255 DHCP DHCP ACK      - Transacti
on ID 0xa4af9217
1290358792.389159  192.168.0.9 -> 192.168.0.255 NBNS Name query NB WORKGROUP<1d>
1290358792.389170  192.168.0.9 -> 192.168.0.1  DNS Standard query A anna.mshome.
net
1290358792.389172  192.168.0.1 -> 192.168.0.9  DNS Standard query response A 192
.168.0.9
1290358792.389179  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358792.389181  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358792.389186  192.168.0.9 -> 192.168.0.255 NBNS Name query NB FRIDA<00>
1290358796.649452 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358802.025054 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358803.672816  192.168.0.1 -> 192.168.0.255 BROWSER Domain/Workgroup Announc
ement INICIOMS, NT Workstation, Domain Enum
1290358807.525280 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
1290358813.025165 QuantaCo_d3:8a:78 -> Broadcast    ARP Who has 192.168.0.129?
Tell 192.168.0.1
device eth1 left promiscuous mode
17 packets captured
root [ ~ ]#
```

Análisis de Resultados:
Tanto con tdpdump como con tshark se pueden capturar los paquetes que ingresan y salen de la interfaz Ethernet. El proyecto incluye el comando *tshark* en lugar de *wireshark*. *tshark* es el equivalente en modo consola de *wireshark* en modo gráfico.

Tabla 3.10 Prueba, Wireshark

3.2.1.4 Fping

Nombre de la prueba	fping
Objetivo de la Prueba	
Verificar el correcto funcionamiento del programa fping, comprobando los host se que encuntran activos en una red.	
Equipos para la prueba	
 ▪ 1 x copia del live CD del sistema ▪ 1 x laptop o pc con interfaz de red	
Procedimiento para realizar la prueba:	
1.- Conectarse a una red de area local (LAN) que tenga un servidor DHCP 2.- Ejecutar el siguente comando: *fping –g IP_inicio IP_fin > fping.txt* 3.- Verificar los hosts activos de la red mediante el comando: *cat fping.txt* | *grep alive*	

Resultados Esperados:

1.- El comando fping se debe ejecutar.

2.- Se debe crear un archivo con el ping realizado a todos los hosts que se encuentran dentro del rango de direcciones IP.

3.- Se pueden verificar los hosts activos en la red.

Resultados Obtenidos:

1.- El comando se ejecuta correctamente como se muestra en la siguiente captura:

```
root [ ~ ]# fping -g 192.168.0.1 192.168.0.250 > fping.txt
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.2
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.3
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.4
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.5
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.6
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.7
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.8
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.10
ICMP Host Unreachable from 192.168.0.218 for ICMP Echo sent to 192.168.0.11
```

2.- Se obtienen las direcciones IPs de los hosts activos como se muestra a continuación:

```
root [ ~ ]# cat fping.txt | grep alive
192.168.0.1 is alive
192.168.0.9 is alive
192.168.0.218 is alive
root [ ~ ]#
```

Análisis de Resultados:

El comando fping verifica si se encuentran activas las direcciones IP del rango especificado. Con la ayuda del comando grep se puede observar las IPs de los hosts que respondieron al comando ping.

Tabla 3.11 Prueba, Fping

3.2.1.5 Nmap

Nombre de la prueba	Nmap
Objetivo de la Prueba	
Verificar el correcto funcionamiento del comando *nmap* para exploración y auditoría de red.	
Equipos para la prueba	
 ▪ 1 x copia del live CD del sistema ▪ 1 x laptop o pc con interfaz de red	
Procedimiento para realizar la prueba:	
1.- Conectarse a una red de area local (LAN) que tenga un servidor DHCP 2.- Ejecutar el siguente comando: *nmap 192.168.1.1*	
Resultados Esperados:	
El comando *nmap* debe funcionar presentando información en pantalla del mapeo de red correspondiente a la dirección IP del argumento ingresado para el comando.	
Resultados Obtenidos:	
Se presenta todos los datos recabados por el comando *nmap* del host con dirección IP: 192.168.1.1	

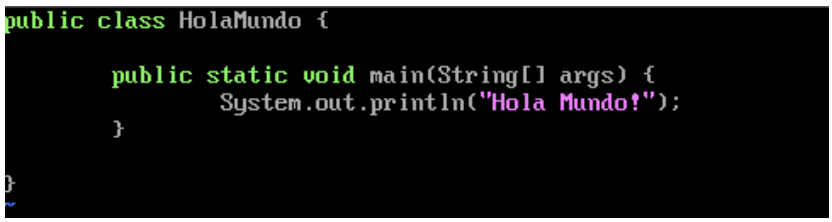
```
Starting Nmap 4.20 ( http://insecure.org ) at 2010-11-21 12:19 ECT
Initiating Parallel DNS resolution of 1 host. at 12:19
Completed Parallel DNS resolution of 1 host. at 12:19, 0.08s elapsed
Initiating SYN Stealth Scan at 12:19
Scanning 192.168.1.1 [1697 ports]
Discovered open port 80/tcp on 192.168.1.1
Completed SYN Stealth Scan at 12:19, 2.83s elapsed (1697 total ports)
Initiating Service scan at 12:19
Scanning 1 service on 192.168.1.1
Completed Service scan at 12:19, 7.32s elapsed (1 service on 1 host)
Initiating OS detection (try #1) against 192.168.1.1
Retrying OS detection (try #2) against 192.168.1.1
Insufficient responses for TCP sequencing (2), OS detection may be less accurate
Retrying OS detection (try #3) against 192.168.1.1
Retrying OS detection (try #4) against 192.168.1.1
Retrying OS detection (try #5) against 192.168.1.1
Host 192.168.1.1 appears to be up ... good.
Interesting ports on 192.168.1.1:
Not shown: 1696 closed ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Embedded Allegro RomPager webserver 4.07 UPnP/1.0 (ZyXEL Zy
WALL 2)
No OS matches for host (If you know what OS is running on it, see http://insecur
e.org/nmap/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=4.20%D=11/21%OT=80%CT=1%CU=44094%PV=Y%DS=1%G=Y%TM=4CE954B7%P=i686
OS:-pc-linux-gnu)SEQ(SP=2D%GCD=A000%ISR=AD%TI=I%II=I%SS=S%TS=U)ECN(R=N)SEQ(
OS:SP=1E%GCD=14000%ISR=9D%TI=I%TS=U)SEQ(SP=24%GCD=A000%ISR=AE%TI=I%II=I%SS=
OS:SP=1E%GCD=14000%ISR=9D%TI=I%TS=U)SEQ(SP=24%GCD=A000%ISR=AE%TI=I%II=I%SS=
OS:S%TS=U)OPS(O1=M300%O2=M300%O3=M300%O4=M300%O5=M300%O6=M300)OPS(O1=M300%O
OS:2=M300%O3=M300%O4=M300%O5=M300%O6=M300)T1(R=N)OPS(O1=M300%O2=M300%O3=M30
OS:0%O4=M300%O5=M300%O6=M300)WIN(W1=600%W2=600%W3=600%W4=600%W5=600%W6=600)
OS:WIN(W1=600%W2=600%W3=600%W4=600%W5=600%W6=600)T2(R=N)WIN(W1=600%W2=600%W
OS:3=600%W4=600%W5=600%W6=600)ECN(R=Y%DF=N%T=FE%W=600%O=M300%CC=N%Q=)ECN(R=
OS:Y%DF=N%T=FE%W=600%O=M300%CC=N%Q=)T3(R=N)ECN(R=Y%DF=N%T=FE%W=600%O=M300%C
OS:C=N%Q=)T1(R=Y%DF=N%T=FE%S=O%A=S+%F=AS%RD=0%Q=)T1(R=Y%DF=N%T=FE%S=O%A=S+%
OS:F=AS%RD=0%Q=)T4(R=N)T1(R=Y%DF=N%T=FE%S=O%A=O%F=AS%RD=0%Q=)T1(R=Y%DF=N%T=
OS:FE%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T2(R=N)T5(R=N)T2(R=N)T3(R=N)T3(R=N)T6(R=
OS:N)T3(R=N)T4(R=N)T4(R=Y%DF=N%T=FE%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=N)T4(R=
OS:Y%DF=N%T=FE%W=0%S=O%A=Z%F=R%O=%RD=0%Q=)T4(R=Y%DF=N%T=FE%W=0%S=A%A=Z%F=R%
OS:O=%RD=0%Q=)T5(R=N)T5(R=Y%DF=N%T=FE%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%D
OS:F=N%T=FE%TOS=0%IPL=38%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=6F0D%RUL=G%RUD=G)T5
OS:(R=Y%DF=N%T=FE%W=0%S=Z%A=O%F=AR%O=%RD=0%Q=)T5(R=Y%DF=N%T=FE%W=0%S=Z%A=S+
OS:%F=AR%O=%RD=0%Q=)T6(R=N)T6(R=Y%DF=N%T=FE%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)IE(R
OS:=N)T6(R=Y%DF=N%T=FE%W=0%S=O%A=Z%F=R%O=%RD=0%Q=)T6(R=Y%DF=N%T=FE%W=0%S=A%
OS:A=Z%F=R%O=%RD=0%Q=)T7(R=N)T7(R=N)T7(R=N)U1(R=N)U1(R=Y%DF=N%T=FE%TOS=0%IP
OS:L=38%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=6F0D%RUL=G%RUD=G)U1(R=N)U1(R=Y%DF=N%
OS:T=FE%TOS=0%IPL=38%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=6F0D%RUL=G%RUD=G)IE(R=N
OS:)IE(R=Y%DFI=N%T=FE%TOSI=S%CD=S%SI=S%DLI=S)IE(R=N)IE(R=Y%DFI=N%T=FE%TOSI=
OS:S%CD=S%SI=S%DLI=S)

Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at http:
//insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 26.165 seconds
               Raw packets sent: 1914 (94.884KB) | Rcvd: 1779 (82.596KB)
root [ ~ ]#
```

Análisis de Resultados:
Se confirma el correcto funcionamiento de la herramiena *nmap(network mapper)* al poder observar información de mapeo del host consultado, como por ejemplo: escaneo de puertos abiertos, número de saltos para llegar al destino,etc.

Tabla 3.12 Prueba, Nmap

3.2.2 SISTEMAS OPERATIVOS

3.2.2.1 Java

Nombre de la prueba	Java
Objetivo de la Prueba	
Verificar que java funcione correctamente mediante la compilación y ejecución de un programa	
Equipos para la prueba	
▪ 1 x copia del live CD del sistema ▪ 1 x laptop o pc	
Procedimiento para realizar la prueba:	
1.- Crear el archivo HolaMundo.java mediante el comando *vi HolaMundo.java* 2.- Insertar el siguiente texto en el archivo: 	

```
public class HolaMundo {

        public static void main(String[] args) {
                System.out.println("Hola Mundo!");
        }

}
```

3.- Compilar el programa con el comando *javac HolaMundo*

4.- Verificar que se cree el archivo HolaMundo*.class*

5.- Ejecutar el comando *java HolaMundo*

Resultados Esperados:

1.- El comando *javac HolaMundo* no debe arrojar errores de compilación.

2.- El fichero HolaMundo*.class* debe ser creado.

3.- Se debe desplegar en pantalla el mensaje: Hola Mundo!

Resultados Obtenidos:

1.- Se crea el fichero HolaMundo*.class.*

2.- Luego de ejecutar el comando: *java HolaMundo* se observa en pantalla el mensaje Hola Mundo!. Este proceso se muestra de manera seguida:

```
root [ ~ ]# javac HolaMundo.java
root [ ~ ]# ls
fping.txt  HolaMundo.class  HolaMundo.java  nmap.txt
root [ ~ ]# java HolaMundo
Hola Mundo!
root [ ~ ]#
```

Análisis de Resultados:

Se puede verificar en la captura anterior que el compilador de java se encuentra funcionando correctamente. Finalmente se observa el mensaje en pantalla Hola Mundo!!.

Tabla 3.12 Prueba, Java

3.2.3 HARDWARE DE CONECTIVIDAD

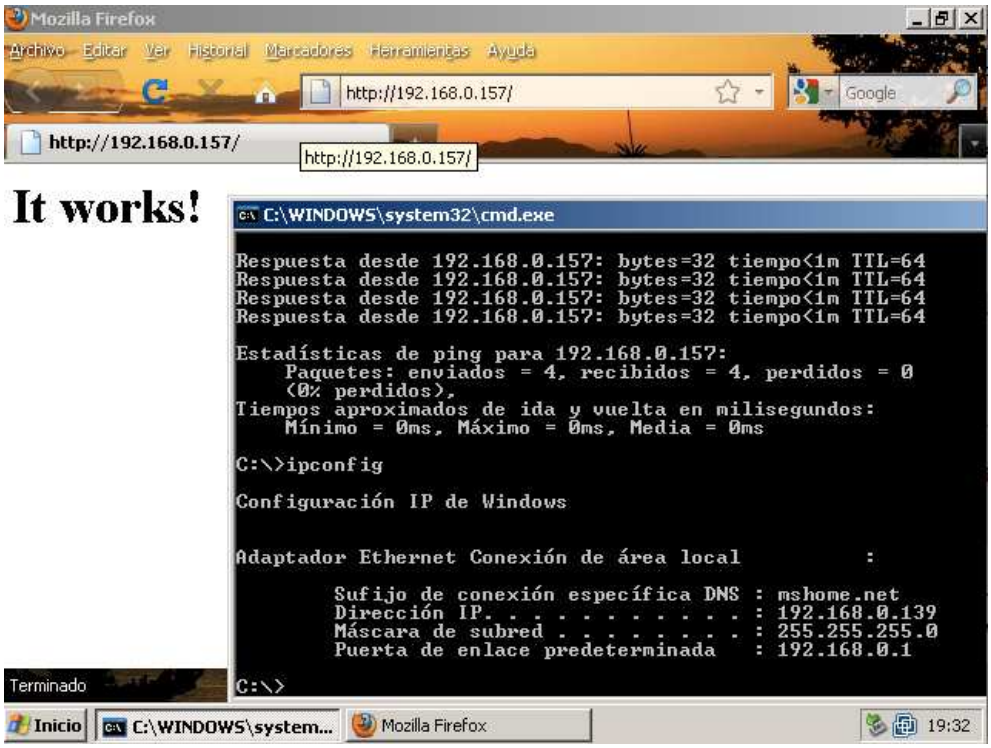
3.2.4.1 Apache

Nombre de la prueba	Apache (httpd)
Objetivo de la prueba	
Verificar el correcto funcionamiento del servicio de páginas web estáticas a través del servicio *httpd*.	
Equipos para la prueba	
▪ 1 x copia del live CD del sistema ▪ 2 x laptops o pc's (1 con el LiveCD y otra con Windows XP)	
Procedimiento para realizar la prueba:	
1.- Interconectar las 2 computadoras a través de una LAN (se realiza la conexión con la ayuda de un switch) 2.- Arrancar la laptop con la distribución LiveCD 2.- Configurar el archivo */etc/httpd.conf*	
Resultados Esperados:	
1.- El servidor Apache, de páginas web estáticas debe inicializarse sin presentar errores. 2.- Se debe tener conectividad (ping) entre los 2 equipos conectados en red. 3.- Se debe poder cargar en el navegador Web de un computador una página de prueba hospedada y servida por el otro computador.	
Resultados Obtenidos:	
El servidor funciona correctamente.	

A continuación se presentan las capturas con los resultados:

