UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE DEFENSA PARA UNA RED CON HERRAMIENTAS DE SOFTWARE LIBRE

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

PRESENTA:

LISANDRO PABLO OLIVARES

DIRECTORA DE TESIS:

M.C. MARÍA JAQUELINA LÓPEZ BARRIENTOS

CIUDAD UNIVERSITARIA 2010

AGRADECIMIENTOS

A mis padres y hermanos

Porque gracias a su cariño y apoyo he llegado a realizar uno de mis más grandes anhelos, concluir mis estudios profesionales. Este es el legado más grande que pudiera recibir y por lo cual les viviré eternamente agradecido.
Con amor.

A mis amigos

Que constantemente estuvieron presentes para apoyarme durante las largas jornadas de trabajo y por esos momentos tan divertidos que hemos disfrutado juntos.

A mi directora de tesis, M.C. Ma. Jaquelina López Barrientos

Por el gran apoyo y profesionalismo para la realización del presente trabajo, muchas gracias por sus observaciones, comentarios y sugerencias.

ÍNDICE

INTRODUCCIÓN

El desarrollo y evolución de las comunicaciones en las últimas décadas han hecho más accesibles los sistemas informáticos, las redes de computadoras son ahora una herramienta necesaria para el desarrollo de las actividades de grandes empresas, gran parte del comercio se realiza mediante el uso de redes sofisticadas con grandes bases de datos, la información se genera de manera digital, al igual que su almacenamiento.

Las redes de computadoras comenzaron a implementarse a finales de los años 70 con el fin de compartir herramientas, recursos y datos para los usuarios. En aquél entonces la única preocupación que tenían las empresas era que alguien obtuviese acceso a una computadora o introdujera algún virus al sistema. El resguardo físico mediante personal o alarmas de seguridad era el único medio de defensa por el que tenían que preocuparse. Con la aparición de Internet y su enorme popularidad y crecimiento las cosas han cambiado, la Internet es una red de redes que proporciona múltiples y muy variados servicios y permite la comunicación a grandes distancias, el intercambio de datos, el comercio electrónico, entre otras actividades importantes. Sin embargo, así como Internet representa una serie de ventajas y servicios que una empresa puede aprovechar, también representa un riesgo ya que no se cuenta con una autoridad rectora que vigile el tipo de actividades se pueden realizar y que restringa los actos que puedan convertirse en una amenaza a las redes que se conectan a Internet.

Por otra parte, Internet no habría crecido de tal manera sin programas de libre distribución o *software libre*. Este concepto en el cual se pone a disposición de la comunidad programas y que a su vez permite la mejora de los mismos, debido a que también se proporciona el código fuente, ha permitido un enorme crecimiento de Internet.

Programas como SendMail y el popular servidor *Apache* Web, han servido de plataforma para la expansión y crecimiento de una red como Internet que sigue ofreciendo posibilidades ilimitadas gracias a programadores anónimos que desarrollan y mejoran los programas que se ejecutan en Internet.

Esta ventaja que nos ofrece el *software libre*, el gran crecimiento de programas de este tipo, y la facilidad con que accedemos a él, también significa un riesgo, pues cualquier persona puede tener acceso a programas y ejecutar ciertas tareas que constituyen amenazas a la seguridad de las redes. En Internet se pueden obtener de forma gratuita, programas que constituyen una amenaza para la seguridad de una red, entre estos, programas que dan acceso a la redes, hacen la negación de servicios, programas que tratan de obtener contraseñas de las cuentas de usuarios, programas que escalan privilegios y permiten el robo o la modificación de datos.

Por estas razones, la preocupación por mantener una red segura es mucho más grande ya que existen amenazas dentro y fuera de la red. Ahora un sistema puede ver vulnerada su seguridad por un ataque hecho desde miles de kilómetros de distancia por un individuo protegido por el anonimato y sin dejar rastro del ataque perpetuado. Aunque también existe la amenaza de un ataque a la seguridad perpetuado desde dentro del sistema, hecho por un administrador o por un usuario descuidado. El simple hecho de darle click a un archivo ejecutable puede vulnerar la seguridad de un sistema mal configurado.

Se ha convertido en práctica frecuente el ataque a empresas con redes de cualquier tamaño, no sólo las grandes redes de empresas. El objetivo ha cambiado y no sólo se centra en obtener información del sistema o datos importantes, sino que ahora utilizan los recursos de la red, como el ancho de banda, para perpetrar ataques desde nuestra red sin ser detectados. Las herramientas para hacer una intrusión en una red están disponibles en Internet y aprovechar cualquier vulnerabilidad de un sistema que por descuido o por desconocimiento del administrador, puedan ser explotadas.

Un ataque puede ser obra de un experto programador o de un principiante sin grandes conocimientos de programación pero con una herramienta poderosa, es por esto que no se debe correr ningún riesgo al momento de asegurar nuestro sistema.

A pesar de que Internet puede constituir una puerta de entrada a sujetos malintencionados a nuestra red, también es nuestra forma de comunicarnos hacia el exterior y brindar servicios a nuestros usuarios, y sin duda representa una mayor ventaja esto último, por lo que prescindir de una red conectada a Internet sólo porque puede significar una amenaza de seguridad es demasiado severo. Es mejor implementar medidas que nos permitan tener la suficiente certeza de que nuestro sistema está a salvo de dichas amenazas.

Tener una red de cualquier tipo conectada a Internet requiere de una arquitectura debidamente protegida por políticas de seguridad, una estrategia que nos permita en cada parte de nuestro sistema, controlar, monitorear y supervisar el funcionamiento adecuado de servicios, programas y herramientas. Para asegurar nuestra red es necesario conocer las características de nuestro sistema operativo, que es el primer elemento donde se instrumentan las medidas de seguridad. Además se requiere determinar aquellas herramientas que podemos implementar en nuestro sistema y los servicios que vamos a ofrecer a los usuarios, así como la información trascendental y los puntos débiles de nuestra red.

De esta manera, el administrador de una red deberá establecer criterios en áreas como las cuentas de usuario, el control de acceso, el control de acceso a la red, el cifrado de datos y el tipo de conexiones permitidas, registro, auditoría y control de red y finalmente la detección de intrusos.

Ciertas áreas se desarrollan desde dentro de la red y su control está enfocado a delimitar accesos y el uso de recursos por usuarios del sistema, como el correcto manejo de cuentas de usuarios y control de accesos. Otras áreas están enfocadas a la interacción entre nuestra red y redes externas, como el control de acceso a la red y el cifrado de datos. Entre otros, la implementación de un *firewall* entra en esta etapa.

Un *firewall* es un elemento fundamental para la protección del acceso a una red; se trata de una aplicación conectada entre Internet y la red a proteger, que tiene implementadas las directivas de seguridad, entre éstas, los servicios accesibles a nuestras terminales, los servicios que se ofrecen hacia fuera de la red, conexiones remotas y los programas que se van a ejecutar localmente. Estas decisiones se refieren principalmente al control de acceso y el uso autenticado de los servicios y programas en la red.

Un *firewall* es una herramienta importante y necesaria para filtrar y analizar paquetes de varios protocolos y realizar evaluaciones condicionales, si una petición se ajusta a una directiva predefinida, se realiza cierta acción, permitir el acceso o ejecución de cierto servicio o el bloqueo de protocolo y contenido. Un *firewall* bien implementado representa una puerta de control efectiva para la autenticación y correcto uso de sesiones en nuestra red, así como un obstáculo contra usuarios no deseados y software malicioso.

Un *firewall* puede mantener fuera de la red amenazas específicas, pero aún cuando están bien configurados, dejan algún tráfico de aplicación que puede ser peligroso. Si algún ataque logra entrar a través de él, se requiere de otro dispositivo de seguridad complementario que proteja al sistema desde el interior, que detecte a los intrusos que incursionan en la red. La detección de intrusiones es la práctica de utilizar herramientas inteligentes y automáticas para detectar intentos de intrusión, estas herramientas se llaman Sistemas de Detección de Intrusos.

Una vez que se traspasa la barrera del *firewall*, las herramientas implementadas para detectar intrusos se convierten en el último y más importante recurso para defender al sistema desde el interior. Estas herramientas permiten el análisis detallado del tráfico en la red y el comportamiento sospechoso como el escaneo de puertos. No sólo permiten el análisis del tipo de tráfico, sino que, además se revisa el contenido y su comportamiento. Un Sistema de Detección de Intrusos es una herramienta que funciona en forma complementaria con el *firewall*, uniendo dos capacidades distintas, la inteligencia de la detección y el poder de bloqueo del *firewall*.

Para realizar el monitoreo, se requiere de una base de datos actualizada que permita reconocer los ataques más recientes y se necesita un mantenimiento constante. Algunos sistemas emplean técnicas más avanzadas que pretender hacer que el mismo sistema aprenda de un ataque y responda en consecuencia. Pueden actuar de manera preventiva,

escuchando el tráfico en la red y tomando las medidas implementadas para las amenazas detectadas, o de manera reactiva, consultando sus bitácoras y respondiendo en consecuencia.

Los Sistemas de Detección de Intrusos pueden ser pasivos y activos; los pasivos únicamente detectan una posible intrusión, almacenan la información y mandan una señal de alerta al administrador de la red para que él se encargue de tomar acciones contra la intrusión. Por el contrario, los activos o reactivos, son aquellos que ante una amenaza generan algún tipo de respuesta como el cierre de la conexión o la reconfiguración del *firewall* para bloquear el tráfico que proviene de la red atacante.