```
root [ ~ ]# apachectl -v
Server version: Apache/2.2.8 (Unix)
Server built:   Oct 23 2010 21:35:58
root [ ~ ]# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436
          txqueuelen:0

eth1      Link encap:Ethernet  HWaddr 00:0C:29:CD:F9:1B
          inet addr:192.168.0.157  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500
          txqueuelen:1000

root [ ~ ]# ping 192.168.0.139
PING 192.168.0.139 (192.168.0.139): 56 data bytes
64 bytes from 192.168.0.139: icmp_seq=0 ttl=128 time=13.541 ms
64 bytes from 192.168.0.139: icmp_seq=1 ttl=128 time=10.586 ms
64 bytes from 192.168.0.139: icmp_seq=2 ttl=128 time=5.671 ms
64 bytes from 192.168.0.139: icmp_seq=3 ttl=128 time=5.933 ms
--- 192.168.0.139 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 5.671/8.933/13.541/3.302 ms
root [ ~ ]#
```

Computador con el servidor Apache ejecutándose.

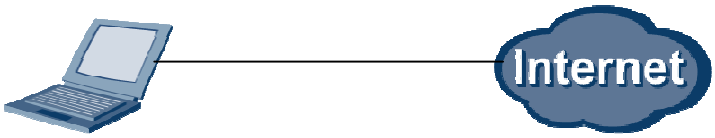


Computador con el navegador Web cargando la página de prueba

Análisis de Resultados:
Se observa en las capturas que los dos computadores se encuentran conectados a la red, con direcciones IP 192.168.0.157 y 192.168.0.139 respectivamente. Se verifica conectividad entre los dos computadores, requisito escencial para poder probar el servicio de Servidor de Páginas Web. Se verifica en el computador con S.O. Windows ® IP (192.168.0.139) y Mozilla Web Browser que se carga la página ospedada en el computador corriendo el S.O. Linux, lo cual corrobora el correcto funcionamiento del servicio Apache.

Tabla 3.13 Prueba, Apache

3.2.4.2 FTP

Nombre de la prueba	Servicio FTP
Objetivo de la prueba:	
Verificar que el sistema Linux cuente con el servicio de FTP Client para realizar conexiones hacia servidores FTP y poder descargar o cargar archivos en el servidor remoto.	
Equipos para la prueba	
 ▪ 1 x copia del live CD del sistema ▪ 1x laptops o pc conectada a internet	
Procedimiento para realizar la prueba:	
1.- Iniciar el computador desde el LiveCD de Linux. 2.- Ejecutar el comando *ftp <dir.ip.del.servidor>* (o nombre DNS); para este caso se ejemplifica con: ftp ftp.mozilla.org.	

3.- Autenticarse como usuario *invitado*.

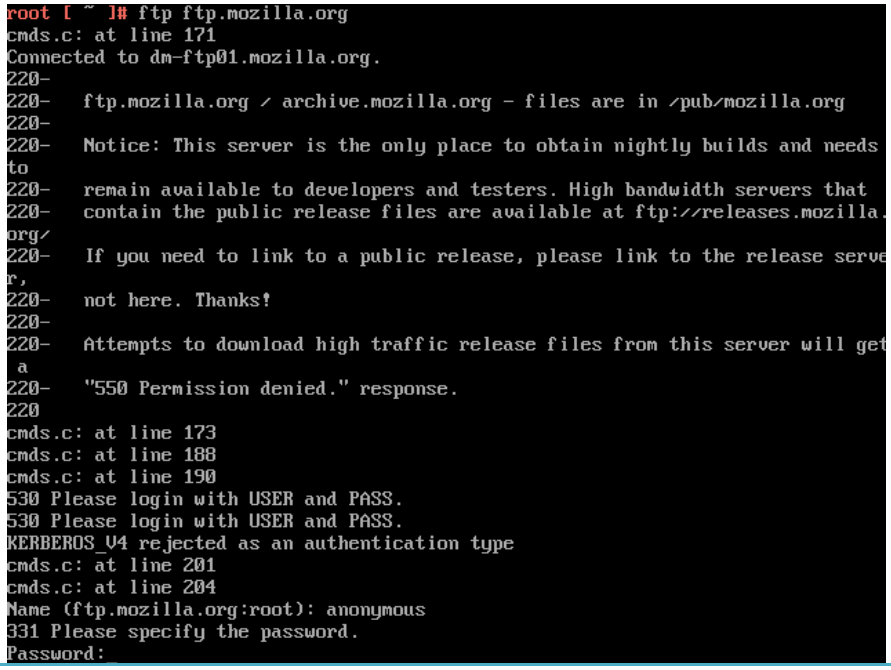
4.- Desconectarse del servidor.

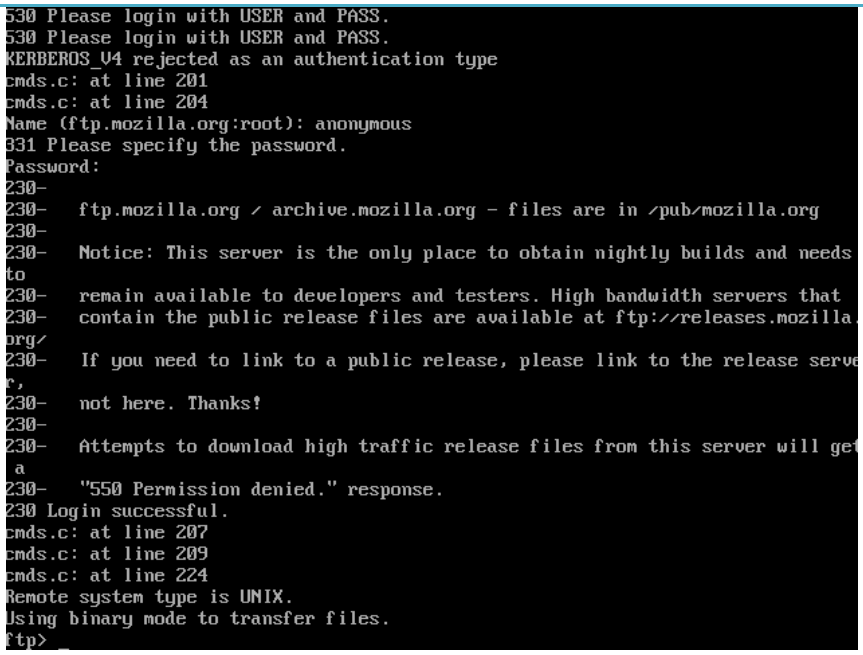
Resultados Esperados:

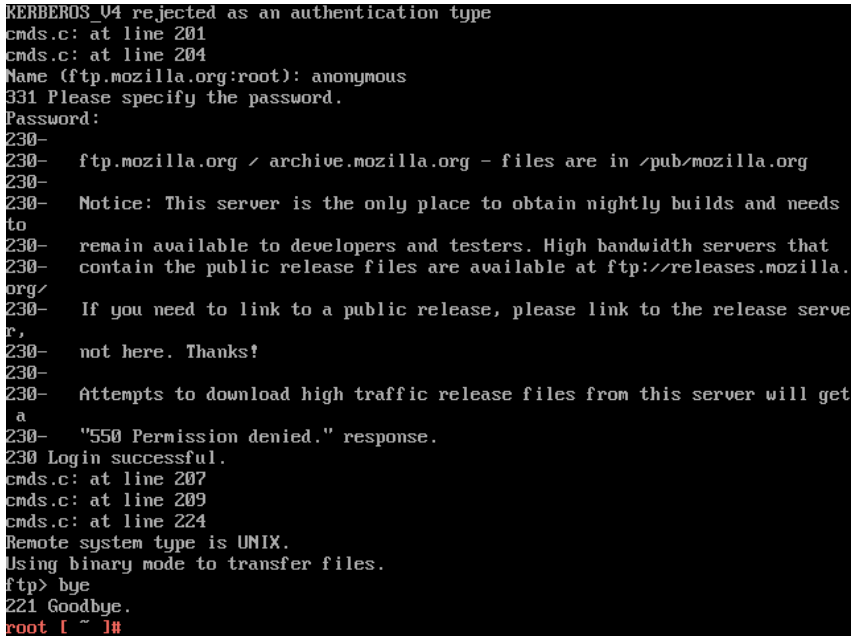
Se espera que el comando ftp funcione y permita conectarse a un servidor ftp remoto.

Resultados Obtenidos:

Con el comando mostrado anteriormente se establece conexión con el servidor FTP. Como se muestra en las siguientes capturas:





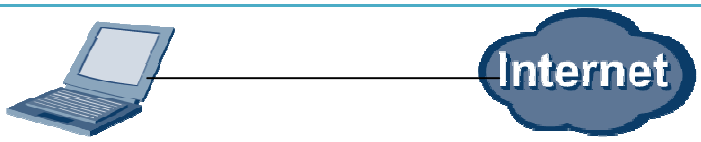


Análisis de Resultados:

Se observa en las capturas el correcto funcionamiento del comando *ftp* cumpliendo con el objetivo de la prueba. Se observa que se establece conexión con el servidor ftp.mozilla.org.y se autentica como usuario *invitado*.

Tabla 3.14 Prueba, FTP

3.2.4.3 DNS

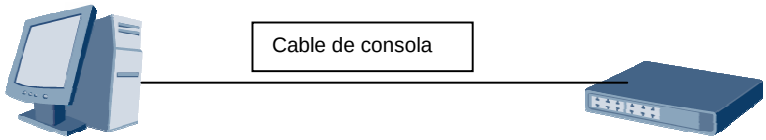
Nombre de la prueba	**DNS**
Objetivo de la prueba	
Verificar que el sistema sea capaz de realizar resolución de nombres	
Equipos para la prueba	
 ▪ 1 x computador portátil con conexión a Internet. ▪ 1 x copia del live CD del sistema.	
Procedimiento para realizar la prueba:	
1.- Iniciar el computador desde el LiveCD de Linux. 2.- Ejecutar el comando: *host* www.epn.edu.ec 3.- Ejecutar el comando: *host 192.188.57.254*	
Resultados Esperados:	
Verificar que el comando *host* realice la resolución de nombre tanto como de la dirección IP dada.	
Resultados Obtenidos:	

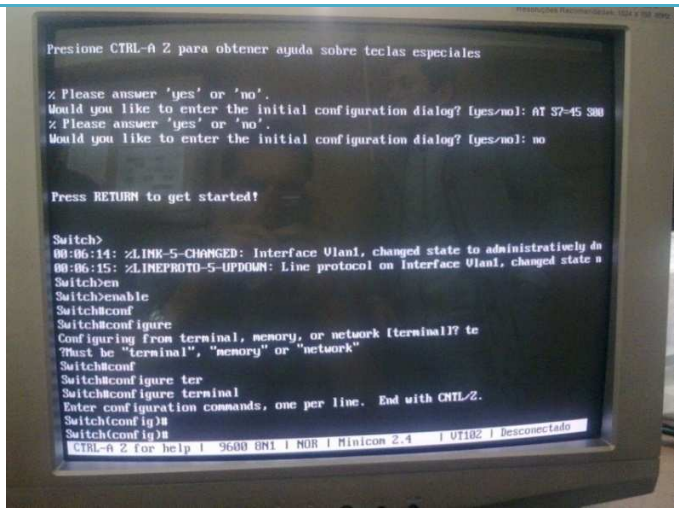
El comando realiza correctamente la resolución de nombres como se puede ver en la siguiente captura:
 `root [ ~ ]# host www.epn.edu.ec`
 `www.epn.edu.ec has address 192.188.57.254`
 `root [ ~ ]# host 192.188.57.254`
 `254.57.188.192.in-addr.arpa domain name pointer www.epn.edu.ec.`
 `root [ ~ ]#`	

Análisis de Resultados:	
El comando *host* funciona resolviendo nombres de dominio a direcciones IP y viceversa. Se corrobora el correcto funcionamiento.	

Tabla 3.15 Prueba, DNS

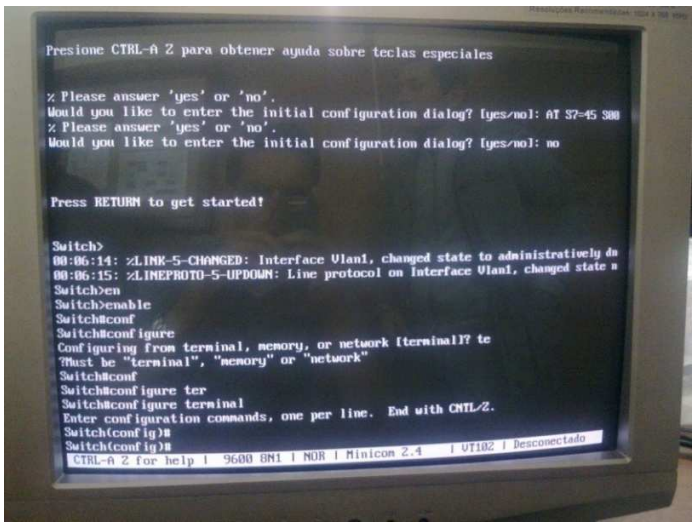
3.2.4.4 Minicom

Nombre de la prueba	Minicom
Objetivo de la Prueba	
Acceder a un equipo de red vía consola con el programa minicom.	
Equipos para la prueba	
 ▪ 1 x copia del live CD del sistema ▪ 1 x pc del laboratorio de Redes ▪ 1 x cable de consola	
Procedimiento para realizar la prueba:	
1.- Arrancar el LiveCD en una computadora del laboratorio. 2.- Conectar el cable de consola desde el computador a un switch del laboratorio. 3.- Configurar el puerto COM del computador con el programa *minicom* con la configuración: *8N1N*.	
Resultados Esperados:	
1.- Se establece comunicación via consola con el switch.	



Resultados Obtenidos:

Se accede exitosamente a la configuración del switch como se muestra en la siguiente imagen:



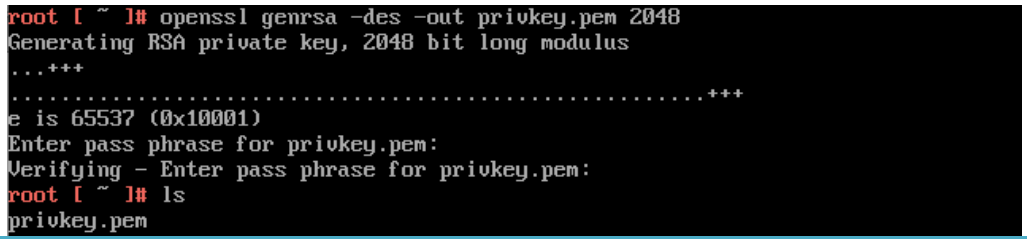
Análisis de Resultados:

El funcionamiento del programa minicom cumple con su objetivo. Sirve para configurar via puerto serial un equipo de red.

Tabla 3.16 Prueba, Minicom

3.2.4 SEGURIDAD DE REDES

3.2.5.1 OpenSSL

Nombre de la prueba	OpenSSL
Objetivo de la prueba	
Verifica el funcionamiento del sistema de cifrado OpenSSL	
Equipos para la prueba	
 ▪ 1 x laptop ▪ 1 x copia del sistema	
Procedimiento para realizar la prueba:	
1.- Arrancar el LiveCD en un computador. 2.- Ejecutar los comandos presentados en las siguientes capturas	
Resultados Esperados:	
Usar las herramientas de OpenSSL para crear certificados digitales y llaves privadas	
Resultados Obtenidos:	
Se puede verificar la creación de un certificado digital y su llave privada como se observa en las siguientes capturas:  root [~]# openssl genrsa -des -out privkey.pem 2048 Generating RSA private key, 2048 bit long modulus ...+++ .....+++ e is 65537 (0x10001) Enter pass phrase for privkey.pem: Verifying - Enter pass phrase for privkey.pem: root [~]# ls privkey.pem	

```
root [ ~ ]# openssl req -new -key privkey.pem -out cert.csr
Enter pass phrase for privkey.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:EC
State or Province Name (full name) [Some-State]:PICHINCHA
Locality Name (eg, city) []:QUITO
Organization Name (eg, company) [Internet Widgits Pty Ltd]:EPN
Organizational Unit Name (eg, section) []:ELECTRONICA
Common Name (eg, YOUR name) []:CRISTIAN GILER
Email Address []:cgiler@epn.edu.ec