Snort es un Sistema de Detección de Intrusos basado en red de libre distribución que puede funcionar tanto de manera pasiva como activa; implementa un motor de detección de ataques y barrido de puertos, esto permite registrar, alertar y responder ante cualquier anomalía previamente definida como patrones que corresponden a ataques, barridos, intentos aprovechar alguna vulnerabilidad, análisis de protocolos, entre otros.

Un Sistema de Detección de Intrusos es muy útil para protegernos contra ataques externos y además nos da la ventaja de localizar los ataques y actividades sospechosas de orígenes internos y responder ante ambas situaciones de manera casi inmediata. Aunque se requiere de un conocimiento más a fondo, este tipo de sistemas de detección, resultan fundamentales para cualquier tipo de redes que compartan servicios, recursos e información y tengan un punto de conexión a Internet.

Con base en esto, el objetivo del presente trabajo de tesis es *Crear Un Sistema De Defensa Con Herramientas De Software Libre* que pueda ser utilizado como esquema general básico, para todos aquellos entornos que requieran seguridad en sus redes de datos.

Asi, los objetivos particulares del proyecto son:

- Hacer uso de las herramientas que permitan conocer los puntos vulnerables de una red para controlar el acceso y proteger tanto los servicios como los recursos compartidos de la misma y sentar bases para el estudio de los sistemas de seguridad de las redes.

- Obtener el conocimiento necesario para el correcto uso de las herramientas de seguridad que permiten proteger una red.

- Verificar que el sistema defensa funcione correctamente en un entorno simulado, para ello, se considera que el sistema contendrá:

- Sistema operativo con seguridad reforzada
- Protección de contraseñas
- Escáner de vulnerabilidades
- Escáner de vulnerabilidades web
- Cortafuegos
- Sistema de detección de intrusos

Aunado a esto, considerando que la seguridad de la información es una necesidad de hoy día en todos los sistemas que generan, procesan, almacenan y transmiten información, y que los profesionales del cuidado de ésta deben estar preparados y contar con las habilidades prácticas para cuidar de ella es que adicionalmente se considera también como un objetivo *desarrollar una práctica sobre hardening* (reforzamiento de la seguridad) para el laboratorio de redes y seguridad de la Facultad de Ingeniería a fin de que los futuros Ingenieros en Computación adquieran los conocimientos básicos que les permitan reforzar la seguridad de los sistemas de información.

Para ello el documento se estructura en 6 capítulos. En el primero se describe la historia del *software libre* y de la libre distribución, además se analizan conceptos fundamentales relacionados con las redes de computadoras, los modelos de referencia *OSI* y TCP/IP, protocolos de red y los tipos de amenazas en una red.

El capítulo 2 contiene un análisis de los requerimientos de seguridad de la red que se va a proteger, se enumeran las posibles amenazas y los riesgos que afectan la red, el tipo de diseño que se puede implementar para una mejor protección de la red y los requerimientos de seguridad del servidor.

En el capítulo 3 se hace una descripción de las herramientas que se van a implementar como parte del sistema de defensa de la red, se detalla el tipo de herramienta a utilizar, el alcance de dicha herramienta, las opciones que se pueden implementar con la misma y la parte del sistema que protege.

Para las herramientas que se utilizan en el sistema de defensa se requiere hacer las instalaciones y configuraciones correspondientes, este proceso se describe paso a paso en el capítulo 4, además se realizan las instalaciones de las librerías y software adicional necesario para crear el sistema de defensa de la red y se realizan las configuraciones que permiten un correcto funcionamiento del sistema.

En el capítulo 5 se hace la prueba de los diferentes elementos del sistema de defensa, se detalla el tipo de prueba a realizar, los resultados esperados y las condiciones en que se va a llevar a cabo cada prueba. En este capítulo se obtienen los resultados de los elementos del sistema bajo condiciones simuladas.

El capítulo 6 es una práctica de reforzamiento de la seguridad con Bastille, una herramienta muy eficaz que permite, mediante una interfaz intuitiva, realizar el aseguramiento de los elementos importantes más vulnerables en un sistema operativo tipo Linux. Uno de los objetivos de esta práctica es que el estudiante comprenda la importancia del reforzamiento de la seguridad y que pueda realizarlo mediante el uso de esta herramienta.

Finalmente, en las conclusiones se definen los alcances logrados en este trabajo de acuerdo con los objetivos planteados inicialmente y se analizan los resultados obtenidos en forma general con el uso de herramientas de *software libre* implementadas como un sistema de defensa para una red.

Cabe mencionar que la parte práctica del presente trabajo, esto es, el desarrollo del sistema de defensa se recreó en el Laboratorio de Redes y Seguridad de la Facultad de Ingeniería de la UNAM, con la finalidad de poner en práctica las diferentes herramientas que permitan levantar el sistema de defensa y hacer las pruebas necesarias a fin de que los interesados en poner en práctica el sistema aquí sugerido, como base para resguardar redes de datos, cuenten con la información y consideraciones necesarias que les permitan implementar un sistema de defensa con herramientas de software libre.

CAPÍTULO 1

ANTECEDENTES

ANTECEDENTES

De acuerdo con el diccionario de la lengua española, un sistema se define como un "conjunto de cosas que relacionadas entre sí ordenadamente contribuyen a determinado objeto". Un sistema de defensa para una red, por lo tanto, es una serie de elementos relacionados entre sí de manera ordenada que tiene como objetivo proteger los activos de una red, y en este trabajo de tesis, los elementos son las herramientas de *software libre*.

En este capítulo se hace una recopilación de los conocimientos previos sobre las redes de computadoras que permiten comprender en lo subsecuente tanto términos como conceptos utilizados a lo largo de este trabajo.

1.1 DESARROLLO DEL *SOFTWARE LIBRE*

El *software libre* ha contribuido ampliamente en el desarrollo de diversos sistemas operativos y aplicaciones, así como en el desarrollo de Internet. Actualmente se crean sistemas muy avanzados y elaborados, bajo este concepto, sistemas operativos, librerías, juegos, bases de datos, procesadores de texto, y diversos programas especializados han sido desarrollados por un número cada vez más grande de comunidades de programadores de *software libre*.

Tanto los usuarios finales como los programadores contribuyen a mejorar el rendimiento de este tipo de programas, crean discusiones en foros, avisan sobre ciertos fallos, y mejoran el código fuente permanentemente, lo que ha conducido a la rápida expansión del movimiento de libre distribución. Algunos programas pueden ser adquiridos a cambio de una retribución económica, otros se adquieren de manera totalmente gratuita y el factor más importante es que se adquiere el código fuente y la libertad de usarlo y adaptarlo a nuestras propias necesidades. Esto tiene que ver con la manera en que surgió el movimiento de *software libre*, y con el concepto de libertad que se maneja en este movimiento.

1.1.1 Historia de la libre distribución

El movimiento del *software libre* o software de libre distribución tiene su origen en el nacimiento de Unix, razón por la que se asocia a la libre distribución con los sistemas Unix y Linux, aún cuando el concepto se ha extendido para casi todos los sistemas operativos de computadoras disponibles.

El sistema operativo Unix fue inventado por los laboratorios Bell, una división de investigación de AT&T a fines de los años sesenta. Poco tiempo después, AT&T permitió a las universidades utilizar su software. Como AT&T estaba regulado, no podía hacer negocios vendiendo Unix, así que ofreció a las universidades el código fuente para sus sistemas operativos. Éstas comenzaron a idear sus propias adiciones y modificaciones para

el código AT&T original. Algunas sólo realizaron cambios mínimos. Otras, como la universidad de Berkeley en California, realizaron tantas modificaciones que crearon un tipo totalmente nuevo del código. Pronto, el campo Unix se dividió en dos: el código AT&T, o mini, y el código base BSD, que generó muchas de las versiones Unix de libre distribución basadas en BSD actuales.

En un principio, los programadores que se desempeñaban en ámbitos empresariales y universitarios, creaban y compartían software sin ningún tipo de restricciones; esto cambió en la década de los 80, cuando las computadoras más modernas comenzaron a utilizar sistemas operativos privativos, forzando a los usuarios a aceptar condiciones restrictivas que impedían realizar modificaciones a dicho software, incluso los programadores tenían que firmar acuerdos de no revelación, con lo cual se impedía la colaboración entre desarrolladores de software.

En 1983, Richard Stallman inició el proyecto GNU (GNU is Not Unix), proyecto de desarrollo de *software libre*, como una forma de eliminar los obstáculos impuestos por los dueños del software privativo y de devolver el espíritu cooperativo que prevalecía en la comunidad computacional en sus inicios. De inmediato se pensó que esta nueva comunidad necesitaría una base sobre la cual pudiera trabajar, un sistema operativo de libre distribución, así fue como Stallman y los desarrolladores del proyecto GNU decidieron crear un sistema operativo libre, cuyo código fuente fuese accesible para cualquier persona, y además tuviera la libertad de usarlo con cualquier propósito, hacerle mejoras, adaptarlo a sus necesidades, copiarlo y distribuirlo.

El nuevo sistema operativo debía ser compatible con Unix porque dicho sistema ya estaba probado y era portable, así sería muy fácil para los usuarios cambiar de sistema operativo al nuevo GNU. El editor GNU Emacs fue una de las primeras tareas del nuevo proyecto GNU, para septiembre de 1985, GNU Emacs ya era usable y se distribuía mediante un pago. De esta manera se inició el negocio de distribución de *software libre*. Poco después, dio comienzo el desarrollo de un nuevo compilador, el GCC (GNU Compiler Collection). Ésta fue una de las herramientas más importantes del proyecto, GNU desarrolló un potente compilador en lenguaje C que siguen usando los sistemas operativos Linux hasta nuestros días.

Stallman creó en 1985 la Free Software Foundation, una organización libre de impuestos para el desarrollo de *software libre*. Esta organización se dedicó a implementar las utilidades que ofrecía Unix para el nuevo sistema operativo libre. A medida que el proyecto GNU crecía, los componentes del nuevo sistema operativo seguían en desarrollo, pero la meta de desarrollar por completo el sistema GNU todavía no se alcanzaba. Cada

componente de un sistema GNU se implementó en un sistema Unix, y para 1990 se tenía el sistema casi completo, el único componente importante que faltaba era el núcleo.

En 1991, Linus Torvalds, un estudiante universitario finlandés, hizo la aportación de la pieza que faltaba para tener un sistema operativo completo. Torvalds quería ejecutar una versión de Unix en su computadora, ya que éste era el sistema que se usaba en la universidad, para ello compró MINIX, una versión de computadora simplificada del sistema operativo Unix. Torvalds se sintió frustrado por las limitaciones de MINIX, particularmente en el área de la emulación de terminal y comenzó a trabajar para crear un programa emulación de terminal de computadora. Cuando terminó su programa comprendió que había creado un núcleo de sistema operativo y lo denominó Linux, pronto lo distribuyó a algunos grupos de noticias de USENET y los usuarios empezaron a sugerir mejoras y complementos. El nuevo núcleo fue adaptado al ambiente de GNU, y esto creó un amplio espectro de aplicaciones para el nuevo sistema operativo de libre distribución, resultado de la unión del software del proyecto GNU, múltiples programas de *software libre* y el núcleo Linux.

La designación Linux fue usada al principio únicamente para el núcleo, sin embargo, dicho núcleo era usado con frecuencia junto con otro software, especialmente del proyecto GNU. Richard Stallman fundador de GNU solicitó que el nombre GNU/Linux fuera usado para reconocer el rol del software GNU, sin embargo el nombre Linux es comúnmente utilizado para designar al sistema operativo completo.

La combinación de las comunicaciones globales de bajo costo y la facilidad del acceso a la información a través de las páginas web originaron un renacimiento en la innovación y el desarrollo del mundo de la libre distribución. Ahora, los programadores podían colaborar instantáneamente y colocar sus sitios web detallando su trabajo para que cualquiera pudiese encontrarlo utilizando motores de búsqueda. Los proyectos de trabajo en rutas paralelas combinaron sus recursos y esfuerzos, surgieron otros grupos a partir de los grupos más grandes, confiando en que ahora podían apoyar sus esfuerzos.

Actualmente existen muchos desarrolladores de *software libre* para distintas plataformas y este software se utiliza incluso junto con software privativo. Sistemas operativos, aplicaciones y librerías, se distribuyen de manera gratuita o a cambio de cierta remuneración en Internet. GNU/Linux es un gran ejemplo del desarrollo del *software libre*, sin embargo no es el único, infinidad de programas se realizan bajo este concepto y cada día se disputan un lugar en el ámbito computacional con software propietario.

1.1.2 La libertad del *software libre*

En su sitio web (http://www.gnu.org/philosophy/free-sw.es.html) el proyecto GNU define una serie de elementos que permiten conocer el concepto de libertad y sus alcances, cuando se habla de *software libre*[1].

El *software libre* es una cuestión de la libertad de los usuarios de ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. Más precisamente, significa que los usuarios de programas tienen las cuatro libertades esenciales.

- La libertad de ejecutar el programa, para cualquier propósito (libertad 0).
- La libertad de estudiar cómo trabaja el programa, y cambiarlo para que haga lo que usted quiera (libertad 1). El acceso al código fuente es una condición necesaria para ello.
- La libertad de redistribuir copias para que pueda ayudar al prójimo (libertad 2).
- La libertad de distribuir copias de sus versiones modificadas a terceros (la 3ª libertad). Si lo hace, puede dar a toda la comunidad una oportunidad de beneficiarse de sus cambios. El acceso al código fuente es una condición necesaria para ello.

Que un programa sea *software libre* no significa que no sea comercial. Un programa libre debe estar disponible para el uso comercial, la programación comercial y la distribución comercial.

El proyecto GNU surgió con la idea principal de dar libertad a los usuarios de *software libre* y que dicho software no se convirtiera en software privativo, dichas libertades se sintetizaron en una licencia especial, la licencia GPL (General Public License). Esta licencia es un conjunto específico de términos de distribución empleados para proteger un programa con copyleft. El Proyecto GNU utiliza esta licencia para la distribución de la mayoría del software de GNU.

El copyleft usa la ley de copyright, pero sirviendo para un propósito opuesto al usual, en lugar de ser un medio de privatizar el software, se transforma en un medio de mantener libre al software. La idea central es dar el permiso para correr el programa, modificarlo, copiarlo y redistribuir versiones modificadas, pero sin permiso para agregar restricciones propias. De esta manera las libertades básicas que definen al *software libre* quedan garantizadas para que cualquier persona tenga una copia y a su vez, estas libertades se convierten en derechos inalienables.

1.2 SISTEMAS OPERATIVOS DE LIBRE DISTRIBUCIÓN

El desarrollo del software ha dado lugar a que surjan nuevos sistemas operativos libres, con características diversas y licencias diferentes, unas más restrictivas que otras.

Existe una gran cantidad de sistemas operativos libres para diferentes arquitecturas, fundamentalmente se encuentran los sistemas operativos basados en BSD, en Linux y otras distribuciones libres, a continuación se describen las características más importantes de algunos de ellos.

1.2.1 Distribuciones basadas en BSD

OpenBSD es un sistema operativo libre tipo Unix, multiplataforma, descendiente de NetBSD, con especial atención en la seguridad y la criptografía. Su filosofía se resume en tres palabras, "libre, funcional y seguro". Se considera uno de los sistemas operativos más seguros y estables, aunque en su instalación por defecto se activen la menor cantidad de servicios posibles, esto también se considera como una práctica de seguridad. Utiliza un algoritmo de cifrado de contraseñas que hace muy difícil el procesamiento en paralelo y por lo tanto, también los intentos de descifrado. Debido a las características de este sistema operativo, se utiliza mucho en el sector de la seguridad informática como sistema operativo base para implementar *firewalls* y sistemas de detección de intrusos.

FreeBSD es un sistema operativo multitarea, multiusuario y multiproceso para procesadores de arquitectura Intel y compatibles con Intel, como AMD y Cyrix. También es posible utilizarlo hasta en otras arquitecturas como Alpha, AMD64, MIPS, PowerPC y UltraSPARC. Está hecho para ser compatible con la norma *Posix*, al igual que varios otros sistemas clones de Unix. El sistema incluye el núcleo, la estructura de archivos del sistema, bibliotecas de la API de C, y algunas utilerías básicas. FreeBSD es compatible con binarios de varios sistemas operativos del tipo Unix, incluyendo Linux, la razón de esto es la necesidad de ejecutar aplicaciones no libres desarrolladas para Linux, que por las ventajas que ofrecen, resulta conveniente portarlas a FreeBSD.

NetBSD es un sistema operativo tipo Unix, libre, disponible para más de 50 plataformas hardware, desarrollado a partir una gran variedad de software que incluye a 4.4BSD Lite de la Universidad de California-Berkeley, Net/2 (Berkeley Networking Release 2), el sistema X de ventanas y software GNU. Su principal ventaja es ofrecer un sistema operativo estable, multiplataforma, seguro y orientado a la investigación. Está diseñado teniendo como prioridad escribir código de calidad y bien organizado, y teniendo muy en cuenta el cumplimiento de estándares.

1.2.2 Distribuciones de GNU/LINUX

Debian GNU/Linux es un sistema operativo libre que se caracteriza por su portabilidad, su versión estable tiene soporte para 11 plataformas, una variedad muy amplia de software disponible y un grupo de herramientas que facilitan el proceso de instalación y actualización. Los sistemas Debian usan el núcleo de Linux. La mayoría de las herramientas básicas que completan el sistema operativo, provienen del proyecto GNU, por supuesto, herramientas de libre distribución. Además viene con más de 20,000 paquetes, es decir, software precompilado para una instalación sencilla en la máquina, todo de libre distribución.

Fedora es una distribución de Linux, libre, de propósitos generales y de código abierto. Destaca la implementación de una gran variedad de políticas de seguridad, como SELinux (Security Enhanced Linux), una colección de programas que modifican el núcleo fortaleciendo los mecanismos de control de acceso y forzando la ejecución de procesos dentro de un entorno con los mínimos privilegios necesarios. Incluye el control de acceso obligatorio (Mandatory Access Control), un elemento de seguridad que, a través de los módulos de seguridad de Linux que están en el kernel del sistema, evita el uso no autorizado de un recurso y el uso de un recurso de manera no autorizada.

Fedora soporta las arquitecturas x86, x86-64 y powerPC. Está diseñado de tal manera que se facilitan las tareas de instalación y configuración ya que incluye instaladores y herramientas gráficas. Sólo contiene una pequeña selección de paquetes de software, pero existen muchos almacenes disponibles para completar esta distribución.

Ubuntu es una distribución Linux que ofrece un sistema operativo enfocado a computadoras de escritorio aunque también proporciona soporte para servidores. Es una de las más importantes distribuciones de GNU/Linux a nivel mundial. Ubuntu es una distribución basada en Debian GNU/Linux y soporta las arquitecturas x86 y AMD64, aunque ha sido portada a otras cinco arquitecturas. Esta distribución ha sido traducida a numerosos idiomas y sus desarrolladores se basan en el trabajo de las comunidades de Debian, GNOME y KDE.