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:prueba
An optional company name []:EPN
root [ ~ ]# ls
cert.csr  privkey.pem
root [ ~ ]# cat cert.csr
-----BEGIN CERTIFICATE REQUEST-----
MIIDATCCAekCAQAwgZAxCzAJBgNVBAYTAkVDMRIwEAYDVQQIEwlQSUNISU5DSEEx
DjAMBgNVBAcTBVFVSVRPMQwwCgYDVQQKEwNFUE4xFDASBgNVBAsTC0VMRUNUUk9O
SUNBMRcwFQYDVQQDEw5DUk1TVE1BTiBHSUxFUjEgMB4GCSqGSIb3DQEJARYRY2dp
bGVyQGVwbi5lZHUuZWMwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDG
aDSh6JGMbJgFNYT5Iv276TssrtYTi9rs8+HMIgAckRsoAyXKxEAO9acgDWVM++jq
Dz400pex8dhP+SCwxTIB/ZFbpNpx2liBB4f2P0NBBazyRM663LufheeaIJKxBVWE
jovLuTd3bbb7/GSNN9i6lDjTsDc3n06jmt2cXMpgcPyrO3PxjctD2lJFB0whPWI1
RnC0GTP9hW7Gn/FTx6mAK3ORerEg/2qPtJAj2T74cYK5xDqnE+QsbfCnLcP9DqTA
H2rOyrFZugLP7DNwUbjwz4nJ9NBGEPciyMaUvb44zzbjo1tR0ei2yausEHASYMCb
Jtt1dkxyX7PCEOAici+JAgMBAAGgKzASBgkqhkiG9w0BCQIxBRMDRVBOMBUGCSqG
SIb3DQEJBzEIEwZwcnVlYmEwDQYJKoZIhvcNAQEFBQADggEBALabLHPSjG9AN2C/
Fhvi2J3NouAJjsUB+YTHwfdhkm8ChHCSwFilMgVq2jQVGKpyEzUOCVu2c4NOQkod
u2Jyj/LEYBa/Lu+/ppnoOOngJ98tnCyArr3MNIEow2M7maKfwkPvAp25YobR5vAG
CCAwIg/626dqyI8ON/dfYb9wZQQPwXOE0zTtySe7GzpdjuaNhfmEKgIbrEZLgEp9
6guS383NwEHv7w06uvrJZtc1cxf4jowvdpfFcXKIpVdZXco2OoFEceHDyjH/ssbJ
9d8kMdG+sI42RHVePkpzd9MKNB068sr6cn1TQP9Qe10UM8Rq+6wVTksghSHNzHZ+
hHhRbRU=
-----END CERTIFICATE REQUEST-----
root [ ~ ]#
```

Análisis de Resultados:

Se verifica el correcto funcionamiento de la herramienta OpenSSL por medio de la creación de una llave privada y un certificado digital.

Tabla 3.17 Prueba, OpenSSL

3.2.5.2 Kerberos

Nombre de la prueba	Kerberos
Objetivo de la prueba	
Verificar el correcto funcionamiento del paquete de MIT-Kerveros	
Resultados Obtenidos:	
Se realizó la instalación del paquete correspondiente a Kerberos, pero la realización de pruebas no fue posible debido a que el LiveCD de Linux luego de configurado debe ser reiniciado, para poder kerberizar el o los servicios que el sistema GNU/Linux ofrezca. Al momento de reiniciar el sistema se pierden las configuraciones realizadas antes del reinicio.	

Tabla 3.18 Prueba, Kerberos

3.2.5.3 IPsec

Nombre de la prueba	IPSec
Objetivo de la Prueba	
Verificar el correcto funcionamiento del paquete de software correspondiete a IPSec	
Resultados Obtenidos:	
No se puede verificar el correcto funcionamiento de este paquete debido a que IPSec no se pudo implementar dentro del sistema Linux por problemas de compatibilidad con el Kernel 2.6.27.27 y los requerimientos mínimos del paquete IPSec.	

Tabla 3.19 Prueba, IPSec

3.2.5 APLICACIONES DISTRIBUIDAS

3.2.6.1 OpenLDAP

Nombre de la prueba	**OpenLDAP**
Objetivo de la prueba	
Verificar el funcionamiento del servidor de directorio activo OpenLDAP	
Equipos para la prueba	
 2 x computadores 1 x red LAN 1 x copia del LiveCD del sistema GNU/Linux	
Procedimiento para realizar la prueba:	
1.- Interconectar en una LAN los dos computadores. 2.- Iniciar uno de los computadores con el LiveCD 3.- Realizar la configuración de los parámetros para el servidor de directorio activo e iniciar el servicio. 4.- Agregar al domino al segundo computador.	
Resultados Esperados:	
Desde el segundo computador realizar pruebas de búsqueda dentro del directorio activo.	
Resultados Obtenidos:	
Se realizó la instalación del paquete correspondiente a OpenLDAP, pero la realización de pruebas no fue posible debido a que el LiveCD de Linux luego de	

configurado debe ser reiniciado. Al momento de reiniciar el sistema se pierden las configuraciones realizadas antes del reinicio. Para solventar dicho problema se debe crear una versión arrancable en FlashUSB donde se pueda realizar cambios que no se pierdan con el reinicio del sistema.

Tabla 3.20 Prueba, OpenLDAP

3.2.6.2 Heartbeat

Nombre de la prueba	Heartbeat
Objetivo de la prueba	
Verificar el correcto funcionamiento del paquete heartbeat para servicio de alta disponibilidad de servidores	
Equipos para la prueba	
Procedimiento para realizar la prueba:	
1.- Inicicar los 2 computadores con los LiveCD de GNU/Linux 2.- Configurar el servicio HA (alta disponibilidad) de Heartbeat en los dos computadores. Siendo un computador el servidor principal y el otro computador como servidor secundario. 3.- Iniciar servicios duplicados en ambos computadores; por ejemplo servicio de páginas web. 4.- Interconectar los dos compuadores a través de la red LAN y a través de cable serial NULL MODEM.	

5.- Realizar el corte simulado, o desconexión del cable serial.
Resultados Esperados:
Verificar el funcionamiento del servidor secundario como servidor activo para el servicio de páginas web.
Resultados Obtenidos:
Se realizan pruebas con dos computadores de los laboratorios de Redes, pudiéndose corroborar el funcionamiento de Heartbeat
Análisis de Resultados:
Luego de realizar las pruebas se verifica que se cumple con el requerimiento de funcionalidad del servicio de Alta Disponibilidad implementado en la distribución GNU/Linux.

Tabla 3.21 Prueba, Heartbeat

CAPÍTULO 4

CONCLUSIONES Y RECOMENDACIONES

4.1CONCLUSIONES

- El presente proyecto se ha visto beneficiado de las ventajas de las licencia GPL de GNU. Esta licencia permite tener las cuatro libertades del software libre: Primera, se pudo obtener fácilmente y sin ningún costo el código fuente de todos los paquetes que conforman el sistema. Segunda, se ejecutó el sistema final con propósito educativo. Tercera, en los casos que ameritó se revisaron archivos de configuración, así como los códigos fuentes de los programas de tal manera que se pueda realizar cambios enfocados en requerimientos propios. Cuarta, se distribuirá el producto final sin reservas.

- El gestor de paquetes no se implementó en el sistema. Aunque en el análisis de requerimientos se pidió la implementación de RPM como gestor de paquetes. Esto no se cumplió debido a que la distribución es enteramente basada en código fuente, y existen paquetes, como es el caso de RPM, que no distribuyen su código fuente, sino sólo sus binarios empaquetados en un archivo. Por tal motivo RPM no puede ser instalado desde la compilación del código fuente.

- La utilidad de implantación de módulos del kernel en este proyecto recae en los siguientes aspectos: Ayudan a hacer más liviano el código del núcleo lo que hace más rápido su arranque. Los módulos se cargan en caliente y solamente en caso de ser requeridos, sin necesidad de recompilar y reinstalar el kernel como por ejemplo al reconocer un dispositivo y cargar su controlador. Además que para crear el LiveCD, *Aufs* y *Squahfs* únicamente pueden ser implementados como

módulos, debido a limitaciones de compatibilidad con el kernel. Estos módulos son los primeros en cargarse al momento del arranque del sistema. En ese instante, el módulo *Squashfs* ayuda a descomprimir los directorios del sistema, y el módulo *Aufs* provee la capacidad de escritura sobre la memoria RAM.

- En el arranque del sistema se pueden apreciar advertencias por la no presencia de interfaces de red. Esto se debe a que el LiveCD de GNU/Linux se configuró para el uso de hasta 3 interfaces de red que en algún momento pueden ser usadas a nivel de laboratorio (incluida la interfaz inalámbrica). Al estar conectada la computadora a una red con sevidor DHCP la interfaz subirá y adquirirá la configuración IP. De no ser así se deberá configurar manualmente la dirección IP en el directorio */etc/sysconfig/network/network-devices/ifconfig.eth#* (donde # corresponde al número de interfaz de red).

- En la actualidad existe a nivel empresarial la flexibilidad para que el usuario adquiera hardware enfocado a las necesidades del mismo. Pero en cuanto a materia de sistemas operativos se carece de dicha flexibilidad, además que los costos de licenciamiento de software son elevados. Por ejemplo, los sistemas operativos de Microsoft® tienen costos por licencias y además no proveen todas las herramientas y servicios que una determinada empresa o institución requiere, ajustada a las necesidades puntuales del usuario. Las peticiones que la empresa o institución realicen para modificación del sistema operativo no son atendidas por el proveedor sino que el proveedor realiza cambios y actualizaciones de versiones según la demanda global de mercado o conveniencias de la empresa desarrolladora. El sistema GNU/Linux al caracterizarse por disponer de código abierto y de libre distribución brinda a las empresas e instituciones bondades como costo cero en licenciamiento y uso, dispone de una comunidad mundial de soporte y desarrollo, y lo más importante, que brinda la flexibilidad para realizar cambios ah hoc enfocados a los requerimientos del usuario.

4.2RECOMENDACIONES

- Como alcance a este proyecto se plantea la implantación de la interfaz gráfica para poder instalar otros programas necesarios como por ejemplo Network Simulator 2 y aprovechar todas las facilidades que brinda esta interfaz a otros programas como por ejemplo Wireshark.

- En caso de implantar nuevamente todo el sistema se recomienda especial cuidado con la instalación de los módulos del núcleo *Squashfs* y *Aufs*. Si Squashfs se encuentra mal instalado, el sistema tardará en arrancar debido a demoras en la descompresión de archivos *tgz*. También hay que tomar en cuenta que *Aufs* es el módulo que permite que se pueda escribir en la memoria virtual lo cual no se puede hacer si está mal instalado generando errores “read-only” en todos los archivos y estructura de directorios del sistema operativo. En medida de lo posible se recomienda el uso de un kernel precompildado con estas dos herramientas con la misma versión del kernel en uso.

- El producto final del presente proyecto es una única versión del sistema GNU/Linux en la que, bajo el cumplimiento de los objetivos del proyecto, constan los paquetes pedidos por los usuarios que harán uso del sistema, sin embargo se recomienda crear varias versiones del sistema operativo para que cada versión se enfoquen aún más en cada una de las materias que requieren el uso del sistema GNU/Linux.

- Por medio del análisis de la situación actual realizado en el presente proyecto se evidencia que los laboratorios de computación de la carrera de Redes no realizan una gestión formal en cuanto a TI. Se recomienda que la Politécnica fomente en sus graduandos el desarrollo de más proyectos de titulación que

se enfoquen específicamente en el tema de gestión de TI, no sólo de los laboratorios de la carrera sino de toda la estructura de TI de la institución. Con esto se lograría causar una avalancha de nuevas ideas y proyectos viables para ser implantados en nuestra Universidad y tener un mayor control de la infraestructura de TI

CAPÍTULO 5

BIBLIOGRAFÍA Y ANEXOS

5.1 BIBLIOGRAFÍA

Libros

[1] SILBERSCHATZ, Abraham; GALVIN, Peter; Sistemas Operativos; Quinta Edición; Pearson Educación; 1999.
[2] STALLINGS, William; Sistemas Operativos; Segunda Edición; Prentice Hall; 1997.
[3] CARRETERO, Jesús; GARCÍA, Felix; PÉREZ, Fernando; DE MIGUEL, Pedro. Sistemas Operativos Una Visión Aplicada; McGraw Hill; 2001.
[4] BEEKMANS, Gerard; Linux from Scratch; Versión 6.3; Agosto 2007.
[5] BEEKMANS, Gerard; BLFS EQUIPO DE DESARROLLO; Beyond Linux from Scratch; Versión 6.3; Agosto 2008.
[6] KROAH-HARTMAN, Greg; Linux Kernel in a Nutshell; 1st. Edition; O'Reilly; USA, Diciembre 2006.
[7] GARCÍA, Luis Martin; ¿Cómo compilar el Kernel Linux paso a paso?; Versión 0.3; España, Marzo 2006.
[8] MOTAB, Islam; LiveCD Crash Course; Aka Phaeronix; Noviembre 2006

Internet

[9] ITIL. http://itil.osiatis.es
[10] TAMAÑO DE MUESTRA. http://www.feedbacknetworks.com/cas/experiencia/sol-preguntar-calcular.htm
[11] HEARTBEAT. http://www.linux-ha.org/

[12] SLAX LIVECD. http://www.linux-live.org/

[13] LIVECD http://www.mirphakblog.com.ar/?q=node/38

[14] SQUASHFS. http://www.squashfs-lzma.org/sqlzma.txt

[15] AUFS. http://aufs.cvs.sourceforge.net/viewvc/aufs/aufs/

[16] LINUX KERNEL. http://www.kernel.org/

[17] NET_SNMP. http://www.net-snmp.org/docs/man/

[18] FPING. http://fping.sourceforge.net/

[19] IPSEC. http://www.ipsec-howto.org/ipsec-howto.pdf

Tesis

[20] Barragán, Santiago; Análisis Diseño y Prototipo de un servicio LDAP para la Polired; 2007.

5.2 ANEXOS

5.2.1 ANEXO A – GUÍA PARA LA ENTREVISTA DE LA GESTIÓN DE SERVICIOS DE TI EN LOS LABORATORIOS

Entrevista

Objetivo General

Conocer la forma en la que se gestionan los servicios de TI basados en ITIL, del sistema operativo Linux en los laboratorios de la carrera de Ingeniería en Electrónica y Redes de Información basados en la metodología ITIL.

Instrucciones

Señalar con una X la opción indicada por el entrevistado y llenar los espacios destinados para ampliar las respuestas en caso de ser necesario.

Información General

Fecha: _____

(AAAA/MM/DD)

Soporte a los Servicios de TI

Ámbito: Gestión de incidentes y problemas

1.- ¿De presentarse algún problema en que tiempo se lo soluciona?

☐ Menos de una hora.
☐ Menos de un día.
☐ Menos de una semana.
☐ Más de una semana.

2.- ¿Se tiene algún proceso formal para registrar un problema sucedido?

☐ Si ☐ No

En caso de responder SI, especifique el proceso:

3.- ¿Se monitorean los cambios realizados después de solucionar un problema?

☐ Si ☐ No

4.- ¿Se investigan las causas que ocasionaron algún incidente?

☐ Si ☐ No

5.- ¿Se cuenta con algún procedimiento de prevención de problemas?

☐ Si ☐ No

En caso de responder SI, especifique el procedimiento:

Ámbito: Gestión de configuraciones

6.- Se lleva un registro actualizado de las configuraciones de:

☐ Software ☐ Hardware ☐ Ninguno

7.- Se realizan auditorías periódicas de las configuraciones de:

☐ Software ☐ Hardware ☐ Ninguno

Ámbito: Gestión de cambios

8.- ¿Se tiene algún procedimiento formal para la petición de nuevos servicios y/o paquetes de software para implementar en el sistema Linux?

☐ Si ☐ No

9.- ¿Se lleva un registro formal de los cambios realizados en el sistema Linux?

☐ Si ☐ No

10.- ¿Los cambios realizados son evaluados?

☐ Si ☐ No

11.- ¿Se cuenta con algún procedimiento para evitar cambios no autorizados?

☐ Si ☐ No

Ámbito: Gestión de versiones

12.- ¿Las versiones anteriores (archivos de configuraciones) del sistema operativo Linux son respaldadas de alguna forma para poder volver a implementarlas de ser necesario posteriormente?

☐ Si ☐ No

En caso de responder SI, especifique de qué forma se respaldan las configuraciones anteriores.

Provisión de Servicios

Ámbito: Gestión de nivel de servicio

13.- ¿Se tienen definidos todos los servicios que brinda el sistema Linux a los usuarios?

☐ Si ☐ No

14.- ¿Se ha definido niveles de servicios, rendimiento o tiempos de respuesta con los usuarios de los laboratorios?

☐ Si ☐ No

Ámbito: Gestión de capacidad

15.- Antes de implementar (instalar) el sistema operativo Linux con todos sus componentes, ¿se hace un análisis de la capacidad de hardware requerida?

☐ Si ☐ No

16.- ¿Se realizan controles periódicos del rendimiento del hardware con respecto al uso del sistema operativo Linux?

☐ Si ☐ No

17.- ¿Se sigue una planificación para ampliaciones de capacidad de hardware a futuro en los laboratorios?

☐ Si ☐ No

Ámbito: Gestión de Continuidad y Disponibilidad

18.- ¿Se evalúan posibles riegos que atenten con la continuidad del servicio brindado por el sistema Linux?

☐ Si ☐ No

19.- ¿Se tiene definidos procedimientos a cumplir, posteriores a fallos no previstos para recuperación de errores del sistema operativo Linux?

☐ Si ☐ No

20.- ¿Se han determinado cronogramas de revisión y mantenimiento preventivo para los laboratorios?

☐ Si ☐ No

Ámbito: Gestión financiera

21.- ¿Se tienen medidos los costos asociados a los servicios de TI que brindan los laboratorios?

☐ Si ☐ No

Ámbito: Gestión de seguridad

22.- ¿Se tienen definidas políticas de seguridad a nivel de: ¿

☐ Software ☐ Hardware ☐ Ninguno

23.- ¿Se tienen definidas políticas de seguridad en contra de acceso de personal no autorizado a los laboratorios?

☐ Si ☐ No

24.- ¿Se han definido políticas en cuanto a la confidencialidad, integridad y disponibilidad de la información del sistema Linux?

☐ Si ☐ No

En caso de responder SI, especifique ¿Cuáles son esas políticas?

_____

Información de configuración actual necesaria

25.- ¿Con qué distribución de Linux se trabaja actualmente en los laboratorios?

☐ Fedora
☐ Debian
☐ Red Hat
☐ CentOS

26.- ¿Cuál es la principal razón para usar la distribución anteriormente señalada?

☐ Disponibilidad de paquetes.
☐ Facilidad en el manejo.
☐ No se conoce la razón.
☐ Otro. Especifique

27.- Especifique ¿cuáles son las características (marca y modelo) de los componentes en las computadoras de los laboratorios?

	Marca	Modelo
Procesador		
Tarjeta de red		
Disco duro		
Monitor		
Tarjeta de Video		
Unidad de CD/DVD		
Teclado		

5.2.2 ANEXO B - ENCUESTA DE LA SITUACIÓN ACTUAL Y DE REQUERIMIENTOS

Encuesta

Fecha: _____
(AAAA/MM/DD)

Objetivo General

Determinar por medio de un sondeo a los profesores que hagan uso del sistema operativo Linux en los laboratorios, el software y paquetes requeridos con los que contará la nueva distribución.

Información General

En las siguientes preguntas, marque con una X la o las opciones que crea convenientes.

1.-. ¿Qué materias dictadas por usted requieren del sistema operativo Linux en los laboratorios?

- ☐ Programación
- ☐ Sistemas Operativos.
- ☐ Prog. Orientada a Objetos
- ☐ Prog con Herram. Visuales
- ☐ Bases de Datos.
- ☐ Aplicaciones Distribuidas.
- ☐ Seguridad en Redes.
- ☐ Gestión de Redes.
- ☐ Redes e Intranets.
- ☐ WLAN.
- ☐ Hardware de Conectividad
- ☐ Evaluación de Redes.

2.-¿Aproximadamente cuántas horas por semana utiliza el sistema Linux?

- ☐ De 1 a 3 horas.
- ☐ De 4 a 6 horas.
- ☐ Más de 7 horas.

Percepción del Nivel de Servicio

En las siguientes preguntas, califique los servicios del 1 al 5, siendo 1 la calificación más baja y 5 la más alta.

3.- Tiempo de respuesta a solicitudes de instalación de nuevos paquetes.

☐1 ☐2 ☐3 ☐4 ☐5

4.- Disponibilidad de herramientas al momento de querer usarlas.

☐1 ☐2 ☐3 ☐4 ☐5

5.- Solución eficaz por parte de los administradores a problemas suscitados en los laboratorios.

☐1 ☐2 ☐3 ☐4 ☐5

6.- Nivel de soporte técnico ofrecido por parte de la administración del laboratorio.

☐1 ☐2 ☐3 ☐4 ☐5

Requerimientos

7.-¿Cuáles son los componentes BÁSICOS que considera debe tener un sistema operativo basado en Linux?

- ☐ Shell (intérprete de comandos).
- ☐ Programas para manipular directorios y ficheros (tar, bzip2, cat, mv, mkdir, etc.).
- ☐ Programas para administración de permisos (chroot, chmod, chown, etc).
- ☐ Programas para búsqueda en disco (find, locate, etc.).
- ☐ Programas para monitorizar el sistema y sus procesos (df, top, uptime, etc.).
- ☐ Programas para manejo de red (ifconfig, ping, telnet, ftp, ip, etc).
- ☐ Editor de texto en modo consola (vim).
- ☐ Interfaz gráfica amigable (KDE o Gnome).
- ☐ Otros (especifique):_____

8.- Especificar los programas ESTRICTAMENTENECESARIOS para impartir sus clases de laboratorio.

- En la columna “Paquete” indique el nombre del paquete de software que usted requiere.

Paquete	Uso	Materia
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	

	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	
	1 clase ☐ Todo el semestre ☐	

- En la columna de “Uso” indique si el paquete va a ser utilizado únicamente durante una clase específica (alrededor de una semana), o a lo largo de todo el semestre.
- En la columna “Materia” indique el nombre de la materia en la que se va a utilizar el paquete (llene la columna en caso de que usted imparta más de una materia)

5.2.3 ANEXO C – FHS (FILESYSTEM HIERARCHY SYSTEM)

Es una norma que define los directorios principales y sus contenidos en el sistema operativo GNU/Linux y otros sistemas de la familia Unix. Estos archivos se organizan en distintos directorios cuya función y nombre son estándar para todos los sistemas UNIX.Estos directorios pueden clasificarse en:

- Estáticos: Contiene archivos que no cambian sin la intervención del administrador (root), sin embargo, pueden ser leídos por cualquier otro usuario. (Por ejemplo /bin, /sbin, /opt, /boot, /usr/bin, etc)

- Dinámicos: Contiene archivos que pueden cambiar sin la intervención del administrador, y pueden leerse y escribirse (algunos sólo por su respectivo usuario y el root).

  Contienen configuraciones, documentos, etc. Para estos directorios, es recomendable una copia de seguridad con frecuencia, o mejor aún, deberían ser montados en una partición aparte en el mismo disco, como por ejemplo, montar el directorio /home en otra partición del mismo disco, independiente de la partición principal del sistema; de esta forma, puede repararse el sistema sin afectar o borrar los documentos de los usuarios. (Por ejemplo: /var/mail, /var/spool, /var/run, /var/lock, /home...)

- Compartidos: Contiene archivos que se pueden encontrar en un ordenador y utilizarse en otro, o incluso compartirse entre usuarios.

- Restringidos: Contiene ficheros que no se pueden compartir, son únicamente modificables por el administrador. (Por ejemplo /etc, /boot, etc)

- **El sistema de archivos raíz (/)**

Los datos contenidos en '/' deben ser suficientes para realizar 3 operaciones críticas: arrancar, reparar y recuperar el sistema. EL kernel del sistema operativo debe ubicarse en /boot o en /.

Directorio	Descripción
/bin	Comandos binarios esenciales para uso de todos los usuarios. /bin no puede contener directorios y deben residir aquí los binarios requeridos cuando no se ha montado ningún otro sistema de archivos el resto puede ser alojado en /usr/bin.
/boot	Archivos estáticos del cargador de arranque. /boot contiene todos los datos que el kernel necesita antes de comenzar a ejecutar programas de modo usuario, es decir, todos los datos involucrados en el proceso de arranque, archivos de mapa de sector y sectores de arranque maestro.
/dev	Archivos de dispositivos.
/etc	Archivos de configuración del sistema, específicos del host. El directorio /etc debe contener únicamente archivos de configuración y contener el directorio /opt
/lib	Librerías compartidas esenciales y módulos del kernel.
/media	Puntos de montaje para medios removibles (floppy, cdrom, cdrecorder y zip debieran existir aquí.
/mnt	Puntos de montaje temporal.
/opt	Software opcional de aplicación. Estas aplicaciones, utilizan el directorio de usuario para guardas sus configuraciones, y de esta forma, cada usuario puede tener una configuración diferente, de la misma aplicación.
/proc	El directorio `/proc` contiene "archivos" especiales que o bien extraen información del kernel o bien la envían a éste.

/sbin	Binarios esenciales del sistema exclusivos del usuario root. /sbin es el responsable de sostener los binarios necesarios para realizar las 3 operaciones básicas ya discutidas.
/srv	Datos que el sistema 'ofrece' a otros (www, ftp, rsync, cvs).
/tmp	Archivos temporales. /tmp debe existir y ser accesible para programas que necesiten de datos temporales, es decir, datos cuyo contenido no necesariamente es el mismo tras invocaciones sucesivas. Se recomienda borrar todo archivo alojado en /tmp tras el arranque del sistema.
/usr	Jerarquía secundaria de los datos de usuario; contiene la mayoría de las utilidades y aplicaciones multiusuario. En otras palabras, contiene los archivos compartidos de sólo lectura. Este directorio puede incluso ser compartido con otras computadoras.
/var	Datos variables.

A continuación se describen los archivos y directorios más importantes que contienen los directorios anteriormente citados.

- **El directorio /etc**

Aquí residen los archivos de configuración del sistema, son usados por el administrador y/u otros programas para almacenar y leer datos que alteran de una u otra forma su funcionamiento. El FHS prohíbe la existencia de binarios en este directorio.

Los siguientes directorios o enlaces simbólicos son requeridos en /etc:

Directorio	Descripción
opt	Archivos de configuración específicos para software complementario deben estar dentro de este directorio.
X11	Configuración para el sistema XWindow.

Los siguientes archivos, o enlaces simbólicos a los archivos, deben estar en /etc:

Archivo	Descripción
exports	Lista de control de acceso para el sistema de archivos NFS.
fstab	Información estática acerca del sistema de archivos.
gateways	Archivo que lista los gateways para enrutamiento.
hosts	Información estática acerca de los nombres de hosts.
Inittab	Archivo de configuración para init.
Issue	Mensaje de pre-login y archivo de identificación.
ld.so.conf	Lista de directorios extra para buscar librerías compartidas.
motd	Archivo con el mensaje de post-login del día.
mtab	Información dinámica acerca de los sistemas de archivos.
networks	Información estática de los nombres de red.
passwd	El archivo de contraseñas.
profile	Un archivo amplio de inicialización del sistema para sh shell logins.
protocols	Listado de protocolos IP.
resolv.conf	Archivo de configuración resolver.
services	Nombres de los puertos para los servicios de red.
syslog.conf	Archivo de configuración para el syslogd.

- **El directorio /dev**

Este directorio alberga archivos de dispositivos del sistema.

Los nombres de dispositivos que comienzan con hd acceden a discos duros. /dev/hda1 hace referencia a la primera partición en /dev/hda. Los nombres de dispositivos que empiezan con sd son dispositivos SCSI.

Los nombres de dispositivos que empiezan por /dev/tty hacen referencia a las consolas virtuales del sistema (accequible mediante las teclas alt-F1, alt-F2, etc). Los nombres que empiezan con /dev/pty hacen referencia a pseudoterminales, usados para proporcionar un terminal a sesiones remotas.
/dev/null es usado como un “agujero negro” cualquier dato enviado a este dispositivo desaparecerá.

- **El directorio /mnt**

Este directorio es provisto para que el administrador pueda montar temporalmente un sistema de archivos si es necesario. No se debe usar este directorio para instalar programas.

- **El directorio /lib**

Contiene las imágenes de librerías compartidas necesarias para ejecutar los binarios en /bin y /sbin. Estas imágenes son necesarias para arrancar el sistema y ejecutar comandos en el sistema de archivos de root.

Al menos uno de los siguientes archivos son requeridos en /lib:

Archivo	Descripción
libc.so.*	La librería C dinámicamente enlazada.
ld*	El enlazador/cargador en tiempo de ejecución.

Los siguientes directorios, o enlaces simbólicos a directorios deben estar en /lib:

Archivo	Descripción
modules	Módulos de kernel cargables.

- **El directorio /bin**

No debe haber subdirectorios en /bin. Los siguientes comandos o enlaces simbólicos deben encontrarse en /bin.

Comando	Descripción
cat	Utilidad para concatenar archivos a una salida estándar
chgrp	Utilidad para cambiar la propiedad de un archivo a un grupo.
chmod	Utilidad para cambiar los permisos de acceso a un archivo.
chown	Utilidad para cambiar propietario y grupo de un archivo
cp	Utilidad para copiar archivos y directorios.
csh	El Shell de C
date	Utilidad para imprimir o colocar la fecha y hora del sistema
dd	Utilidad para convertir y copiar un archivo.
df	Utilidad para reportar el espacio usado en disco de un sistema de archivos.
drnesg	Utilidad para para imprimir o controlar el buffer de mensajes del kernel.

echo	Utilidad para desplegar una línea de texto
false	Utilidad para no hacer nada, no exitosamente.
gzip	La utilidad de compresión de GNU.
gunzip	La utilidad de descompresión de GNU.
hostname	Utilidad para mostrar o colocar el hostname del sistema.
kill	Utilidad para mandar señales a los procesos.
ln	Utilidad para crear vínculos entre archivos
login	Utilidad para empezar una sesión en el sistema.
ls	Utilidad para mostrar el contenido de los directorios
mkdir	Utilidad para crear directorios.
mknod	
more	Utilidad para ver ficheros página por página, pantalla por pantalla
mount	Utilidad para montar un sistema de archivos.
mv	Utilidad para mover o renombrar archivos.
netstat	La utilidad para estadísticas de red.
ping	La utilidad de red de ICMP.
ps	Utilidad para reportar el estado de un proceso.
pwd	Utilidad para imprimir el nombre del directorio de trabajo actual.
rm	Utilidad para remover archivos o directorios.
rmdir	Utilidad para remover directorios vacios.
sed	El editor de caracteres sed.
sh	El Shell de comandos Bourne.
sstty	Utilidad para cambiar e imprimir configuraciones del terminal.
su	Utilidad para cambiar el ID de usuario.
sync	Utilidad para vaciar los buffers del sistema de archivos.
tar	La utilidad de archivar tar.
true	Utilidad para no hacer nada, exitosamente.
umount	Utilidad para desmontar archivos del sistema.
uname	Utilidad para imprimir información del sistema.

- **El directorio /sbin**

Utilidades usadas por el administrador del sistema y otros comandos usados sólo por el /root son almacenadas en /sbin, /usr/sbin y en /usr/local/sbin. /sbin contiene binarios esenciales para el arranque, restauración, recuperación y/o reparación del sistema.

Los siguientes comandos o enlaces simbólicos a comandos son requeridos es /sbin.

Comando	Descripción
shutdown	Comando par apagar el sistema

Los siguientes archivos o enlaces simbólicos a los archivos, deben estar en /sbin. Si el correspondiente subsistema es instalado.

Comando	Descripción
fastboot	Reiniciar el sistema sin revisar los discos.
fasthalt	Detener el sistema sin revisar los discos.
fdisc	Manipulador de la tabla de particiones.
fsck	Utilidad para revisar y repara el sistema de archivos.
getty	El programa getty.
halt	Comando para detener el sistema.
ifconfig	Configurar una interfaz de red.
init	Proceso inicial.
mkfs	Comando para construir un sistema de archivos.
mkswap	Comando para asignar un área para swap.
reboot	Comando para reiniciar el sistema.
route	Utilidad para la tabla de enrutamiento IP.

swapon	Habilita paginación e intercambio.
swapoff	Deshabilita paginación e intercambio.
update	Demonio para vaciar periódicamente los buffers del sistema de archivos.

- **El directorio /var**

Contiene datos de carácter variable, en esta categoría están especialmente incluidos los archivos y directorios de 'spooling' (mail, impresión) y los archivos y directorios de 'logging' (mensajes del kernel, etc). /var tiene un comportamiento mixto, hay en el, datos compartibles y datos no compartibles, un ejemplo del primer caso son los datos que componen la 'cola' de emails, un ejemplo de los segundos son los archivos bajo /var/lock. Se recomienda a los programadores y usuarios en general, no crear directorios directamente sobre /var.

Los siguientes directorios, o enlaces simbólicos a directorios deben estar en /var:

Directorio	Descripción
cache	Datos de la cache de aplicaciones.
crash	Se depositan datos e información, referentes a las caídas o errores del sistema operativo.
games	Datos variables de juegos.
lib	Información de estado variable.
local	Datos variables para /usr/local
lock	Archivos bloqueados.
log	Archivos y directorios de log.
mail	Archivos de mail del usuario.
opt	Datos variables para /opt
run	Datos relevantes para los programas q están corriendo

spool	¨carretes¨ de tareas a la espera de ser procesadas.
temp	Archivos temporales guardados entre reinicios del sistema.
yp	Base de datos del NIS.

- **El directorio /usr**

/usr contiene datos de sólo lectura compartidos. En gran mediad bajo /usr el sistema se duplica a si mismo, repite la estructura presentada por / con la excepción de algunos directorios relevantes.

Los siguientes directorios, o enlaces simbólicos a directorios deben estar en /usr:

Archivo	Descripción
bin	Contiene la mayoría de los comandos de usuario.
include	Archivos de cabecera incluidos los de los programas en C.
lib	Contiene librerías.
local	Jerarquía local
sbin	Binarios del sistema no vitales.
share	Datos independientes de la arquitectura
games	Binarios de juegos y educación
src	Código fuente.

La jerarquía /usr/local es usada por el administrador del sistema cuando se instala software localmente. Necesita ser seguro para ser sobrescrito cuando el software del sistema es actualizado. Puede ser usado por programas y datos que son compartidos entre un grupo de hosts.

Los siguientes directorios, o enlaces simbólicos a directorios deben estar en /usr/local:

Archivo	Descripción
bin	Contiene binarios locales..
etc	Configuración del sistema especifica del host para binarios locales.
games	Binarios de juegos locales
include	Archivos de cabecera incluidos los de los programas en C locales.
lib	Contiene librerías locales.
man	Contiene manuales en línea locales.
sbin	Binarios del sistema no vitales locales.
share	Datos independientes de la arquitectura locales.
src	Código fuente local.

5.2.4 ANEXO D – GUÍA DE IMPLEMENTACIÓN

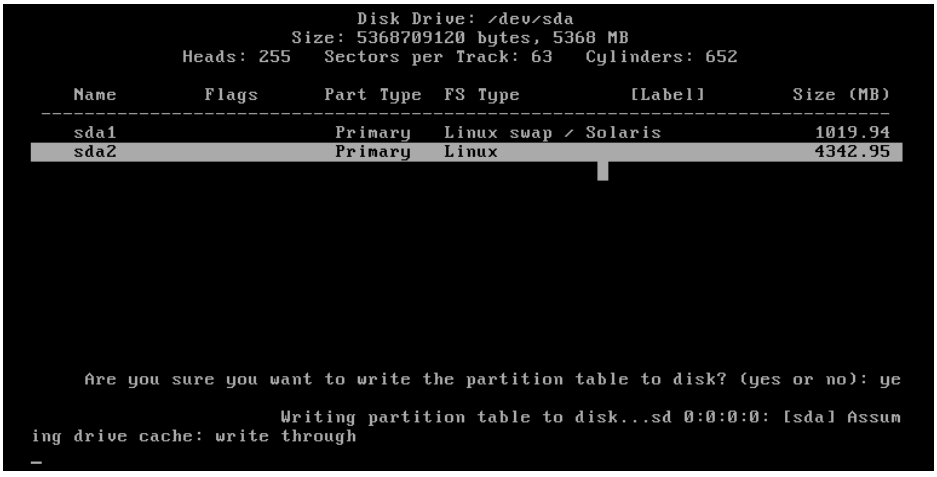
Para seguir esta guía se debe cumplir con los siguientes requisitos:

- Distribución anfitriona LFS live CD 6.3
- Máquina Virtual con 8GB en disco y 512 MB en memoria RAM (VMware ®)
- Todas las herramientas y parches a instalar deben estar en el directorio */mnt/lfs/sources*

Todos los paquetes necesarios para la instalación de los sistema temporal y básico están dentro del LiveCD de Linux from Scratch. Esta guía contiene todo el proceso de implementación hasta la culminación del sistema base que albergará todas las herramientas.

Script 1. Preparativos para la construcción.

Se construyen primero las particiones que albergarán al sistema y la partición de intercambio. Se utiliza el comando *cfdisk* y se particiona el disco como se muestra en la siguiente figura:



```
#############################################################################################
#       CREACION DEL SISTEMA DE ARCHIVOS, INICIAR Y ACTIVAR LAS PARTICIONES       #
#############################################################################################
#
# Creación del sistema de archivos de tipo ext3 en la segunda partición con el
# comando:
mke2fs -j /dev/sda2
# Inicialización de la partición de intercambio:
mkswap /dev/sda1
# Activación la partición de swap se utiliza el comando:
swapon –s
#############################################################################################
#                          MONTAR LA PARTICION PRINCIPAL                                    #
#############################################################################################
#
# Crea el punto de montaje y se monta el sistema de archivos
export LFS=/mnt/lfs
mkdir -p $LFS
mount -t ext3 /dev/sda2$LFS
#############################################################################################
#               CREACION DE DIRECTORIOS Y FUENTES PARA INSTALACIONES                        #
#############################################################################################
#
# Creación del directorio en donde se guardaran los parches y directorios
# durante el proceso de instalación.
mkdir $LFS/sources
# Este directorio debe ser escribible para poder guardar posteriormente más
# paquetes y pegajoso (sticky) es decir q solo el propietario de un archivo
# puede borrarlo, esto se hace con el comando:
chmod a+wt $LFS/sources
# Creación del directorio para compilar todas las herramientas que van a formar
parte
# del sistema temporal
mkdir $LFS/tools
# Crear un link estático en el sistema anfitrión que apunte al directorio
anteriormente
ln -s $LFS/tools /
#############################################################################################
#                          CREACION UN GRUPO Y UN USUARIO                                   #
#############################################################################################
#
```