OpenSUSE es una distribución basada en Linux, auspiciada por Novell y AMD, completamente de código abierto, los desarrolladores de OpenSUSE han centrado sus esfuerzos en migrar ReiserFS a ext3 como sistema de archivos y en incluir soporte legal de MP3 y mejoras en los tiempos de carga. Novell continúa el desarrollo a puerta cerrada de dos distribuciones dedicadas al ámbito empresarial, SUSE Linux Enterpries Desktop y SUSE Linux Enterprise Server.

1.3 LAS REDES DE COMPUTADORAS

Una red de computadoras es un sistema de interconexión entre computadoras que permite compartir recursos e información[2]. En una red de computadoras se puede compartir información y recursos tanto de hardware como de software, lo que supone muchas ventajas para sus usuarios dependiendo de la finalidad con la que se haya diseñado la red.

A continuación se abordan diferentes clasificaciones de redes, de acuerdo con diferentes parámetros; también se tratan temas que tienen que ver con la manera en que se reconoce un equipo en una red, cómo se asignan las direcciones y que máscaras de subred se utilizan en las redes.

1.3.1 Diferentes clasificaciones de las redes de computadoras

Dependiendo cobertura geográfica:

- Si se conectan todas las computadoras dentro de un mismo edificio se denomina LAN (Local Area Network).
- Si se encuentran en edificios diferentes distribuidos en distancias no superiores al ámbito urbano, se denomina MAN (Metropolitan Area Network).
- Si están instaladas en edificios diferentes de la misma o distinta localidad, provincia o país, se denomina WAN (Wide Area Network).

Las redes de computadoras también se pueden clasificar de acuerdo con su topología. Se denomina topología a la forma geométrica en que están distribuidas las estaciones de trabajo y los cables que las conectan.

Las estaciones de trabajo de una red se comunican entre sí mediante una conexión física y el objeto de la topología es buscar la forma más económica y eficaz de conectarlas para, al mismo tiempo, facilitar la fiabilidad del sistema, evitar los tiempos de espera en la transmisión de los datos, mejorar el control de la red y permitir de forma eficiente el aumento de las estaciones de trabajo.

Las formas más utilizadas son:

Configuración en bus

En ella todas las estaciones comparten el mismo canal de comunicaciones, toda la información circula por ese canal y cada una de ellas recoge la información que le corresponde, véase la figura 1.1, Configuración en bus. Esta configuración es fácil de instalar, la cantidad de cable a utilizar es mínima, tiene una gran flexibilidad a la hora de

aumentar o disminuir el número de estaciones y el fallo de una estación no repercute en la red, aunque la ruptura de un cable la dejará totalmente inutilizada.

Entre sus inconvenientes destacan la facilidad de intervenir este tipo de redes, por usuarios de fuera de la red; su longitud no puede sobrepasar los 2000 metros. Además, el control del flujo se dificulta, ya que aunque varias estaciones intenten transmitir a la vez, como hay un único bus, sólo una de ellas podrá hacerlo, por lo que entre más estaciones tiene la red, es más complicado el control del flujo. Es la configuración más extendida actualmente y es usada por la red Ethernet.

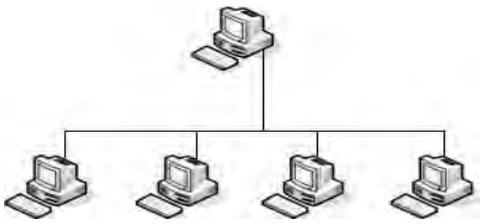


Figura 1.1, Configuración en bus

Configuración de anillo

En ella todas las estaciones están conectadas entre sí formando un anillo, de forma que cada estación sólo tiene contacto directo con otras dos, como se muestra en la figura 1.2, Configuración de anillo. En las primeras redes de este tipo los datos se movían en una única dirección, de manera que toda la información tenía que pasar por todas las estaciones hasta llegar a la estación destino. Las redes más modernas disponen de dos canales y transmiten en direcciones diferentes por cada uno de ellos. Este tipo de redes permite aumentar o disminuir el número de estaciones sin dificultad; pero, a medida que aumenta el flujo de información, será menor la velocidad de respuesta de la red.

Esta configuración tiene algunos inconvenientes, entre ellos, el que un fallo en una estación puede dejar bloqueada la red. También, un fallo en un canal de comunicaciones la dejará bloqueada en su totalidad y será bastante difícil localizar el fallo y repararlo de forma inmediata. Aunque su instalación es compleja, su uso está extendido por el entorno industrial.

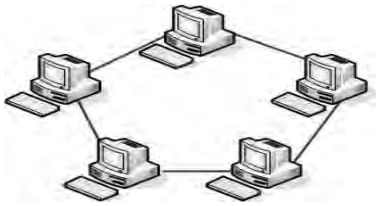


Figura 1.2, Configuración de anillo

Configuración en estrella

Esta forma de configuración es una de las más antiguas. Todas las estaciones están conectadas directamente al servidor y todas las comunicaciones se han de hacer necesariamente a través de él, como se muestra en la figura 1.3, Configuración en estrella. Permite incrementar y disminuir fácilmente el número de estaciones, si se produce un fallo en una de ellas no repercutirá en el funcionamiento general de la red; pero si se produce un fallo en el servidor, la red completa se vendrá abajo. Tiene un tiempo de respuesta rápido en las comunicaciones de las estaciones con el servidor, pero es más lento cuando se establece comunicación entre las distintas estaciones de trabajo.

La configuración en estrella no es muy conveniente para grandes instalaciones y su costo es caro debido a la gran cantidad de cableado y a la complejidad de la tecnología que se necesita para el servidor.

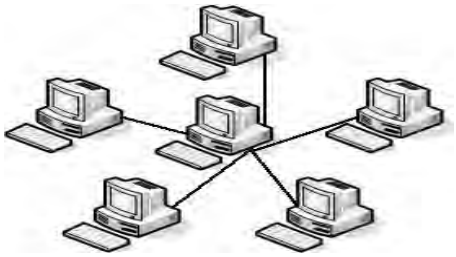


Figura 1.3, Configuración en estrella

Configuración híbrida

En esta configuración son diversas las formas en que se pueden combinar las topología mencionadas, entre las cuales se encuentra la configuración mixta en estrella/bus, en la cual un multiplexor de señal ocupa el lugar de la computadora central de la configuración en estrella, estando determinadas estaciones de trabajo conectadas a él, y otras conectas en bus junto con otro u otros multiplexores, una forma de configuración híbrida es la que se muestra a continuación en la figura 1.4, Configuración híbrida.

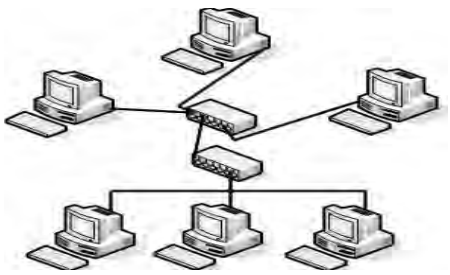


Figura 1.4, Configuración híbrida

1.3.2 Denominación de un equipo en TCP/IP

Debido a las múltiples ventajas que ofrece Internet, cada vez más redes de computadoras tienen una conexión a la red de redes. Esto ha hecho que Internet crezca de manera inusitada, y que existan millones de equipos activos conectados de manera temporal o permanente, equipos que tienen que ser identificados de alguna manera.

Es importante que se establezca la identificación del equipo que forma parte de una red, de una forma que evite su duplicidad dentro de todas las computadoras que puedan conectarse. Para ello, en TCP/IP se utiliza el nombre del usuario y el nombre del dominio de la red. Para identificar al usuario es necesario nombrarlo evitando que pueda haber dos con el mismo nombre y produzca confusiones al servidor de la red.

Para identificar a la red se utiliza el concepto de dominio. La estructura del dominio se asemeja a un árbol invertido, es decir, el tronco se encuentra en la parte superior y las ramas en la parte inferior, y cada hoja corresponde a un dominio. La identificación de un dominio está formada por varios apartados separados por un punto. Cada uno de ellos recibe el nombre de subdominio. El subdominio situado más a la derecha es el de carácter más general y recibe el nombre de dominio de nivel alto.

El nombre de un dominio completamente calificado (FQDN, Full Qualified Domain Name) comienza con el nombre de la estación de trabajo o host, un punto, y el nombre de la red (Dominio). Por ejemplo, si se denomina a la computadora como COMP001 y a la red principal como RED1, la identificación completa de la estación de trabajo sería COMP001. RED1. Si, a su vez, esta red formara parte de otra red superior, se volvería a poner otro punto y el nombre de dicha red, por ejemplo, COMP001.RED1.MEC; después del host vendría el o los subdominios y, para finalizar, el dominio.

También es interesante identificar a la institución de la que forma parte la red, así como la organización o el país a la que pertenece, en la tabla 1.1, Dominio de alto nivel de organización, se muestran las abreviaturas para los dominios de alto nivel de organización. Estos dos nuevos conceptos se añaden separados por puntos. Por ejemplo, para una red que pertenece a una institución educativa mexicana, se tendría la terminación .edu.mx.

Tabla 1.1 Dominio de alto nivel de organización

Dominio	Significado
com	Organización comercial
edu	Institución educativa
gov	Institución gubernamental
int	Organización internacional
mil	Organización militar
net	Organización de red
org	Organización sin ánimo de lucro
mx	Organización mexicana

1.3.3 Direcciones IPV4

Una dirección IP es un número binario de 32 bits utilizado para identificar excepcionalmetne al host y a su red[3].

Las direcciones *IP* consiguen que el envío de datos entre computadoras se realice de forma eficaz, de forma parecida a como se utilizan los número de teléfono en las llamadas telefónicas. La dirección del equipo indica el número que corresponde a la computadora dentro de la red.

Actualmente, las direcciones *IP* de la versión actual (Ipv4) tienen 32 bits, formados por cuatro campos de 8 bits cada uno, separados por puntos, por tanto, las direcciones *IP* están representadas en forma binaria. Cada uno de los campos de 8 bits puede tener un valor que esté comprendido entre 0 y 255 decimal. Normalmente y debido a la dificultad del sistema binario, la dirección *IP* se representa en decimal. Los cuatro octetos de la dirección *IP* componen una dirección de red y una dirección de equipo que están en función de la clase de red correspondiente.

Existen cinco clases de redes: A, B, C, D o E, esta diferencia está dada en función del número de computadoras que va a tener la red[4].

La clase A contiene 7 bits para la dirección de red, el primer bit del octeto siempre es un cero, y los 24 bits restantes representan a direcciones de equipo. De esta manera, se puede tener un máximo de 128 redes, aunque en realidad se tienen 126, ya que están reservadas las redes cuya dirección de red empieza por cero y por 127, cada una de las cuales puede tener 16,777,214 computadoras ya que se reservan aquellas direcciones de equipo en binario, cuyos valores sean todos ceros o todos unos. Las direcciones en representación decimal están comprendidas entre 0.0.0.0 y 127.255.255.255 y la máscara de subred es 255.0.0.0.

La clase B contiene 14 bits para direcciones de red, ya que el valor de los dos primero bits del primer octeto es siempre 10 y 16 bits para direcciones de equipo, lo que permite tener un máximo de 16,384 redes, cada una de las cuales puede tener 65,536 equipos, aunque en realidad tienen 65,534 cada una, ya que se reservan aquellas direcciones de equipo, en binario, cuyos valores sean todos ceros o todos unos. Las direcciones, en representación decimal están comprendidas entre 128.0.0.0 y 191.255.255.255 y su máscara de subred es 255.255.0.0.

La clase C contiene 21 bits para direcciones de red, ya que el valor de los tres primero bits del primer octeto ha de ser siempre 110 y 8 bits para direcciones de equipo, lo que le permite tener un máximo de 2,097,152 redes, cada una de las cuales puede tener 256 equipos, aunque en realidad tienen 254 equipos cada una, ya que se reservan aquellas direcciones de equipo, en binario, cuyos valores sean todos ceros o todos unos. Las direcciones, en representación decimal, están comprendidas entre 192.0.0.0 y 223.255.255.255. y su máscara de subred es 255.255.255.0.

La clase D se reserva todas las direcciones para multidestino o multicasting, es decir, una computadora transmite un mensaje a un grupo específico de computadoras de esta clase. El valor de los cuatro primeros bits del primer octeto ha de ser siempre 1110 y los últimos 28 bits representan los grupos multidestino. Las direcciones, en representación decimal, están comprendidas entre 224.0.0.0 y 239.255.255.255.

La clase E se utiliza con fines experimentales únicamente y no está disponible para el público. El valor de los cuatro primeros bits del primer octeto ha de ser siempre 1111 y las direcciones, en representación decimal, están comprendidas entre 240.0.0.0 y 255.255.255.255.

1.4 SERVICIOS DE RED

El objetivo de una red es que los usuarios de la misma puedan compartir recursos e información, y así mejorar el rendimiento global de la organización. El uso de la red y los servicios que esta ofrece proporcionan una serie de ventajas, como la facilidad en la comunicación, una mejora en la competitividad, reducción de costos, mejoras en la administración, mejoras en integridad de datos y seguridad de la información.

Las redes con una arquitectura cliente-servidor permiten la distribución de las tareas entre programas que se ejecutan en el servidor o en la terminal del usuario, el agente que requiere que se haga determinada tarea se llama cliente y el agente que realiza dicha tarea se llama servidor. Una arquitectura de este tipo permite mejorar el rendimiento, reduce los costos, al compartir recursos, y facilita la administración, al concentrarse los trabajos en los servidores.

Existen múltiples recursos que pueden ser compartidos en una red para beneficio de sus usuarios. Algunos recursos proporcionan servicios que se identifican como servicios de hardware y otros como servicios de software, por ejemplo, el uso de un servidor de impresión se identifica como un servicio de hardware, mientras que el uso de un programa como el servidor *apache*, se identifica como un servicio de software. A continuación se describen algunos servicios de software que se ofrecen en la mayoría de las redes.

Uno de los servicios básicos que ofrecen todas las redes de computadoras es el control de acceso, que comprende tanto los elementos de verificación de la identidad de los usuarios como los servicios y herramientas necesarios para delimitar el uso de los recursos por los usuarios. Este servicio implementa mecanismos que permiten establecer permisos para la ejecución de tareas dentro de la red, acceso a directorios y archivos y el uso adecuado de los recursos de la red.

Otro servicio básico es el almacenamiento. Ofrecer una cierta capacidad de almacenamiento significa implementar los mecanismos de seguridad e integridad en la información y configurar el sistema de tal manera que el usuario pueda disponer de sus datos en cualquier momento. Este servicio permite disminuir las capacidades de almacenamiento de las terminales del usuario.

La ejecución de programas especializados y el uso de recursos de gran capacidad son algunos de los servicios más importantes para los usuarios de una red y representan una de las principales ventajas de implementar una red. Si todos los usuarios de la red pueden hacer uso de la capacidad de cálculo de los equipos de la red y ejecutar programas especializados que requieren de procesadores potentes, no necesitan hacer instalaciones de gran capacidad en sus terminales ni requieren la instalación de software especializado para

desempeñar su trabajo, de esta manera se facilita el trabajo en la organización y se reducen costos al utilizar la red como una entidad única.

Un aspecto básico para el uso de una red es el acceso a la información; de acuerdo esta premisa básica, una red que ofrece acceso a una base de datos se convierte en un instrumento vital para el desarrollo de las actividades de toda organización. Los servidores de bases de datos constituyen una herramienta fundamental para el almacenamiento de datos de los usuarios, que a diferencia del almacenamiento en directorios, una base de datos ofrece información estructurada, organizada y disponible para el uso en otras aplicaciones.

El correo electrónico es otro servicio que permite mejorar la comunicación entre los usuarios de la organización a un costo menor que el uso de teléfonos. Asimismo, se pueden ofrecer otros servicios que varían dependiendo de las necesidades de las organizaciones, servidores de imágenes, de impresión y videoconferencias son sólo algunos ejemplos de ellos.

Una red de computadoras también puede funcionar como una puerta para acceder a todos los servicios que brinda Internet, entre los que destacan, el correo electrónico, las videoconferencias, el chat, las descargas de programas, los buscadores, compras, noticias y un sinfín de servicios informáticos.

1.5 MODELOS *OSI* Y TCP/IP

Para poder establecer una comunicación entre computadoras, lo mismo que para establecerla entre personas, es necesario contar con una serie de normas que regulen dicho proceso, a este conjunto de normas se le denomina protocolo y es lo que hace posible el intercambio fiable de comunicación entre dos equipos informáticos. Esas normas son definidas por organismos internacionales de normalización.

Al principio del desarrollo de la computación cada fabricante establecía los procedimientos de comunicación entre sus computadoras de forma independiente, por lo que resultaba muy difícil la comunicación entre computadoras de fabricantes distintos. Poco a poco se fue haciendo necesario disponer de unas normas comunes que permitiesen la intercomunicación entre todas las computadoras.

1.5.1 Modelo *OSI*

De todos los protocolos propuestos destaca el modelo *OSI* (Open Systems Interconnection), Interconexión de Sistemas Abiertos, que fue propuesto por la Organización Internacional de Normalización (ISO).

ISO es una organización no gubernamental fundada en 1947, tiene por misión la coordinación del desarrollo y aprobación de estándares a nivel internacional. Su ámbito de trabajo cubre todas las áreas, incluyendo las redes locales, a excepción de las áreas electrotécnicas que son coordinadas por IEC (International Electrotechnical Commission).

Cada país únicamente puede estar representado en ISO por una organización y en el caso de Estados Unidos está representada por ANSI (American National Standards Institute).

El modelo *OSI*, cuya actividad se empezó a desarrollar en 1977 y llegó a constituirse como estándar internacional en 1983, trata de establecer las bases para la definición de protocolos de comunicación entre sistemas informáticos. Este modelo propone dividir en niveles todas las tareas que se llevan a cabo en una comunicación entre computadoras, todos los niveles están bien definidos y no interfieren con los demás.

En total se tienen siete niveles, como se aprecia en la figura 1.5, Modelo *OSI*, los cuatro primero tienen funciones de comunicación y los tres restantes de proceso. Cada uno de los siete niveles dispone de los protocolos específicos para el control de dicho nivel[2].

Nivel físico

En este nivel se definen las características eléctricas y mecánicas de la red necesarias para establecer y mantener la conexión física, se incluyen las dimensiones físicas de los conectores, los cables y los tipos de señales que van a circular por ellos. Los sistemas de redes locales más habituales definidos en este nivel son: Ethernet, red en anillo con paso de testigo (*Token ring*) e interfaz de datos distribuidos por fibra (*FDDI*, Fiber Distributed Data Interface).

Nivel de enlace de datos

Se encarga de establecer y mantener el flujo de datos que discurre entre los usuarios. Controla si se van a producir errores y los corrige (se incluye el formato de los bloques de datos, los códigos de dirección, el orden de los datos transmitidos, la detección y la recuperación de errores). Las normas Ethernet y *Token ring* también están definidas en este nivel.