```
# Creación del grupo y el usuario lfs
groupadd lfs
useradd -s /bin/bash -g lfs -m -k /dev/null lfs
# Asignación de una contraseña a lfs
passwd lfs
# Como se va a trabajar sobre los directorios $LFS/tools y $LFS/sources se le
# da al usuario ‘lfs’ la propiedad de los mismos con los comandos:
chown lfs $LFS/tools
chown lfs $LFS/tools
# Finalmente se entra como usuario lfs
su – lfs
#################################################################################################
#                                CONFIGURACION DEL ENTORNO                                      #
# El entorno del usuario creado, mediante la creación de dos nuevos ficheros de    #
# inicio para el intérprete de  comandos bash estos ficheros son: .bash_profile y  #
# .bashrc                                                                                       #
# .bash_profile es el más importante. Los comandos escritos en el son escritos y   #
# ejecutados cada vez que el usuario entra en el sistema.                                       #
# .bashrc es leído cuando el usuario arranca un subshell.                                       #
#################################################################################################
#
# El comando exec env -i.../bin/bash reemplaza el intérprete de comandos en
# ejecución por uno nuevo con un entorno completamente vacío. Esto asegura que en el
# entorno de construcción no aparezcan variables de entorno indeseadas procedentes
# del sistema anfitrión
#
cat > ~/.bash_profile << "EOF"
exec env -i HOME=$HOME TERM=$TERM PS1='\u:\w\$ ' /bin/bash
EOF
#
# Al entrar como usuario lfs el intérprete de comandos es de ingreso y lee el
# /etc/profile del sistema anfitrión y luego lee .bash_profile.
#
cat > ~/.bashrc << "EOF"
set +h
umask 022
LFS=/mnt/lfs
LC_ALL=POSIX
PATH=/tools/bin:/bin:/usr/bin
export LFS LC_ALL PATH
EOF
```

```
# Finalmente se debe cargar el perfil de usuario recién creado, para tener el
# entorno listo para construir el sistema temporal
source ~/.bash_profile
```

Script 2. Instalación de Binutils, GCC y Glibc. Ajuste de Herramientas.

Nota: Los nombres de cada paquete deben ser revisados antes de la ejecución del script

```
#############################################################################################
#                    INSTALACION DE BINUTILS, GCC Y GLIBC                                   #
#############################################################################################
#
#                          Instalacion de Binutils
#
cd $LFS/sources
tar –xf binutils.tar.gz
cd binutils
# Construccion de binutils
CC="gcc -B/usr/bin/" ../binutils-2.17/configure --prefix=/tools --disable-werror
# Compilacion e instalacion
make
make install
# Preparacion del enlazador para la segunda fase
make -C ld clean
make -C ld LIB_PATH=/tools/lib
cp ld/ld-new /tools/bin
#
#                          Instalacion de GCC
#
cd $LFS/sources
tar –xf gcc.tar.gz
cd gcc
# Construccion de GCC
CC="gcc    -B/usr/bin/"    ../gcc-4.1.2/configure    --prefix=/tools    --with-local-
prefix=/tools --enable-shared --enable-languages=c
# Compilacion e instalacion
make bootstrap
make install
```

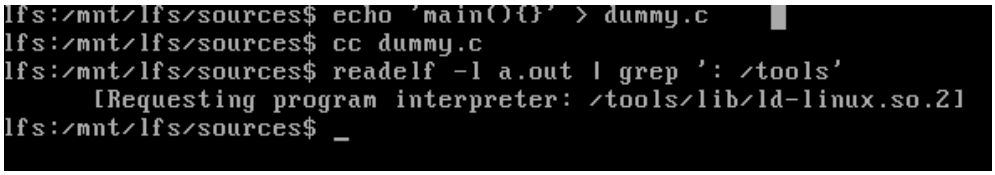
```
ln -s gcc /tools/bin/cc
#
#                   Instalacion de las cabeceras del API de Linux
#
# Se instalan para exponer la API del núcleo para ser usada por Glibc, debido a que
# el núcleo de Linux necesita exponer una API para ser usada por la librería de C
# del sistema.
#
cd $LFS/sources
tar –xf linuxheaders.tar.gz
cd linuxheaders
make mrproper
make headers_check
make INSTALL_HDR_PATH=dest headers_install
cp -r dest/include/* /tools/include
#
# Instalacion de Glibc
#
cd $LFS/sources
tar –xf glibc.tar.gz
cd glibc
#
# Construccion de Glibc
#
../glibc-2.5.1/configure --prefix=/tools --disable-profile --enable-add-ons \\
--enable-kernel=2.6.0 --without-gd --with-headers=/tools/include
#
# Compilacion e instalacion
make
make install
#
#############################################################################################
#                           AJUSTE DE LAS HERRAMIENTAS                                      #
#############################################################################################
# Una vez instaladas todas las librerías de C temporales, las herramientas que se
# compilaran para el resto del sistema temporal deberán enlazarse contra estas
# librerías
mv -v /tools/bin/{ld,ld-old}
mv -v /tools/$(gcc -dumpmachine)/bin/{ld,ld-old}
mv -v /tools/bin/{ld-new,ld}
ln -sv /tools/bin/ld /tools/$(gcc -dumpmachine)/bin/ld
```

```
# Lo que sigue es apuntar GCC al nuevo enlazador dinámico Esto se verificará en el
# fichero Specs ya que "/lib/ld-linux.so.2" se verá reemplazado por "/tools/lib/ld-
# linux.so.2". El enlazador dinámico de la distribución anfitriona es:
# ld-linux.so.2.
gcc -dumpspecs | sed 's@^/lib/ld-linux.so.2@/tools&@g' > `dirname \\
$(gcc -print-libgcc-file-name)`/specs
```

En este punto se deberá comprobar si la compilación y enlazado de las herramientas funciona de forma esperada comprobando el prefijo del enlazador dinámico, Esto se realiza creando un pequeño programa en C (*dummy.c*), compilándolo (*cc dummy.c*) y viendo la información contenida en los segmentos de las cabeceras del archivo objeto con formato ELF[35](*readelf -l a.out*):

```
echo 'main(){}' > dummy.c
cc dummy.c
readelf -l a.out | grep ': /tools'
```

Luego de ejecutar los comandos anteriores se tiene la salida indicada en la siguiente figura:



Como se observa en la anterior figura el enlazador dinámico esta dentro de /tools/lib con lo cual se comprueba que el compilador y enlazador del sistema temporal funcionan correctamente.

Finalmente se borran los archivos creados para la comprobación anterior.

[35] ELF: Executable and Linkable Format, es un formato estándar de archivos ejecutables, librerías, archivos objeto, etc.

```
rm dummy.c a.out
```

Script 3. Instalación de herramientas para ejecutar los bancos de pruebas e Instalación de herramientas para brindar mayor funcionalidad

```
#########################################################################################
#        INSTALACION DE LAS HERRAMIENTAS PARA EJECUTAR LOS BANCOS DE PRUEBAS       #
#########################################################################################
#
#                              Instalacion de Tcl
#
cd $LFS/sources
tar –xf tcl.tar.gz
cd tcl
cd unix
#
# Construccion de Tcl
cd unix
./configure --prefix=/tools
#
# Compilacion e instalacion
make
make install-private-headers
make install
#
ln -s tclsh8.4 /tools/bin/tclsh
#
#                           Instalacion de Expect
#
cd $LFS/sources
tar –xf expect.tar.gz
cd expect
patch -Np1 -i ../expect-5.43.0-spawn-1.patch
# Se fuerza al guión de configuración de Expect a usar /bin/sttyen lugar de
# /usr/local/bin/stty que podría estar en el sistema anfitrión.
cp configure{,.bak}
sed 's:/usr/local/bin:/bin:' configure.bak > configure
#
# Construccion de expect
```

```
./configure --prefix=/tools --with-tcl=/tools/lib \\
--with-tclinclude=/tools/include --with-x=no
# Compilacion e instalacion
make
make SCRIPTS=”” install
#
#                          Instalacion de DejaGNU
#
cd $LFS/sources
tar –xf dejagnu.tar.gz
cd dejagnu
#
# Construccion de DejaGNU
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
################################################################################################
#          HERRAMIENTAS PARA BRINDAR MAYOR FUNCIONALIDAD                                       #
################################################################################################
#
#                         Instalacion de GCC fase 2
#
cd gcc
patch -Np1 -i ../gcc-4.1.2-specs-1.patch
#
# Construccion de GCC
../gcc-4.1.2/configure --prefix=/tools --with-local-prefix=/tools \\
--enable-clocale=gnu --enable-shared --enable-threads=posix  \\
--enable-__cxa_atexit --enable-languages=c,c++ --disable-libstdcxx-pch
#
# Compilacion e instalación
make
make install
#
#                         Instalacion de Binutils fase 2
#
cd binutils
#
```

```
# Construccion de Binutils
../binutils-2.17/configure --prefix=/tools --with-lib-path=/tools/lib
#
# Compilacion e instalación
make
make install
#
# Finalmente se prepara el enlazador para la fase de reajuste que se dará en la
# construcción del sistema definitivo
make -C ld clean
make -C ld LIB_PATH=/usr/lib:/lib
cp ld/ld-new /tools/bin
#
#                        Instalacion de Ncurses
#
cd $LFS/sources
tar –xf ncurses.tar.gz
cd ncurses
#
# Construccion de Ncurses
./configure --prefix=/tools --without-debug --without-ada --enable-overwrite
#
# Compilacion e instalación
make
make install
#
#                        Instalacion de Bash
#
cd $LFS/sources
tar –xf bash.tar.gz
cd bash
patch -Np1 -i ../bash-3.2-fixes-5.patch
#
# Construccion de Bash
./configure --prefix=/tools --without-bash-malloc
#
# Compilacion e instalación
make
make install
# Como paso final se debe crear un enlace para los programas que usan sh como
# intérprete de comandos
```

```
ln -vs bash /tools/bin/sh
#
#                       Instalacion de Bzip2
#
cd $LFS/sources
tar –xf bzip2.tar.gz
cd bzip2
#
# Compilacion e instalación
make
make PREFIX=/tools install
#
#                       Instalacion de Coreutils
#
cd $LFS/sources
tar –xf coreutils.tar.gz
cd coreutils
#
# Construccion de coreutils
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Findutils
#
cd $LFS/sources
tar –xf findutils.tar.gz
cd findutils
#
# Construccion de findutils
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Gawk
#
cd $LFS/sources
```

```
tar –xf gawk.tar.gz
cd gawk
#
# Construccion de gawk
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Gettext
#
cd $LFS/sources
tar –xf gettext.tar.gz
cd gettext
#
# Construccion de gettext
cd gettext-tools
./configure --prefix=/tools --disable-shared
#
# Compilacion e instalación
make -C gnulib-lib
make -C src msgfmt
#
#                       Instalacion de Grep
#
cd $LFS/sources
tar –xf grep.tar.gz
cd grep
#
# Construccion de grep
./configure --prefix=/tools --disable-perl-regexp
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Gzip
#
cd $LFS/sources
tar –xf gzip.tar.gz
```

```
cd gzip
#
# Construccion de gzip
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Make
#
cd $LFS/sources
tar –xf make.tar.gz
cd make
#
# Construccion de make
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Patch
#
cd $LFS/sources
tar –xf patch.tar.gz
cd patch
#
# Construccion de patch
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Perl
#
cd $LFS/sources
tar –xf perl.tar.gz
cd perl
patch -Np1 -i ../perl-5.8.8-libc-2.patch
```

```
#
# Construccion de patch
./configure.gnu --prefix=/tools -Dstatic_ext='Data/Dumper Fcntl IO POSIX'
#
# Compilacion e instalación
make perl utilities
# Instalacion de herramientas con las librerías de perl
cp -v perl pod/pod2man /tools/bin
mkdir -p /tools/lib/perl5/5.8.8
cp -R lib/* /tools/lib/perl5/5.8.8
#
#                       Instalacion de Sed
#
cd $LFS/sources
tar –xf sed.tar.gz
cd sed
#
# Construccion de sed
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Tar
#
cd $LFS/sources
tar –xf tar.tar.gz
cd tar
#
# Construccion de tar
./configure --prefix=/tools
#
# Compilacion e instalación
make
make install
#
#                       Instalacion de Util-linux
#
cd $LFS/sources
```

```
tar –xf util-linux.tar.gz
cd util-linux
#
# Para que este paquete utilice las cabeceras y librerías instaladas en el
#  directorio /tools, se debe modificar el guión configure de la siguiente manera:
sed -i 's@/usr/include@/tools/include@g' configure
#
# Construccion de util-linux
./configure
#
#
# Compilacion e instalación
make -C mount mount umount
make -C text-utils more
# Finalmente se copian estas herramientas al directorio en donde están las
# herramientas temporales:
cp -v mount/{,u}mount text-utils/more /tools/bin
#
#                       Eliminacion de simbolos
#
strip --strip-debug /tools/lib/*
strip --strip-unneeded /tools/{,s}bin/*
#
# Finalmente se cambia el propietario de $LFS/tools del usuario lfs al root, esto se
# realiza como usuario root.
chown -R root:root $LFS/tools
```

Script 4. Construcción del sistema base

```
###################################################################################
#                       PREPARATIVOS PARA LA CONSTRUCCION                         #
###################################################################################
# Se prepara los sistemas de archivos virtuales del núcleo, para poblarlos
# y montarlos
mkdir -pv $LFS/{dev,proc,sys}
mknod -m 600 $LFS/dev/console c 5 1
mknod -m 666 $LFS/dev/null c 1 3
mount -v --bind /dev $LFS/dev
mount -vt devpts devpts $LFS/dev/pts
```

```
mount -vt tmpfs shm $LFS/dev/shm
mount -vt proc proc $LFS/proc
mount -vt sysfs sysfs $LFS/sys
#
# Se entra al entorno chroot
chroot "$LFS" /tools/bin/env -i \
  HOME=/root TERM="$TERM" PS1='\u:\w\$ ' \
  PATH=/bin:/usr/bin:/sbin:/usr/sbin:/tools/bin \
  /tools/bin/bash --login +h
#
# Se crea la estructura de directorios según FHS
#
mkdir -pv /{bin,boot,etc/opt,home,lib,mnt,opt}
mkdir -pv /{media/{floppy,cdrom},sbin,srv,var}
install -dv -m 0750 /root
install -dv -m 1777 /tmp /var/tmp
mkdir -pv /usr/{,local/}{bin,include,lib,sbin,src}
mkdir -pv /usr/{,local/}share/{doc,info,locale,man}
mkdir -v /usr/{,local/}share/{misc,terminfo,zoneinfo}
mkdir -pv /usr/{,local/}share/man/man{1..8}
for dir in /usr /usr/local; do
ln -sv share/{man,doc,info} $dir
done
mkdir -v /var/{lock,log,mail,run,spool}
mkdir -pv /var/{opt,cache,lib/{misc,locate},local}
ln -sv /tools/bin/{bash,cat,echo,grep,pwd,stty} /bin
ln -sv /tools/bin/perl /usr/bin
ln -sv /tools/lib/libgcc_s.so{,.1} /usr/lib
ln -sv /tools/lib/libstdc++.so{,.6} /usr/lib
ln -sv bash /bin/sh
touch /etc/mtab
#
# Se crea el archivo necesario para manejo de contraseñas en el sistema
#
cat > /etc/passwd << "EOF"
root:x:0:0:root:/root:/bin/bash
nobody:x:99:99:Unprivileged User:/dev/null:/bin/false
EOF
#
# Se crea el archivo necesario para manejo de permisos y grupos en el sistema
#
```

```
cat > /etc/group << "EOF"
root:x:0:
bin:x:1:
sys:x:2:
kmem:x:3:
tty:x:4:
tape:x:5:
daemon:x:6:
floppy:x:7:
disk:x:8:
lp:x:9:
dialout:x:10:
audio:x:11:
video:x:12:
utmp:x:13:
usb:x:14:
cdrom:x:15:
mail:x:34:
nogroup:x:99:
EOF
# La metodología recomienda la ejecución de los siguientes comandos para eliminar
# el mensaje “I have no name!” del símbolo del sistema.
exec /tools/bin/bash --login +h
# Archivos necesarios para mantener un registro con información sobre quienes
# están y estaban dentro del sistema.
touch /var/run/utmp /var/log/{btmp,lastlog,wtmp}
chgrp -v utmp /var/run/utmp /var/log/lastlog
chmod -v 664 /var/run/utmp /var/log/lastlog
#
################################################################################################
#                  INSTALACION DE PAQUETES BASICOS DEL SISTEMA                                 #
################################################################################################
#
#
#                           Cabeceras API de Linux
#
cd $LFS/sources
tar –xf glibc.tar.gz
cd glibc.tar.gz
# Se excluye el uso de /usr/include/scsi para ser reemplazado por el
# proporcionado por Glibc.
```

```
sed -i '/scsi/d' include/Kbuild
# Instalación de las cabeceras
make mrproper
make headers_check
make INSTALL_HDR_PATH=dest headers_install
cp -rv dest/include/* /usr/include
#
#                          Instalación de Man-Pages
#
cd $LFS/sources
tar –xf man-pages.tar.gz
cd man-pages
make install
#
#                         Instalación de Glibc
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf glibc-libidn.tar.gz
cd glibc-libidn
make install
cd $LFS/sources
tar -xf ../glibc-libidn-2.5.1.tar.gz
mv glibc-libidn-2.5.1 libidn
sed -i '/vi_VN.TCVN/d' localedata/SUPPORTED
sed -i \
's|libs -o|libs -L/usr/lib -Wl,-dynamic-linker=/lib/ld-linux.so.2 -o|' \
scripts/test-installation.pl
sed -i 's|@BASH@|/bin/bash|' elf/ldd.bash.in
mkdir -v ../glibc-build
cd ../glibc-build
#
# Configuración del paquete
../glibc-2.5.1/configure --prefix=/usr \
--disable-profile --enable-add-ons \
--enable-kernel=2.6.0 --libexecdir=/usr/lib/glibc
#
# Compilación e instalación del paquete
make
make -k check 2>&1 | tee glibc-check-log
grep Error glibc-check-log
```