Nivel de red

Se encarga de decidir por dónde se han de transmitir los datos dentro de la red, incluye la administración y gestión de los datos, la emisión de mensajes y la regulación del tráfico de la red. Entre los protocolos más utilizados definidos en este nivel se encuentran: Protocolo Internet (*IP*) y el intercambio de paquetes entre redes (IPX, Internetwork Packet Exchange) de Novell.

Nivel de transporte

Asegura la transferencia de la información a pesar de los fallos que pudieran ocurrir en los niveles anteriores, incluye la detección de bloqueos, caídas del sistema, asegurar la igualdad entre la velocidad de transmisión y la velocidad de recepción y la búsqueda de rutas alternativas. Entre los protocolos de este nivel más utilizados se encuentran el Protocolo de Control de Transmisión (*TCP*, Transmission Control Protocol) de Internet y el intercambio secuencial de paquetes (SPX, Sequenced Packet Exchange) de Novell.

Nivel de sesión

Organiza las funciones que permiten que dos usuarios se comuniquen a través de la red. En este nivel se consideran las tareas de seguridad, contraseñas de usuarios y la administración del sistema.

Nivel de presentación

Traduce la información del formato de la máquina aun formato comprensible por los usuarios, se incluye el control de las impresoras, emulación de terminal y los sistemas de codificación.

Nivel de aplicación

Se encarga del intercambio de información entre los usuarios y el sistema operativo, se incluye la transferencia de archivos y los programas de aplicación como telnet, ftp, messenger, correo electrónico, etc.

Figura 1.5, Modelo OSI

El proceso que se produce desde que un usuario envía un mensaje hasta que llega a su destino consiste en una bajada a través de todos los niveles, con sus correspondientes protocolos, desde el nivel séptimo hasta llegar al primero. Allí se encontrará en el canal de

datos que le dirigirá al usuario destino y volverá a subir por todos los niveles hasta llegar al último de ellos, la figura 1.6, Niveles del Modelo *OSI*, muestra este proceso.

Los niveles inferiores proporcionan servicios a los niveles superiores; cada nivel dispone de un conjunto de servicios, que están definidos mediante protocolos. Los programadores y diseñadores de productos sólo deben preocuparse por los protocolos del nivel en el que trabajan, los servicios proporcionados a los niveles superiores y los servicios proporcionados por los niveles inferiores.

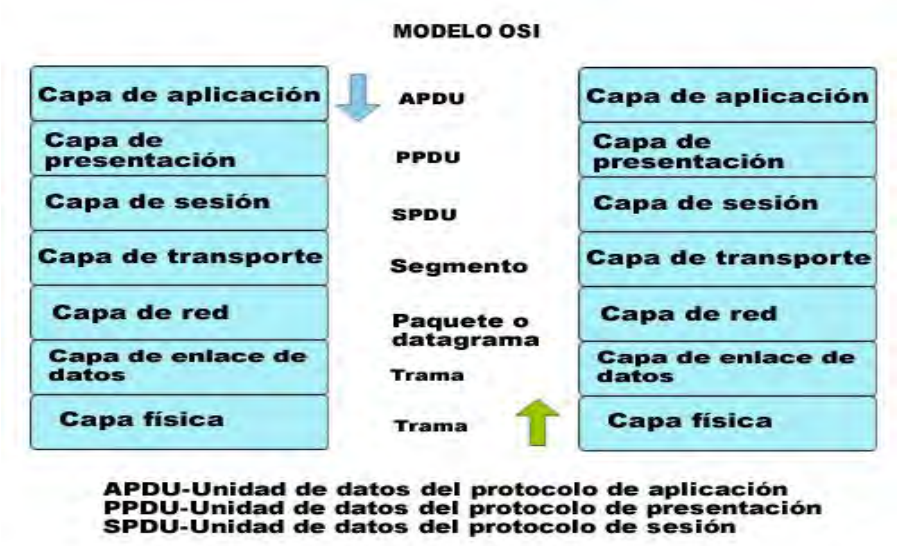


Figura 1.6, Niveles del Modelo OSI

1.5.2 Modelo TCP/IP

TCP/IP fue desarrollado mucho antes de que el modelo OSI de siete capas propuesto por ISO fuera especificado y encaja en su propio modelo de cuatro capas. Todos los protocolos del conjunto de protocolos TCP/IP se ubican en las tres capas superiores de este modelo.

El protocolo TCP/IP es en realidad un conjunto de protocolos aceptados por la industria que permiten la comunicación en un entorno muy variado[3]. Permite acceder a Internet y a sus recursos y se ha convertido en un estándar en la interconexión de redes y en la interoperabilidad entre distintos tipos de equipos. Fue desarrollado por el departamento de la defensa de Estados Unidos con el objetivo de mantener enlaces de comunicación entre sitios en el caso de una guerra mundial; utiliza una arquitectura escalable, cliente-servidor adaptable a las necesidades de conexión.

Este modelo consta de cuatro niveles, como se muestra en la figura 1.7, Modelo TCP/IP, el más bajo es la capa de interfaz de red, que especifica los detalles físicos relativos a la

forma de transmisión de los datos por una red, en este nivel se define el medio eléctrico de transmisión de los datos entre dispositivos de hardware como cable coaxial, fibra óptica o cable par trenzado. En esta capa se implementan los estándares Ethernet, *Token ring*, *FDDI*, X.25, *Frame relay* y otros similares.

La capa de Internet empaqueta los datos en datagramas *IP*, que contienen información de las direcciones de origen y destino utilizada para reenviar los datagramas entre hosts y a través de redes, además, realiza el enrutamiento de los datagramas *IP*. Entre los protocolos incluidos están *IP*, *ICMP*, *IGMP* y *ARP*.

La capa de transporte permite administrar las sesiones de comunicación entre equipos host. Define el nivel de servicio y el estado de la conexión utilizada al transportar datos. Entre los protocolos incluidos están *UDP* y *TCP*.

La capa de aplicación define los protocolos de aplicación TCP/IP y cómo se conectan los programas de host a los servicios del nivel de transporte para utilizar la red. La capa de aplicación contiene protocolos y servicios de aplicación tales como HTTP, Telnet, FTP, DNS, SMTP y X Windows.

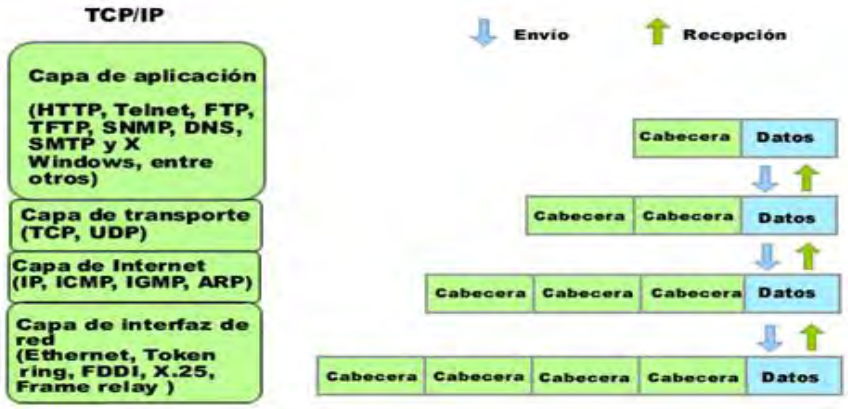


Figura 1.7, Modelo TCP/IP

De igual manera que en el modelo *OSI*, el proceso que se produce desde que un usuario envía un mensaje hasta que llega a su destino consiste en una bajada a través de todas las capas del modelo TCP/IP, con sus correspondientes protocolos, desde la capa de aplicación hasta la capa de interfaz de red. Allí se encontrará en el canal de datos que le dirigirá al usuario destino y volverá a subir por todas las capas hasta llegar nuevamente a la capa de aplicación.

Se puede establecer una correspondencia entre los modelos *OSI* y TCP/IP como se muestra en la figura 1.8, Modelos *OSI* y TCP/IP.

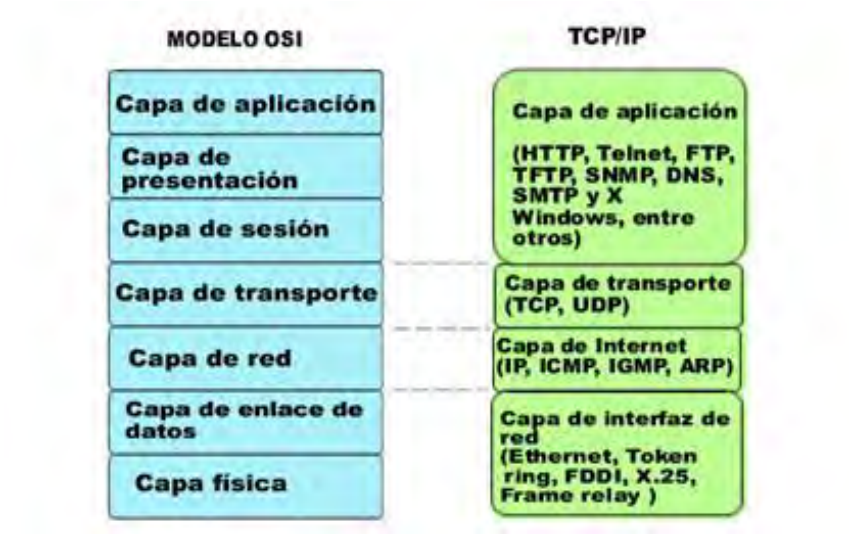


Figura 1.8, Modelos OSI y TCP/IP

1.6 PROTOCOLOS DE RED

Un protocolo es una especificación estándar para dar formato a los datos y transferirlos de una manera que tanto los equipos emisores como los receptores sigan las mismas reglas y así asegurar la comunicación y entendimiento entre ambos.

A continuación se analizan los protocolos básicos, cabe mencionar que esta enumeración no es exhaustiva y por lo tanto no se considera la totalidad de protocolos y subprotocolos existentes.

1.6.1 Protocolo Internet

El protocolo Internet (*IP*, Internet Protocol) se encarga de la clasificación y entrega de paquetes o datagramas en la capa de red del modelo *OSI* y su equivalente *IP* en el modelo TCP/IP. Cada paquete entrante o saliente, o datagrama *IP*, incluye la dirección *IP* origen del remitente y la dirección *IP* del destinatario deseado; las direcciones *IP* de un datagrama no se modifican durante la transmisión de un paquete por la red. *IP* es el responsable del enrutamiento de los paquetes; los datagramas pasan a *IP* desde el *UDP* y *TCP* y desde los adaptadores, *IP* examina la dirección de destino de cada datagrama, la compara con una tabla de enrutamiento y decide qué acción adoptar. Las rutas y las tablas de enrutamiento se pueden configurar de forma estática o dinámica mediante el protocolo de enrutamiento de Internet o abrir la ruta de acceso más corta primero (OSPF, Open Shortest Path First).

El protocolo *IP* es de los denominados “no fiable”, ya que, en busca de una mayor eficiencia, no realiza comprobaciones de una correcta recepción por parte del equipo destino. Para el control de los posibles errores, existe el protocolo *ICMP* que genera los

avisos pertinentes, además la fiabilidad de la comunicación la proporcionan protocolos superiores como *TCP*. En la figura 1.9, Cabecera *IP*, se muestran los campos que conforman una cabecera *IP*.

Figura 1.9, Cabecera *IP*.

Versión (version) (4 bits): el campo versión indica el formato de la cabecera *IP*, en la actualidad sólo se manejan dos:

- IPV4: IP estándar
- IPV6: La versión 6 del protocolo.

Longitud (Internet header length, IHL) (4 bits): indica la longitud de la cabecera expresada en palabras de 32 bits, siendo la longitud mínima 5 palabras, que equivaldría a un paquete *IP* con una cabecera sin opciones.

Tipo de servicio (type of service, TOS) (8 bits): el tipo de servicio se emplea para determinar parámetros como la fiabilidad, la procedencia, el retardo y la capacidad de salida asociadas a este paquete.

Longitud total del datagrama (total length) (16 bits): especifica la longitud total del datagrama *IP* expresada en bytes, incluyendo la cabecera y todos los encapsulados de ésta.

Identificación (identification) (16 bits): este campo identifica de manera unívoca cada paquete enviado por el emisor, de esta forma, se hace posible la reconstrucción, por parte del receptor, de paquetes grandes que tuvieron que ser fragmentados en algún punto de la red.

Banderas (flags) (3 bits):

- Flag more flag (MF): es el primer bit y se usa en la fragmentación, indica si es el último paquete o si le siguen más.

- Don´t fragment (DF): es el segundo bit y especifica si el emisor permite una fragmentación del datagrama.
- El tercer bit en la actualidad no se usa.

Desplazamiento del fragmento (fragment offset) (13 bits): este fragmento indica la posición (desplazamiento) del paquete dentro del paquete original, lo que hace posible su reconstrucción en el destino.

Tiempo de vida (time to live, TTL) (8 bits): indica el tiempo máximo que el paquete permanecerá en la red; si el valor llega a 0, el paquete será destruido, este valor realmente indica el número de saltos (router) por los que puede llegar a pasar.

Protocolo (protocol) (8 bits): especifica el protocolo del siguiente nivel que recibirá los datos, el contenido del campo comenzará, por ejemplo, por *TCP* o *UDP*.

Suma de comprobación (header checksum) (16 bits): este campo se usa par comprobar la integridad de la cabecera, esto es debido a que, durante la transmisión, ciertos campos de la cabecera van modificando su tamaño en cada salto; por ejemplo, se ha de recalcular este valor y verificarse en cada punto intermedio para garantizar una transmisión correcta.

Dirección *IP* de origen (source address) (32 bits): indica la dirección *IP* del dispositivo que originó la transmisión.

Dirección *IP* Destino (destination address) (32 bits): indica la dirección *IP* del dispositivo de destino.

Opciones (options): de longitud variable, puede tener 0 o más opciones, guarda las opciones solicitadas por el emisor, generalmente de seguridad, source routing, timestamps, etc.

Relleno (padding): de longitud variable, su tarea es asegurar que la cabecera *IP* acaba en múltiplo de 32 bits.

Datos (data): de longitud variable, contiene los datos a enviar, siendo su longitud múltiplo de 8 bits y el tamaño máximo 65,535 bytes (64 Kbytes). El campo comenzará con el contenido de la cabecera del protocolo del siguiente nivel, *TCP* o *UDP*.

Para mayor información consulte el RFC 791 - http://www.faqs.org/rfcs/rfc791.html

1.6.2 Protocolo *TCP*

El protocolo de control de transmisión (*TCP*, Transport Control Protocol) proporciona un servicio seguro, basado en conexión de flujo de bytes a las aplicaciones. Se utiliza para inicios de sesiones, para compartir archivos e impresoras, para procesos de duplicación entre controladores de dominio, para la transferencia de listas exploradas y otras funciones comunes. Se puede utilizar únicamente para comunicaciones uno a uno en la que el punto de partida y los extremos están definidos y la ruta de transmisión se establece mediante un protocolo de enlace.

El protocolo *TCP* es empleado por la mayoría de los servicios dentro de la red, es un protocolo de los considerados fiables, desde el punto de vista de que asegura una correcta recepción de los paquetes, ya que utiliza secuencias y confirmación de recepciones mediante el uso de unos paquetes especiales denominados ACK (ACKnowledgement). Es un protocolo, en consecuencia orientado a la conexión. Esto quiere decir que ambos interlocutores han de conocer sus direcciones *IP* para establecer la conexión y emitir y recibir por un puerto especificado en el paquete y conocido por ambos, emisor y receptor, de igual modo que la conexión, se ha de finalizar de una manera correcta. El protocolo *TCP* establece una conexión de tipo stream, es decir, de flujos de información de 8 bits (1 byte) por lo que la información viaja sin marcadores de datos. En la siguiente figura, 1.10, Protocolo *TCP*, se muestran los campos de esta cabecera.

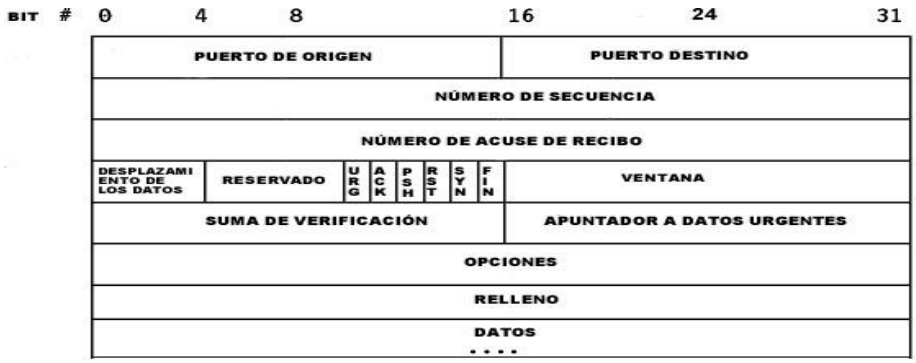


Figura 1.10, Protocolo TCP.

Puerto de origen (source port) (16 bits): puerto de origen o aplicación en el sistema origen.

Puerto destino (destination port) (16 bits): puerto de destino o aplicación en el sistema destino.

Número de secuencia (sequence number) (32 bits): indica el número de secuencia del primer byte de datos del segmento *TCP*. Cuando el flag SYN está activo (establecimiento

de conexión) éste se considera como parte de los datos y se usa el número inicial de la secuencia.

Número de acuse de recibo (acknowledgement number) (32 bits): si el flag ACK está activo, este campo contiene el valor del siguiente número de secuencia que el sistema espera recibir. Este campo siempre tendrá un valor una vez que la conexión se haya establecido.

Desplazamiento de los datos (data offset) (4 bits): número de palabras de 32 bits en la cabecera *TCP*, de tal manera que queda totalmente claro dónde finaliza la cabecera y comienzan los datos.

Reservado (reserved) (6 bits): está reservado para un uso futuro.

Banderas (flags) (6 bits): determinan el funcionamiento de la conexión *TCP*.

URG: flag de urgencia, especifica al receptor la existencia de información urgente dentro del flujo de datos.

ACK: reconoce una recepción correcta de datos, indicando que el campo "acknowledgement number" es válido.

PSH: la función push indica al receptor que ha de pasar los datos a la capa superior (aplicación) tan rápido como sea posible.

RST: indica un reinicio en la conexión.

SYN: paquete que se usa durante el inicio de la sesión para sincronizar los número de secuencia.

FIN: indica que no existen más datos y comienza la negociación de fin de comunicación.

Ventana (window) (16 bits): la ventana *TCP* se utiliza para el control del flujo de datos. Contiene el número de bytes de datos que pueden ser recibidos, empezando por el que se indica en el campo acknowledgement.

Suma de verificación (checksum) (16 bits): es un campo para el control de la integridad de la cabecera *TCP*. En este caso el valor 0 no es un valor correcto.