```
touch /etc/ld.so.conf
make install
# Creación del archivo de configuración de Glibc
cat > /etc/nsswitch.conf << "EOF"
# Begin /etc/nsswitch.conf
passwd: files
group: files
shadow: files
hosts: files dns
networks: files
protocols: files
services: files
ethers: files
rpc: files
# End /etc/nsswitch.conf
EOF
cp -v --remove-destination /usr/share/zoneinfo/<xxx> \
/etc/localtime
# Creación del archivo para configurar el cargador dinámico del sistema
cat > /etc/ld.so.conf << "EOF"
# Begin /etc/ld.so.conf
/usr/local/lib
/opt/lib
# End /etc/ld.so.conf
EOF
#############################################################################
#                             REAJUSTAR LAS HERRAMIENTAS                    #
#############################################################################
#

mv -v /tools/bin/{ld,ld-old}
mv -v /tools/$(gcc -dumpmachine)/bin/{ld,ld-old}
mv -v /tools/bin/{ld-new,ld}
ln -sv /tools/bin/ld /tools/$(gcc -dumpmachine)/bin/ld
gcc -dumpspecs | sed \
  -e 's@/tools/lib/ld-linux.so.2@/lib/ld-linux.so.2@g' \
  -e '/\*startfile_prefix_spec:/{n;s@.*@/usr/lib/ @}' \
  -e '/\*cpp:/{n;s@$@ -isystem /usr/include@}' > \
  `dirname $(gcc --print-libgcc-file-name)`/specs
echo 'main(){}' > dummy.c
cc dummy.c -v -Wl,--verbose &> dummy.log
```

```
readelf -l a.out | grep ': /lib'
grep -o '/usr/lib.*/crt[1in].*succeeded' dummy.log
grep -B1 '^ /usr/include' dummy.log
grep 'SEARCH.*/usr/lib' dummy.log |sed 's|; |\n|g'
grep "/lib/libc.so.6 " dummy.log
grep found dummy.log
rm -v dummy.c a.out dummy.log
#
#                           Instalación de Binutils
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf binutils.tar.gz
cd binutils
mkdir -v ../binutils-build
cd ../binutils-build
#
# Configuración del paquete
../binutils-2.17/configure --prefix=/usr --enable-shared
#
# Compilación e instalación del paquete
make tooldir=/usr
make check
make tooldir=/usr install
cp -v ../binutils-2.17/include/libiberty.h /usr/include
#
#                           Instalación de GCC
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf gcc.tar.gz
cd gcc
sed -i 's/install_to_$(INSTALL_DEST) //' libiberty/Makefile.in
sed -i 's/^XCFLAGS =$/& -fomit-frame-pointer/' gcc/Makefile.in
sed -i 's@\./fixinc\.sh@-c true@' gcc/Makefile.in
sed -i 's/@have_mktemp_command@/yes/' gcc/gccbug.in
mkdir -v ../gcc-build
cd ../gcc-build
#
# Configuración del paquete
../gcc-4.1.2/configure --prefix=/usr \
```

```
--libexecdir=/usr/lib --enable-shared \
--enable-threads=posix --enable-__cxa_atexit \
--enable-clocale=gnu --enable-languages=c,c++
#
# Compilación e instalación del paquete
make
make –k check
../gcc-4.1.2/contrib/test_summary
make install
ln -sv ../usr/bin/cpp /lib
ln -sv gcc /usr/bin/cc
echo 'main(){}' > dummy.c
cc dummy.c -v -Wl,--verbose &> dummy.log
readelf -l a.out | grep ': /lib'
grep -o '/usr/lib.*/crt[1in].*succeeded' dummy.log
grep -B3 '^ /usr/include' dummy.log
grep 'SEARCH.*/usr/lib' dummy.log |sed 's|; |\n|g'
grep "/lib/libc.so.6 " dummy.log
grep found dummy.log
rm -v dummy.c a.out dummy.log
#
cd $LFS/sources
tar –xf berkeley-db
cd berkeley-db
patch -Np1 -i ../db-4.5.20-fixes-1.patch
cd build_unix
../dist/configure --prefix=/usr --enable-compat185 --enable-cxx
make
make docdir=/usr/share/doc/db-4.5.20 install
chown -Rv root:root /usr/share/doc/db-4.5.20
#
#                           Instalación de SED
#
cd $LFS/sources
tar –xf sed.tar.gz
cd sed
./configure --prefix=/usr --bindir=/bin --enable-html
make
make install
#
#                           Instalación de E2fsprogs (ext2)
```

```
#
# Preparativos para la construccion
cd $LFS/soruces
tar –xf e2fsprogs.tar.gz
sed -i -e 's@/bin/rm@/tools&@' lib/blkid/test_probe.in
mkdir -v build
cd build
#
# Configuración del paquete
../configure --prefix=/usr --with-root-prefix="" --enable-elf-shlibs
#
# Compilación e instalación del paquete
make
make install
make install-libs
#
#                          Instalación de Coreutils
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf coreutils.tar.gz
cd coreutils
patch -Np1 -i ../coreutils-6.9-uname-1.patch
patch -Np1 -i ../coreutils-6.9-suppress_uptime_kill_su-1.patch
patch -Np1 -i ../coreutils-6.9-i18n-1.patch
chmod +x tests/sort/sort-mb-tests
#
# Configuración del paquete
./configure --prefix=/usr
#
# Compilación e instalación del paquete
Make
make NON_ROOT_USERNAME=nobody check-root
echo "dummy:x:1000:nobody" >> /etc/group
su-tools nobody -s /bin/bash -c "make RUN_EXPENSIVE_TESTS=yes check"
sed -i '/dummy/d' /etc/group
make install
mv -v /usr/bin/{cat,chgrp,chmod,chown,cp,date,dd,df,echo} /bin
mv -v /usr/bin/{false,hostname,ln,ls,mkdir,mknod,mv,pwd,readlink,rm} /bin
mv -v /usr/bin/{rmdir,stty,sync,true,uname} /bin
mv -v /usr/bin/chroot /usr/sbin
```

```
mv -v /usr/bin/{head,sleep,nice} /bin
#
#                          Instalación de Iana-Etc
#
cd $LFS/sources
tar –xf iana-etc
cd iana-etc
make
make install
#
#                          Instalación de M4 (gestor de macros)
#
cd $LFS/sources
tar –xf m4.tar.gz
cd m4
./configure –prefix/usr
make
make install
#
#                          Instalación de Bison
#
cd $LFS/sources
tar –xf bison.tar.gz
cd bison
./configure --prefix=/usr
echo '#define YYENABLE_NLS 1' >> config.h
make
make install
#
#                          Instalación de Ncurses
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf ncurses.tar.gz
cd ncurses
patch -Np1 -i ../ncurses-5.6-coverity_fixes-1.patch
#
# Configuración del paquete
./configure --prefix=/usr --with-shared --without-debug --enable-widec
#
# Compilación e instalación del paquete
```

```
make
make install
chmod -v 644 /usr/lib/libncurses++w.a
mv -v /usr/lib/libncursesw.so.5* /lib
ln -sfv ../../lib/libncursesw.so.5 /usr/lib/libncursesw.so
for lib in curses ncurses form panel menu ; do \
  rm -vf /usr/lib/lib${lib}.so ; \
  echo "INPUT(-l${lib}w)" >/usr/lib/lib${lib}.so ; \
  ln -sfv lib${lib}w.a /usr/lib/lib${lib}.a ; \
done
ln -sfv libncurses++w.a /usr/lib/libncurses++.a
rm -vf /usr/lib/libcursesw.so
echo "INPUT(-lncursesw)" >/usr/lib/libcursesw.so
ln -sfv libncurses.so /usr/lib/libcurses.so
ln -sfv libncursesw.a /usr/lib/libcursesw.a
ln -sfv libncurses.a /usr/lib/libcurses.a
#
#                           Instalación de Procps
#
cd $LFS/sources
tar –xf procps.tar.gz
cd procps
make
make install
#
#                           Instalación de Libtool
#
cd $LFS/sources
tar –xf libtool.tar.gz
cd libtool
./configure –prefix=/usr
make
make install
#
#                           Instalación de Perl
#
cd $LFS/sources
tar –xf perl.tar.gz
cd perl
echo "127.0.0.1 localhost $(hostname)" > /etc/hosts
./configure.gnu --prefix=/usr \
```

```
   -Dman1dir=/usr/share/man/man1 \
   -Dman3dir=/usr/share/man/man3 \
   -Dpager="/usr/bin/less -isR"
make
make install
#
#                          Instalación de Readline
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf readline.tar.gz
cd readline
sed -i '/MV.*old/d' Makefile.in
sed -i '/{OLDSUFF}/c:' support/shlib-install
patch -Np1 -i ../readline-5.2-fixes-3.patch
#
# Configuración del paquete
./configure --prefix=/usr --libdir=/lib
#
# Compilación e instalación del paquete
make SHLIB_LIBS=-lncurses
make install
mv -v /lib/lib{readline,history}.a /usr/lib
rm -v /lib/lib{readline,history}.so
ln -sfv ../../lib/libreadline.so.5 /usr/lib/libreadline.so
ln -sfv ../../lib/libhistory.so.5 /usr/lib/libhistory.so
#
#                          Instalación de Zlib
#
cd $LFS/sources
tar –xf zlib.tar.gz
cd zlib
#
# Configuración del paquete
./configure --prefix=/usr --shared --libdir=/lib
#
# Compilación e instalación del paquete
make
make install
rm -v /lib/libz.so
ln -sfv ../../lib/libz.so.1.2.3 /usr/lib/libz.so
```

```
make clean
#
# Configuración del paquete
./configure --prefix=/usr
make
make install
chmod -v 644 /usr/lib/libz.a
#
#                           Instalación de Autoconf
#
cd $LFS/sources
tar –xf autoconf.tar.gz
cd autoconf
./configure --prefix=/usr
make
make install
#
#                           Instalación de Automake
#
cd $LFS/sources
tar –xf automake.tar.gz
cd automake
./configure --prefix=/usr
make
make install
#
#                           Instalación de Bash
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf bash.tar.gz
cd bash
tar -xvf ../bash-doc-3.2.tar.gz
sed -i "s|htmldir = @htmldir@|htmldir = /usr/share/doc/bash-3.2|" \
Makefile.in
patch -Np1 -i ../bash-3.2-fixes-5.patch
#
# Configuración del paquete
./configure --prefix=/usr --bindir=/bin --without-bash-malloc \
--with-installed-readline
#
```

```
# Compilación e instalación del paquete
make
sed -i 's/LANG/LC_ALL/' tests/intl.tests
sed -i 's@tests@& </dev/tty@' tests/run-test
chown -Rv nobody ./
su-tools nobody -s /bin/bash -c "make tests"
#
# Compilación e instalación del paquete
make install
# Ahora se lanza el programa bash recien compilado, sustituyendo al que se
# estaba ejecutando hasta ahora.
exec /bin/bash --login +h
#
#                          Instalación de Bzip2
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf bzip2.tar.gz
cd bzip2
patch -Np1 -i ../bzip2-1.0.4-install_docs-1.patch
#
# Compilación e instalación del paquete
make -f Makefile-libbz2_so
make clean
make
make PREFIX=/usr install
cp -v bzip2-shared /bin/bzip2
cp -av libbz2.so* /lib
ln -sv ../../lib/libbz2.so.1.0 /usr/lib/libbz2.so
rm -v /usr/bin/{bunzip2,bzcat,bzip2}
ln -sv bzip2 /bin/bunzip2
ln -sv bzip2 /bin/bzcat
#
#                           Instalación de Diffutils
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf diffutils.tar.gz
cd diffutils
patch -Np1 -i ../diffutils-2.8.1-i18n-1.patch
touch man/diff.1
```

```
#
# Configuración del paquete
./configure --prefix=/usr
#
# Compilación e instalación del paquete
make
make install
#
#                          Instalación de File
#
cd $LFS/sources
tar –xf file.tar.gz
cd file
./configure --prefix=/usr
make
make install
#
#                           Instalación de findutils
#
cd $LFS/sources
tar –xf findutils.tar.gz
cd findutils
./configure --prefix=/usr --libexecdir=/usr/lib/findutils \
 --localstatedir=/var/lib/locate
make
make install
mv -v /usr/bin/find /bin
sed -i -e 's/find:=${BINDIR}/find:=\/bin/' /usr/bin/updatedb
#
#                           Instalación de Flex
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf flex.tar.gz
cd flex
#
# Configuración del paquete
./configure --prefix=/usr
#
# Compilación e instalación del paquete
make
```

```
make install
ln -sv libfl.a /usr/lib/libl.a
# Para mantener la compatibilidad con el comando LEX, se crea el
# siguiente archivo
cat > /usr/bin/lex << "EOF"
#!/bin/sh
# Begin /usr/bin/lex
exec /usr/bin/flex -l "$@"
# End /usr/bin/lex
EOF
chmod -v 755 /usr/bin/lex
#
#                           Instalación de GRUB
#
cd $LFS/sources
tar –xf grub.tar.gz
cd grub
patch -Np1 -i ../grub-0.97-disk_geometry-1.patch
./configure --prefix=/usr
make
make install
mkdir -v /boot/grub
cp -v /usr/lib/grub/i386-pc/stage{1,2} /boot/grub
#
#                           Instalación de GAWK
#
cd $LFS/sources
tar –xf gawk.tar.gz
cd gawk
patch -Np1 -i ../gawk-3.1.5-segfault_fix-1.patch
#
# Configuración del paquete
./configure --prefix=/usr --libexecdir=/usr/lib
cat >> config.h << "EOF"
#define HAVE_LANGINFO_CODESET 1
#define HAVE_LC_MESSAGES 1
EOF
make
make install
#
#                           Instalación de GREP
```

```
cd $LFS/sources
tar –xf grep.tar.gz
cd grep
patch -Np1 -i ../grep-2.5.1a-redhat_fixes-2.patch
chmod +x tests/fmbtest.sh
#
# Configuración del paquete
./configure --prefix=/usr --bindir=/bin
make
make install
#
#                          Instalación de GROFF
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf groff.tar.gz
cd gruff
patch -Np1 -i ../groff-1.18.1.4-debian_fixes-1.patch
#
# Configuración del paquete
sed -i -e 's/2010/002D/' -e 's/2212/002D/' \
-e 's/2018/0060/' -e 's/2019/0027/' font/devutf8/R.proto
PAGE=<paper_size> ./configure --prefix=/usr --enable-multibyte
#
# Compilación e instalación del paquete
make
make install
ln -sv eqn /usr/bin/geqn
ln -sv tbl /usr/bin/gtbl

#
#                          Instalación de GZip
#
cd $LFS/sources
tar –xf gzip.tar.gz
cd gzip
#
# Configuración del paquete
./configure --prefix=/usr --bindir=/bin
make
make install
```

```
mv -v /bin/{gzexe,uncompress,zcmp,zdiff,zegrep} /usr/bin
mv -v /bin/{zfgrep,zforce,zgrep,zless,zmore,znew} /usr/bin
#
#                          Instalación de Inetutils
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf inetutils.tar.gz
cd inetutils
patch -Np1 -i ../inetutils-1.5-no_server_man_pages-2.patch
#
# Configuración del paquete
./configure --prefix=/usr --libexecdir=/usr/sbin \
--sysconfdir=/etc --localstatedir=/var \
--disable-logger
#
# Compilación e instalación del paquete
make
make install
mv -v /usr/bin/ping /bin
#
#                           Instalación de IPRoute
#
cd $LFS/sources
tar –xf iproute.tar.gz
cd iproute
sed -i -e '/tc-bfifo.8/d' -e '/tc-pfifo.8/s/pbfifo/bfifo/' Makefile
make SBINDIR=/sbin
make SBINDIR=/sbin install
mv -v /sbin/arpd /usr/sbin
#
#                           Instalación de Kbd
#
cd $LFS/sources
tar –xf kbd.tar.gz
cd kbd
patch -Np1 -i ../kbd-1.12-backspace-1.patch
patch -Np1 -i ../kbd-1.12-gcc4_fixes-1.patch
./configure --datadir=/lib/kbd
#
# Compilación e instalación del paquete
```

```
make
make install
mv -v /usr/bin/{kbd_mode,openvt,setfont} /bin
#
#                          Instalación de Less
#
cd $LFS/sources
tar –xf less.tar.gz
cd less
./configure --prefix=/usr --sysconfdir=/etc
make
make install
#
#                          Instalación de Make
#
cd $LFS/sources
tar –xf make.tar.gz
cd make
./configure --prefix=/usr
make
make install
#
#                         Instalación de Man-DB
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf man-db.tar.gz
cd man-db
mv man/de{_DE.88591,}
mv man/es{_ES.88591,}
mv man/it{_IT.88591,}
mv man/ja{_JP.eucJP,}
sed -i 's,\*_\*,??,' man/Makefile.in
sed -i -e '\%\t/usr/man%d' -e '\%\t/usr/local/man%d' src/man_db.conf.in
cat >> include/manconfig.h.in << "EOF"
#define WEB_BROWSER "exec /usr/bin/lynx"
#define COL "/usr/bin/col"
#define VGRIND "/usr/bin/vgrind"
#define GRAP "/usr/bin/grap"
EOF
patch -Np1 -i ../man-db-2.4.4-fixes-1.patch
```

```
#
# Configuración del paquete
./configure --prefix=/usr --enable-mb-groff --disable-setuid
#
# Compilación e instalación del paquete
make
make install
cat >> convert-mans << "EOF"
#!/bin/sh -e
FROM="$1"
TO="$2"
shift ; shift
while [ $# -gt 0 ]
do
FILE="$1"
shift
iconv -f "$FROM" -t "$TO" "$FILE" >.tmp.iconv
mv .tmp.iconv "$FILE"
done
EOF
install -m755 convert-mans /usr/bin
#
#                           Instalación de Mktemp
#
cd $LFS/sources
tar –xf mktemp.tar.gz
cd mktemp
patch -Np1 -i ../mktemp-1.5-add_tempfile-3.patch
./configure --prefix=/usr --with-libc
make
make install
make install-tempfile
#
#                          Instalación de Module-Init-Tools
#
cd $LFS/sources
tar –xf module-init-tools.tar.gz
cd module-init-tools
patch -Np1 -i ../module-init-tools-3.2.2-modprobe-1.patch
./configure
make check
```

```
make distclean
./configure --prefix=/ --enable-zlib
make
make INSTALL=install install
#
#                           Instalación de Patch
#
cd $LFS/sources
tar –xf patch.tar.gz
cd patch
./configure --prefix=/usr
make
make install
#
#                           Instalación de Psmisc
#
cd $LFS/sources
tar –xf psmisc.tar.gz
cd psmisc
./configure --prefix=/usr --exec-prefix=""
make
make install
mv -v /bin/pstree* /usr/bin
ln -sv killall /bin/pidof
#
#                           Instalación de Shadow
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf shadow.tar.gz
cd shadow
patch -Np1 -i ../shadow-4.0.18.1-useradd_fix-2.patch
#
# Configuración del paquete
./configure --libdir=/lib --sysconfdir=/etc --enable-shared \
--without-selinux
sed -i 's/groups$(EXEEXT) //' src/Makefile
find man -name Makefile -exec sed -i 's/groups\.1 / /' {} \;
sed -i -e 's/ ko//' -e 's/ zh_CN zh_TW//' man/Makefile
for i in de es fi fr id it pt_BR; do
    convert-mans UTF-8 ISO-8859-1 man/${i}/*.?
```

```
done
for i in cs hu pl; do
    convert-mans UTF-8 ISO-8859-2 man/${i}/*.?
done
convert-mans UTF-8 EUC-JP man/ja/*.?
convert-mans UTF-8 KOI8-R man/ru/*.?
convert-mans UTF-8 ISO-8859-9 man/tr/*.?
sed -i -e 's@#MD5_CRYPT_ENAB.no@MD5_CRYPT_ENAB yes@' \
   -e 's@/var/spool/mail@/var/mail@' etc/login.defs
#
# Compilación e instalación del paquete
make
make install
mv -v /usr/bin/passwd /bin
mv -v /lib/libshadow.*a /usr/lib
rm -v /lib/libshadow.so
ln -sfv ../../lib/libshadow.so.0 /usr/lib/libshadow.so
pwconv
grconv
useradd -D -b /home
sed -i 's/yes/no/' /etc/default/useradd
passwd root
#
#                          Instalación de Sysklogd
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf sysklogd.tar.gz
cd sysklogd
patch -Np1 -i ../sysklogd-1.4.1-fixes-2.patch
patch -Np1 -i ../sysklogd-1.4.1-8bit-1.patch
#
# Compilación e instalación del paquete
make
make install
cat > /etc/syslog.conf << "EOF"
# Begin /etc/syslog.conf
auth,authpriv.* -/var/log/auth.log
*.*;auth,authpriv.none -/var/log/sys.log
daemon.* -/var/log/daemon.log
kern.* -/var/log/kern.log
```

```
mail.* -/var/log/mail.log
user.* -/var/log/user.log
*.emerg *
# End /etc/syslog.conf
EOF
#
#                          Instalación de Sysvinit
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf sysvinit.tar.gz
cd sysvinit
sed -i 's@Sending processes@& configured via /etc/inittab@g' \
src/init.c
#
# Compilación e instalación del paquete
make -C src
make -C src install
#
# Se crea el archivo inittab
cat > /etc/inittab << "EOF"
# Begin /etc/inittab
id:3:initdefault:
si::sysinit:/etc/rc.d/init.d/rc sysinit
l0:0:wait:/etc/rc.d/init.d/rc 0
l1:S1:wait:/etc/rc.d/init.d/rc 1
l2:2:wait:/etc/rc.d/init.d/rc 2
l3:3:wait:/etc/rc.d/init.d/rc 3
l4:4:wait:/etc/rc.d/init.d/rc 4
l5:5:wait:/etc/rc.d/init.d/rc 5
l6:6:wait:/etc/rc.d/init.d/rc 6
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
su:S016:once:/sbin/sulogin
1:2345:respawn:/sbin/agetty tty1 9600
2:2345:respawn:/sbin/agetty tty2 9600
3:2345:respawn:/sbin/agetty tty3 9600
4:2345:respawn:/sbin/agetty tty4 9600
5:2345:respawn:/sbin/agetty tty5 9600
6:2345:respawn:/sbin/agetty tty6 9600
# End /etc/inittab
EOF
```

```
#
#                          Instalación de Tar
#
cd $LFS/sources
tar –xf tar-1.18.tar.gz
cd tar-1.18
./configure --prefix=/usr --bindir=/bin --libexecdir=/usr/sbin
make
make install
#
#                           Instalación de Texinfo
#
cd $LFS/sources
tar –xf Texinfo.tar.gz
cd Texinfo
# Se aplican los parches necesarios previos a la instalación
patch -Np1 -i ../texinfo-4.9-multibyte-1.patch
patch -Np1 -i ../texinfo-4.9-tempfile_fix-1.patch
# Configuracion del paquete
./configure --prefix=/usr
#
# Compilación e instalación del paquete
make
make install
make TEXMF=/usr/share/texmf install-tex
cd /usr/share/info
rm dir
for f in *
do install-info $f dir 2>/dev/null
done
#
#                           Instalación de Udev
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf udev.tar.gz
cd udev
tar -xvf ../udev-config-6.3.tar.bz2
#
# Compilación e instalación del paquete
install -dv /lib/{firmware,udev/devices/{pts,shm}}
```

```
mknod -m0666 /lib/udev/devices/null c 1 3
ln -sv /proc/self/fd /lib/udev/devices/fd
ln -sv /proc/self/fd/0 /lib/udev/devices/stdin
ln -sv /proc/self/fd/1 /lib/udev/devices/stdout
ln -sv /proc/self/fd/2 /lib/udev/devices/stderr
ln -sv /proc/kcore /lib/udev/devices/core
make EXTRAS="`echo extras/*/`"
make DESTDIR=/ EXTRAS="`echo extras/*/`" install
cp -v etc/udev/rules.d/[0-9]* /etc/udev/rules.d/
cd udev-config-6.3
make install
make install-doc
make install-extra-doc
cd ..
install -m644 -v docs/writing_udev_rules/index.html \
/usr/share/doc/udev-113/index.html
#
#                          Instalación de Util-linux
#
# Preparativos para la construccion
cd $LFS/sources
tar –xf util-linux.tar.gz
cd util-linux
sed -e 's@etc/adjtime@var/lib/hwclock/adjtime@g' \
-i $(grep -rl '/etc/adjtime' .)
mkdir -pv /var/lib/hwclock
patch -Np1 -i ../util-linux-2.12r-cramfs-1.patch
patch -Np1 -i ../util-linux-2.12r-lseek-1.patch
#
# Configuración del paquete
./configure
#
# Compilación e instalación del paquete
make HAVE_KILL=yes HAVE_SLN=yes
make HAVE_KILL=yes HAVE_SLN=yes install
#
#                          Instalación de Vim
#
cd $LFS/sources
tar –xf vim.tar.gz
cd vim
```

```
patch -Np1 -i ../vim-7.1-fixes-1.patch
patch -Np1 -i ../vim-7.1-mandir-1.patch
echo '#define SYS_VIMRC_FILE "/etc/vimrc"' >> src/feature.h
#
# Configuración del paquete
./configure --prefix=/usr --enable-multibyte
#
# Compilación e instalación del paquete
make install
ln -sv vim /usr/bin/vi
for L in "" fr it pl ru; do
ln -sv vim.1 /usr/share/man/$L/man1/vi.1
done
ln -sv ../vim/vim71/doc /usr/share/doc/vim-7.1
# Los cometarios para el archive vimrc se declaran con el símbolo "
cat > /etc/vimrc << "EOF"
" Begin /etc/vimrc
set nocompatible
set backspace=2
syntax on
if (&term == "iterm") || (&term == "putty")
set background=dark
endif
" End /etc/vimrc
EOF
#
#                   Eliminiación de los símbolos
#
logout
chroot $LFS /tools/bin/env -i \
HOME=/root TERM=$TERM PS1='\u:\w\$ ' \
PATH=/bin:/usr/bin:/sbin:/usr/sbin \
/tools/bin/bash –login
/tools/bin/find /{,usr/}{bin,lib,sbin} -type f \
-exec /tools/bin/strip --strip-debug '{}' ';'
```

5.2.5 ANEXO E – SCRIPT COMPRESSDOC

```
#!/bin/bash
# VERSION: 20080421.1623
#
# Compress (with bzip2 or gzip) all man pages in a hierarchy and
# update symlinks - By Marc Heerdink <marc @ koelkast.net>
#
# Modified to be able to gzip or bzip2 files as an option and to deal
# with all symlinks properly by Mark Hymers <markh @ linuxfromscratch.org>
#
# Modified 20030930 by Yann E. Morin <yann.morin.1998 @ anciens.enib.fr>
# to accept compression/decompression, to correctly handle hard-links,
# to allow for changing hard-links into soft- ones, to specify the
# compression level, to parse the man.conf for all occurrences of MANPATH,
# to allow for a backup, to allow to keep the newest version of a page.
#
# Modified 20040330 by Tushar Teredesai to replace $0 by the name of the
# script.
# (Note: It is assumed that the script is in the user's PATH)
#
# Modified 20050112 by Randy McMurchy to shorten line lengths and
# correct grammar errors.
#
# Modified 20060128 by Alexander E. Patrakov for compatibility with Man-DB.
#
# Modified 20060311 by Archaic to use Man-DB manpath utility which is a
# replacement for man --path from Man.
#
# Modified 20080421 by Dan Nicholson to properly execute the correct
# compressdoc when working recursively. This means the same compressdoc
# will be used whether a full path was given or it was resolved from PATH.
#
# Modified 20080421 by Dan Nicholson to be more robust with directories
# that don't exist or don't have sufficient permissions.
#
# Modified 20080421 by Lars Bamberger to (sort of) automatically choose
# a compression method based on the size of the manpage. A couple bug
# fixes were added by Dan Nicholson.
#
```

```
# Modified 20080421 by Dan Nicholson to suppress warnings from manpath
# since these are emitted when $MANPATH is set. Removed the TODO for
# using the $MANPATH variable since manpath(1) handles this already.
#
# TODO:
# - choose a default compress method to be based on the available
# tool : gzip or bzip2;
# - offer an option to restore a previous backup;
# - add other compression engines (compress, zip, etc?). Needed?
# Funny enough, this function prints some help.
function help ()
{
if [ -n "$1" ]; then
echo "Unknown option : $1"
fi
( echo "Usage: $MY_NAME <comp_method> [options] [dirs]" && \
cat << EOT
Where comp_method is one of :
--gzip, --gz, -g
--bzip2, --bz2, -b
Compress using gzip or bzip2.
--automatic
Compress using either gzip or bzip2, depending on the
size of the file to be compressed. Files larger than 5
kB are bzipped, files larger than 1 kB are gzipped and
files smaller than 1 kB are not compressed.
--decompress, -d
Decompress the man pages.
--backup Specify a .tar backup shall be done for all directories.
In case a backup already exists, it is saved as .tar.old
prior to making the new backup. If a .tar.old backup
exists, it is removed prior to saving the backup.
In backup mode, no other action is performed.
And where options are :
-1 to -9, --fast, --best
The compression level, as accepted by gzip and bzip2.
When not specified, uses the default compression level
for the given method (-6 for gzip, and -9 for bzip2).
Not used when in backup or decompress modes.
--force, -F Force (re-)compression, even if the previous one was
the same method. Useful when changing the compression
```

```
ratio. By default, a page will not be re-compressed if
it ends with the same suffix as the method adds
(.bz2 for bzip2, .gz for gzip).
--soft, -S Change hard-links into soft-links. Use with _caution_
as the first encountered file will be used as a
reference. Not used when in backup mode.
--hard, -H Change soft-links into hard-links. Not used when in
backup mode.
--conf=dir, --conf dir
Specify the location of man_db.conf. Defaults to /etc.
--verbose, -v Verbose mode, print the name of the directory being
processed. Double the flag to turn it even more verbose,
and to print the name of the file being processed.
--fake, -f Fakes it. Print the actual parameters compressdoc will use.
dirs A list of space-separated _absolute_ pathnames to the
man directories. When empty, and only then, use manpath
to parse ${MAN_CONF}/man_db.conf for all valid occurrences
of MANDATORY_MANPATH.
Note about compression:
There has been a discussion on blfs-support about compression ratios of
both gzip and bzip2 on man pages, taking into account the hosting fs,
the architecture, etc... On the overall, the conclusion was that gzip
was much more efficient on 'small' files, and bzip2 on 'big' files,
small and big being very dependent on the content of the files.
See the original post from Mickael A. Peters, titled
"Bootable Utility CD", dated 20030409.1816(+0200), and subsequent posts:
http://linuxfromscratch.org/pipermail/blfs-support/2003-April/038817.html
On my system (x86, ext3), man pages were 35564KB before compression.
gzip -9 compressed them down to 20372KB (57.28%), bzip2 -9 got down to
19812KB (55.71%). That is a 1.57% gain in space. YMMV.
What was not taken into consideration was the decompression speed. But
does it make sense to? You gain fast access with uncompressed man
pages, or you gain space at the expense of a slight overhead in time.
Well, my P4-2.5GHz does not even let me notice this... :-)
EOT
) | less
}
# This function checks that the man page is unique amongst bzip2'd,
# gzip'd and uncompressed versions.
# $1 the directory in which the file resides
# $2 the file name for the man page
```

```
# Returns 0 (true) if the file is the latest and must be taken care of,
# and 1 (false) if the file is not the latest (and has therefore been
# deleted).
function check_unique ()
{
# NB. When there are hard-links to this file, these are
# _not_ deleted. In fact, if there are hard-links, they
# all have the same date/time, thus making them ready
# for deletion later on.
# Build the list of all man pages with the same name
DIR=$1
BASENAME=`basename "${2}" .bz2`
BASENAME=`basename "${BASENAME}" .gz`
GZ_FILE="$BASENAME".gz
BZ_FILE="$BASENAME".bz2
# Look for, and keep, the most recent one
LATEST=`(cd "$DIR"; ls -1rt "${BASENAME}" "${GZ_FILE}" "${BZ_FILE}" \
2>/dev/null | tail -n 1)`
for i in "${BASENAME}" "${GZ_FILE}" "${BZ_FILE}"; do
[ "$LATEST" != "$i" ] && rm -f "$DIR"/"$i"
done
# In case the specified file was the latest, return 0
[ "$LATEST" = "$2" ] && return 0
# If the file was not the latest, return 1
return 1
}
# Name of the script
MY_NAME=`basename $0`
# OK, parse the command-line for arguments, and initialize to some
# sensible state, that is: don't change links state, parse
# /etc/man_db.conf, be most silent, search man_db.conf in /etc, and don't
# force (re-)compression.
COMP_METHOD=
COMP_SUF=
COMP_LVL=
FORCE_OPT=
LN_OPT=
MAN_DIR=
VERBOSE_LVL=0
BACKUP=no
FAKE=no
```

```
MAN_CONF=/etc
while [ -n "$1" ]; do
case $1 in
--gzip|--gz|-g)
COMP_SUF=.gz
COMP_METHOD=$1
shift
;;
--bzip2|--bz2|-b)
COMP_SUF=.bz2
COMP_METHOD=$1
shift
;;
--automatic)
COMP_SUF=TBD
COMP_METHOD=$1
shift
;;
--decompress|-d)
COMP_SUF=
COMP_LVL=
COMP_METHOD=$1
shift
;;
-[1-9]|--fast|--best)
COMP_LVL=$1
shift
;;
--force|-F)
FORCE_OPT=-F
shift
;;
--soft|-S)
LN_OPT=-S
shift
;;
--hard|-H)
LN_OPT=-H
shift
;;
--conf=*)
```

```
MAN_CONF=`echo $1 | cut -d '=' -f2-`
shift
;;
--conf)
MAN_CONF="$2"
shift 2
;;
--verbose|-v)
let VERBOSE_LVL++
shift
;;
--backup)
BACKUP=yes
shift
;;
--fake|-f)
FAKE=yes
shift
;;
--help|-h)
help
exit 0
;;
/*)
MAN_DIR="${MAN_DIR} ${1}"
shift
;;
-*)
help $1
exit 1
;;
*)
echo "\"$1\" is not an absolute path name"
exit 1
;;
esac
done
# Redirections
case $VERBOSE_LVL in
0)
# 0, be silent
```

```
DEST_FD0=/dev/null
DEST_FD1=/dev/null
VERBOSE_OPT=
;;
1)
# 1, be a bit verbose
DEST_FD0=/dev/stdout
DEST_FD1=/dev/null
VERBOSE_OPT=-v
;;
*)
# 2 and above, be most verbose
DEST_FD0=/dev/stdout
DEST_FD1=/dev/stdout
VERBOSE_OPT="-v -v"
;;
esac
# Note: on my machine, 'man --path' gives /usr/share/man twice, once
# with a trailing '/', once without.
if [ -z "$MAN_DIR" ]; then
MAN_DIR=`manpath -q -C "$MAN_CONF"/man_db.conf \
| sed 's/:/\\n/g' \
| while read foo; do dirname "$foo"/.; done \
| sort -u \
| while read bar; do echo -n "$bar "; done`
fi
# If no MANDATORY_MANPATH in ${MAN_CONF}/man_db.conf, abort as well
if [ -z "$MAN_DIR" ]; then
echo "No directory specified, and no directory found with \`manpath'"
exit 1
fi
# Check that the specified directories actually exist and are readable
for DIR in $MAN_DIR; do
if [ ! -d "$DIR" -o ! -r "$DIR" ]; then
echo "Directory '$DIR' does not exist or is not readable"
exit 1
fi
done
# Fake?
if [ "$FAKE" != "no" ]; then
echo "Actual parameters used:"
```

```
echo -n "Compression.......: "
case $COMP_METHOD in
--bzip2|--bz2|-b) echo -n "bzip2";;
--gzip|--gz|-g) echo -n "gzip";;
--automatic) echo -n "compressing";;
--decompress|-d) echo -n "decompressing";;
*) echo -n "unknown";;
esac
echo " ($COMP_METHOD)"
echo "Compression level.: $COMP_LVL"
echo "Compression suffix: $COMP_SUF"
echo -n "Force compression.: "
[ "foo$FORCE_OPT" = "foo-F" ] && echo "yes" || echo "no"
echo "man_db.conf is....: ${MAN_CONF}/man_db.conf"
echo -n "Hard-links........: "
[ "foo$LN_OPT" = "foo-S" ] &&
echo "convert to soft-links" || echo "leave as is"
echo -n "Soft-links........: "
[ "foo$LN_OPT" = "foo-H" ] &&
echo "convert to hard-links" || echo "leave as is"
echo "Backup............: $BACKUP"
echo "Faking (yes!).....: $FAKE"
echo "Directories.......: $MAN_DIR"
echo "Verbosity level...: $VERBOSE_LVL"
exit 0
fi
# If no method was specified, print help
if [ -z "${COMP_METHOD}" -a "${BACKUP}" = "no" ]; then
help
exit 1
fi
# In backup mode, do the backup solely
if [ "$BACKUP" = "yes" ]; then
for DIR in $MAN_DIR; do
cd "${DIR}/.."
if [ ! -w "`pwd`" ]; then
echo "Directory ``pwd`' is not writable"
exit 1
fi
DIR_NAME=`basename "${DIR}"`
echo "Backing up $DIR..." > $DEST_FD0
```

```
[ -f "${DIR_NAME}.tar.old" ] && rm -f "${DIR_NAME}.tar.old"
[ -f "${DIR_NAME}.tar" ] &&
mv "${DIR_NAME}.tar" "${DIR_NAME}.tar.old"
tar -cvf "${DIR_NAME}.tar" "${DIR_NAME}" > $DEST_FD1
done
exit 0
fi
# I know MAN_DIR has only absolute path names
# I need to take into account the localized man, so I'm going recursive
for DIR in $MAN_DIR; do
MEM_DIR=`pwd`
if [ ! -w "$DIR" ]; then
echo "Directory '$DIR' is not writable"
exit 1
fi
cd "$DIR"
for FILE in *; do
# Fixes the case were the directory is empty
if [ "foo$FILE" = "foo*" ]; then continue; fi
# Fixes the case when hard-links see their compression scheme change
# (from not compressed to compressed, or from bz2 to gz, or from gz
# to bz2)
# Also fixes the case when multiple version of the page are present,
# which are either compressed or not.
if [ ! -L "$FILE" -a ! -e "$FILE" ]; then continue; fi
# Do not compress whatis files
if [ "$FILE" = "whatis" ]; then continue; fi
if [ -d "$FILE" ]; then
# We are going recursive to that directory
echo "-> Entering ${DIR}/${FILE}..." > $DEST_FD0
# I need not pass --conf, as I specify the directory to work on
# But I need exit in case of error. We must change back to the
# original directory so $0 is resolved correctly.
(cd "$MEM_DIR" && eval "$0" ${COMP_METHOD} ${COMP_LVL} ${LN_OPT} \
${VERBOSE_OPT} ${FORCE_OPT} "${DIR}/${FILE}") || exit $?
echo "<- Leaving ${DIR}/${FILE}." > $DEST_FD1
else # !dir
if ! check_unique "$DIR" "$FILE"; then continue; fi
# With automatic compression, get the uncompressed file size of
# the file (dereferencing symlinks), and choose an appropriate
# compression method.
```

```
if [ "$COMP_METHOD" = "--automatic" ]; then
declare -i SIZE
case "$FILE" in
*.bz2)
SIZE=$(bzcat "$FILE" | wc -c) ;;
*.gz)
SIZE=$(zcat "$FILE" | wc -c) ;;
*)
SIZE=$(wc -c < "$FILE") ;;
esac
if (( $SIZE >= (5 * 2**10) )); then
COMP_SUF=.bz2
elif (( $SIZE >= (1 * 2**10) )); then
COMP_SUF=.gz
else
COMP_SUF=
fi
fi
# Check if the file is already compressed with the specified method
BASE_FILE=`basename "$FILE" .gz`
BASE_FILE=`basename "$BASE_FILE" .bz2`
if [ "${FILE}" = "${BASE_FILE}${COMP_SUF}" \
-a "foo${FORCE_OPT}" = "foo" ]; then continue; fi
# If we have a symlink
if [ -h "$FILE" ]; then
case "$FILE" in
*.bz2)
EXT=bz2 ;;
*.gz)
EXT=gz ;;
*)
EXT=none ;;
Esac
if [ ! "$EXT" = "none" ]; then
LINK=`ls -l "$FILE" | cut -d ">" -f2 \
| tr -d " " | sed s/\.$EXT$//`
NEWNAME=`echo "$FILE" | sed s/\.$EXT$//`
mv "$FILE" "$NEWNAME"
FILE="$NEWNAME"
else
LINK=`ls -l "$FILE" | cut -d ">" -f2 | tr -d " "`
```

```
fi
if [ "$LN_OPT" = "-H" ]; then
# Change this soft-link into a hard- one
rm -f "$FILE" && ln "${LINK}$COMP_SUF" "${FILE}$COMP_SUF"
chmod --reference "${LINK}$COMP_SUF" "${FILE}$COMP_SUF"
else
# Keep this soft-link a soft- one.
rm -f "$FILE" && ln -s "${LINK}$COMP_SUF" "${FILE}$COMP_SUF"
fi
echo "Relinked $FILE" > $DEST_FD1
# else if we have a plain file
elif [ -f "$FILE" ]; then
# Take care of hard-links: build the list of files hard-linked
# to the one we are {de,}compressing.
# NB. This is not optimum has the file will eventually be
# compressed as many times it has hard-links. But for now,
# that's the safe way.
inode=`ls -li "$FILE" | awk '{print $1}'`
HLINKS=`find . \! -name "$FILE" -inum $inode`
if [ -n "$HLINKS" ]; then
# We have hard-links! Remove them now.
for i in $HLINKS; do rm -f "$i"; done
fi
# Now take care of the file that has no hard-link
# We do decompress first to re-compress with the selected
# compression ratio later on...
case "$FILE" in
*.bz2)
bunzip2 $FILE
FILE=`basename "$FILE" .bz2`
;;
*.gz)
gunzip $FILE
FILE=`basename "$FILE" .gz`
;;
esac
# Compress the file with the given compression ratio, if needed
case $COMP_SUF in
*bz2)
bzip2 ${COMP_LVL} "$FILE" && chmod 644 "${FILE}${COMP_SUF}"
echo "Compressed $FILE" > $DEST_FD1
```

```
;;
*gz)
gzip ${COMP_LVL} "$FILE" && chmod 644 "${FILE}${COMP_SUF}"
echo "Compressed $FILE" > $DEST_FD1
;;
*)
echo "Uncompressed $FILE" > $DEST_FD1
;;
esac
# If the file had hard-links, recreate those (either hard or soft)
if [ -n "$HLINKS" ]; then
for i in $HLINKS; do
NEWFILE=`echo "$i" | sed s/\.gz$// | sed s/\.bz2$//`
if [ "$LN_OPT" = "-S" ]; then
# Make this hard-link a soft- one
ln -s "${FILE}$COMP_SUF" "${NEWFILE}$COMP_SUF"
else
# Keep the hard-link a hard- one
ln "${FILE}$COMP_SUF" "${NEWFILE}$COMP_SUF"
fi
# Really work only for hard-links. Harmless for soft-links
chmod 644 "${NEWFILE}$COMP_SUF"
done
fi
else
# There is a problem when we get neither a symlink nor a plain
# file. Obviously, we shall never ever come here... :-(
echo -n "Whaooo... \"${DIR}/${FILE}\" is neither a symlink "
echo "nor a plain file. Please check:"
ls -l "${DIR}/${FILE}"
exit 1
fi
fi
done # for FILE
done # for DIR
```

5.2.6 ANEXO F – SCRIPT PARA CONSTRUCCIÓN DE DATOS DEL LIVE CD

```
#!/bin/bash
#
# run this script to create a LiveCD in /tmp/live_data_1234
#
# Author: Tomas M. <http://www.linux-live.org>

export PATH=.:./tools:../tools:/usr/sbin:/usr/bin:/sbin:/bin:/

CHANGEDIR=$(dirname $(readlink -f $0))
echo "Changing current directory to $CHANGEDIR"
cd $CHANGEDIR

. liblinuxlive || exit 1
. ./.config || exit 1

# only root can continue, because only root can read all files from your system
allow_only_root

# live CD name
echo -ne "Name of your live distro [hit enter for $LIVECDNAME]: "
read NEWLIVECDNAME
if [ "$NEWLIVECDNAME" != "" ]; then LIVECDNAME=$NEWLIVECDNAME; fi
LIVECDNAME=$(echo $LIVECDNAME | tr -d ' ')

. ./install
if [ "$ROOT" -a "$ROOT" != "/" ]; then
   . ./install $ROOT
fi

# search for kernel
VMLINUZ=$ROOT/boot/vmlinuz
if [ -L "$VMLINUZ" ]; then VMLINUZ=$(readlink -f $VMLINUZ); fi
echo -ne "Enter path for the kernel you'd like to use [hit enter for $VMLINUZ]: "
read NEWKERNEL
if [ "$NEWKERNEL" != "" ]; then VMLINUZ="$NEWKERNEL"; fi
if [ "$(ls $VMLINUZ 2>>$DEBUG)" = "" ]; then echo "cannot find $VMLINUZ"; exit 1; fi

header "Creating LiveCD from your Linux"
```

```
echo "some debug information can be found in $DEBUG"

mkdir -p $CDDATA/$LIVECDNAME/{base,modules,optional,rootcopy,tools}

echo "copying cd-root to $CDDATA, using kernel from $VMLINUZ"
cp -R cd-root/boot $CDDATA
for i in isolinux syslinux; do
   cat cd-root/boot/$i/$i.cfg | sed -r "s/LABEL linux/LABEL $LIVECDNAME/" | sed -r
"s/Run linux/Run $LIVECDNAME/" > $CDDATA/boot/$i/$i.cfg
done
mv $CDDATA/boot/dos/linux.bat $CDDATA/boot/dos/${LIVECDNAME:0:8}.bat
cat cd-root/boot/dos/readme.txt | sed -r "s/LINUX.BAT/"${LIVECDNAME:0:8}.bat"/" >
$CDDATA/boot/dos/readme.txt

mkdir -p $CDDATA/$LIVECDNAME
cp -R cd-root/linux/* $CDDATA/$LIVECDNAME
cp tools/* $CDDATA/$LIVECDNAME/tools
cp -R DOC/LICENSE $CDDATA/$LIVECDNAME
cp $VMLINUZ $CDDATA/boot/vmlinuz

echo "creating initrd image..."
echo "Using kernel modules from $ROOT/$LMK"
cd initrd
./initrd_create $LIVECDNAME
if [ "$?" -ne 0 ]; then exit; fi
cd ..

cp initrd/initrd.gz $CDDATA/boot/initrd.gz
rm initrd/initrd.gz

echo "creating compressed images..."

for dir in $MKMOD; do
    if [ -d $ROOT/$dir ]; then
      echo "base/$dir.lzm ..."
      echo -ne > exclude.txt
      for i in $EXCLUDE; do
         part=$(echo "x/$i" | tr -s / | sed -r "s:x/[^/]+/::")
         if [ -e "$ROOT/$dir/$part" ]; then echo "$ROOT/$dir/$part" >> exclude.txt;
fi
      done
```

```
      cat exclude.txt
      create_module  $ROOT/$dir  $CDDATA/$LIVECDNAME/base/$dir.lzm  -keep-as-directory
-ef exclude.txt
      if [ $? -ne 0 ]; then exit; fi
      rm exclude.txt
      echo
    fi
done

cd $CDDATA/$LIVECDNAME
echo "--------done----------"
echo
echo "* run $CDDATA/$LIVECDNAME/make_iso.bat to create ISO image"
echo "* or copy content of $CDDATA to your USB device"
echo "and run ./boot/bootinst.sh (from the device!) to setup boot sector"
echo
echo "Now press Enter..."
read junk
```

5.2.7 ANEXO G – SCRIPT PARA LA CREACIÓN DEL ISO

```
#!/bin/bash
# ------------------------------------------------
# Script to create bootable ISO in Linux
# usage: make_iso.sh [ /tmp/slax.iso ]
# author: Tomas M. <http://www.linux-live.org>
# ------------------------------------------------

if [ "$1" = "--help" -o "$1" = "-h" ]; then
  echo "This script will create bootable ISO from files in curent directory."
  echo "Current directory must be writable."
  echo "example: $0 /mnt/hda5/slax.iso"
  exit
fi

CDLABEL="TESIS"
ISONAME=$(readlink -f "$1")

cd $(dirname $0)

if [ "$ISONAME" = "" ]; then
   SUGGEST=$(readlink -f ../../$(basename $(pwd)).iso)
   echo -ne "Target ISO file name [ Hit enter for $SUGGEST ]: "
   read ISONAME
   if [ "$ISONAME" = "" ]; then ISONAME="$SUGGEST"; fi
fi

mkisofs -o "$ISONAME" -v -J -R -D -A "$CDLABEL" -V "$CDLABEL" \
-no-emul-boot -boot-info-table -boot-load-size 4 \
-b boot/isolinux/isolinux.bin –C boot/isolinux/isolinux.boot ../.
```

5.2.8 ANEXO H – GUÍA DE USUARIO

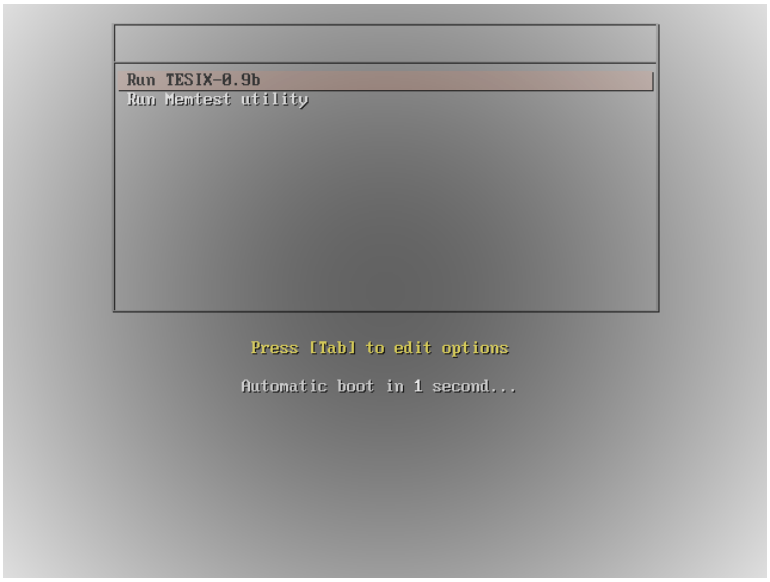
Para arrancar el sistema GNU/Linux llamado TESIX (ver. 0.9c) se deben tomar en cuenta las siguientes consideraciones:

- Disponer de una copia del CD del sistema GNU/Linux: TESIX 0.9c
- El computador en el cual se va a ejecutar el sistema debe estar configurado en el BIOS para que arranque desde la unidad óptica de CD/DVD como primera opción de arranque.
- La memoria RAM del computador debe ser de al menos 64MB.
- El teclado del computador debe ser del tipo Español/Lationamérica (ISO)

Los pasos a seguir para el arranque del sistema son:

Paso 1: Inserte el CD en la unidad lectora de CD/DVD.

Paso 2: Arranque el computador, con el CD dentro de la bandeja lectora.



Paso 3: Espere que aparezca la opción de *login* del sistema GNU/Linux, seguido a eso, ingrese los siguientes datos:

LOGIN: **root**
PASSWORD: (vacío) – Solo presione ***intro***

Paso 4: Luego del correcto ingreso al sistema, el usuario puede hacer uso del sistema GNU/Linux como cualquier distribución LINUX que funciona en modo de consola. (*sin interfaz gráfica*).

En caso de que la red no poseea el servicio DHCP, una vez iniciado el sistema, se debe asignar una dirección IP estática a la interfaz de red en el archivo: */etc/sysconfig/network/network-devices/ifconfig.eth#,* donde # corresponde al número de interfaz física en la cual se reconoció el hardware de red.

Para reiniciar o apagar el sistema sólo es necesario el uso de los siguientes comandos:

APAGADO: ***shutdown –h now***
REINICIO: ***shutdown –r now***