Apuntador a datos urgentes (urgent pointer) (16 bits): el objetivo de este campo es indicar el puntero con el desplazamiento donde comienzan los datos urgentes, con la finalidad de que el receptor pueda adquirirlos directamente. Este campo sólo tiene sentido cuando el flag URG está activo.

Opciones (options): de longitud variable, puede tener 0 o más opciones, lo más típico dentro de *TCP* es MSS (Maximum Segment Size) permitiendo al sistema especificar el tamaño máximo del segmento que espera recibir, suele indicarse en el paquete con flag SYN, durante el establecimiento de la comunicación.

Relleno (padding): de longitud variable, tiene como misión hacer que la cabecera tenga un tamaño múltiplo de 32, añadiendo ceros para conseguirlo.

Datos (data): este campo almacena los datos y, evidentemente, es de longitud variable.

Para mayor detalle RFC 793 Consulte: http://www.faqs.org/rfcs/rfc793.html

1.6.3 Protocolo de mensajes de control de Internet

El protocolo de mensajes de control de Internet (*ICMP*, Internet Control Messaging Protocol) proporciona funciones de mantenimiento y enrutamiento para el Protocolo de Internet. Los mensajes *ICMP* se encapsulan en datagramas *IP* y pueden enrutarse entre redes. El protocolo genera y mantiene tablas de enrutamiento, realiza labores de descubrimiento de enrutadores, ayuda a determinar la unidad máxima de transmisión de ruta (PMTU, Path Maximum Transmission Unit), ajusta el flujo de control para evitar la saturación de enlaces o enrutamiento, y proporciona herramientas de diagnóstico, como ping.

ICMP se emplea para el control de errores, es decir, para la notificación de errores o para ciertas situaciones que requieran determinada atención. El protocolo se encuentra definido en el RFC 792.

Los paquetes *ICMP* viajan dentro de paquetes *IP* y este protocolo es, en ocasiones, considerado de un nivel superior. Existen diversos tipos de mensajes para la notificación de errores, así como peticiones de información. Cuando se realiza un ping entre dos máquinas se están enviando paquetes empleando este protocolo.

Un paquete *ICMP* contiene 8 primeros bits del paquete *IP* que lo generó, así el sistema receptor del paquete será capaz de extraerlo de la red y sabrá asociarlo a *TCP* o a *UDP*. En la figura 1.1, Protocolo *ICMP*, se muestra el contenido de este tipo de paquete.

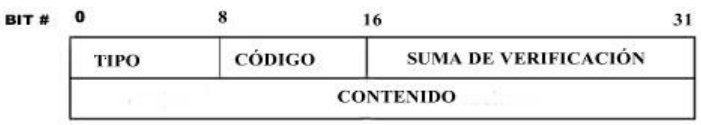


Figura 1.11, Protocolo *ICMP*.

Tipo (type) (8 bits): identifica el tipo específico de mensaje *ICMP*, puede tener 15 posibles valores.

Código (code) (8 bits): para un mismo tipo de mensaje, en este campo se especifican las diferentes condiciones.

Suma de verificación (checksum) (16 bits): es un campo de comprobación de integridad para el total del mensaje *ICMP*, este campo es de carácter obligatorio.

Contenido (contents): de longitud variable, depende del tipo de mensaje.

Si se analizan los mensajes *ICMP* de error, se puede apreciar que uno de los más típicos es el de tipo 3, el que no se puede llegar al destino (destination unreachable).

Además este protocolo se usa para la obtención de información de la red mediante la utilización de pares de paquetes (petición/respuesta); es curioso también su empleo para la obtención de la fecha y hora mediante el manejo de timestamps.

Además el uso más típico es para conectividad, como ya se mencionó antes, utilizando el conocido ping.

1.6.4 Protocolo de resolución de direcciones

El protocolo de resolución de direcciones (*ARP*, Address Resolution Protocol) resuelve direcciones *IP* para las direcciones de control de acceso a medios (MAC) utilizadas por los dispositivos hardware de red, como adaptadores. Una dirección MAC identifica un dispositivo dentro de su propia red física mediante un número de 6 bytes (48 bits) programados en la memoria de sólo lectura. Las direcciones MAC suelen visualizarse en notación hexadecimal.

El funcionamiento de este protocolo está descrito en el RFC 826. El protocolo *ARP* tiene como misión realizar la asociación entre la dirección *IP* y un dispositivo físico, es decir, la dirección física (MAC). Ejemplo: Dirección *IP* 192.168.0.51, Dirección MAC 00-10-60-5b-13-9c.

En la figura 1.12, Protocolo *ARP*, se muestran los campos contenidos en esta cabecera.

Figura 1.12, Protocolo *ARP*.

Descripción de los campos de ARP

Protocolo de hardware (hardware protocol) (16 bits): tecnología de red empleada por debajo de TCP/IP (Ethernet).

Protocolo de red (network protocol) (16 bits): tipo de protocolo empleado a nivel 3 (nivel *IP*).

Longitud de dirección de red de hardware (hardware address length) (8 bits): longitud de la dirección de red de hardware.

Longitud de la dirección de red (network address length) (8 bits): longitud de la dirección de red *IP*.

Operación (operation) (16 bits): tipo de operación que nos da información sobre si se trata de una petición o una respuesta *ARP*.

Dirección física de la interfaz del emisor (sender hardware address) (48 bits): dirección física (MAC) de la interfaz de red del emisor.

Dirección *IP* del emisor (sender network address) (32 bits): dirección *IP* del emisor.

Dirección física de la interfaz del receptor (target hardware address) (48 bits): dirección física (MAC) de la interfaz de red del receptor.

Dirección *IP* del receptor (target network address) (32 bits): la dirección *IP* del receptor.

La conexión se realiza a nivel físico, pero existe algún punto en la red donde tiene que haber una asociación entre esas direcciones físicas, es decir, las MAC de los dispositivos y las direcciones *IP*. Los encargados de efectuar este tipo de relaciones serán los routers y los switches que son capaces de guardar información a ese nivel. Si bien dichos dispositivos memorizan la tabla de correspondencias entre la dirección *IP* de una máquina y su

dirección MAC, para conseguir el encapsulamiento entre las capas, existe un protocolo que en esencia tiene una misión, éste protocolo es el *ARP* o protocolo de resolución de nombre.

Cuando un equipo intenta establecer una conexión con una dirección *IP* determinada, necesita determinar cuál es la MAC del dispositivo de red al que le corresponde la dirección *IP* que está solicitando; mediante la dirección *IP*, los paquetes van saltando de router en router hasta que llegan por fin a la red donde se encuentra dicha dirección *IP* y, una vez allí, es el router el que determina la asociación con la MAC y se establece la conexión.

El router tiene memorizadas esas correspondencias en una tabla, la cual no es infinita, y las relaciones se van generando dinámicamente según su uso, por ello las relaciones antiguas serán eliminadas de la tabla dando paso a las nuevas. La manera por la cual el router conoce estas asociaciones es mediante peticiones *ARP*, en las cuales pregunta quién tiene una determinada MAC y el equipo que coincida responderá dando su dirección *IP* y esta será almacenada en las tablas.

La técnica de memorizar esa tabla de resolución aporta velocidad a la red y una reducción drástica del número de paquetes *ARP* que estarían circulando por la red, optimizando el tráfico de éstos, ya que sólo se realiza una petición *ARP* cuando su correspondencia no está dentro de la tabla.

Los routers son piezas fundamentales en toda red, por tanto, su seguridad es considerada como vital para el funcionamiento de la red.

1.6.5 Protocolo de datagrama de usuarios

El protocolo de datagrama de usuarios (*UDP*, User Datagram Protocol) proporciona un servicio de transporte sin conexión, no confiable y utilizado, por lo general, para comunicaciones de uno a muchos, que emplean datagramas *IP* de difusión y multidifusión. Dado que la entrega de los datagramas *UDP* no es segura, las aplicaciones que utilizan *UDP* pueden suministrar sus propios mecanismos de fiabilidad. *UDP* se puede usar para proceso de inicio de sesión, exploración y resolución de nombres.

El protocolo *UDP* es muy simple, no fiable y no orientado a la conexión, por lo que los protocolos de niveles superiores son los encargados de asegurar una transmisión correcta de datos. No orientado a la conexión significa que no espera a recibir confirmación positiva del destinatario de que la transmisión está siendo recibida de manera correcta, se limita a localizar al destinatario y a entregar. Cada operación de envío genera un único paquete de datagrama *UDP*. Se usa en aplicaciones multimedia, para el envío de flujos de información

sin un costo de conexión asociado. La figura 1.13, Protocolo *UDP*, nos muestra el contenido de este datagrama.

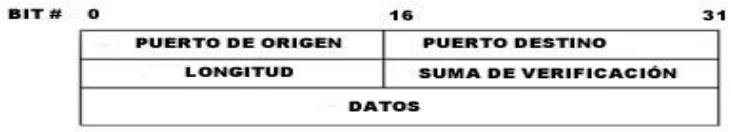


Figura 1.13, Protocolo UDP.

Puerto de origen (source port) (16 bits): puerto de origen o aplicación en el sistema de origen.

Puerto destino (destination port) (16 bits): puerto de destino o aplicación en el sistema destino.

Longitud (length) (16 bits): longitud en bytes del datagrama, incluyendo la cabecera *UDP* y los datos.

Suma de verificación (checksum) (16 bits): comprobación de la integridad de la cabecera *UDP*; este campo puede tomar el valor 0, lo que quiere decir que no se han de realizar comprobaciones de integridad.

Datos: contenido de longitud variable.

Para mayor detalle RFC 793 Consulte: http://www.faqs.org/rfcs/rfc768.html

1.7 NORMAS DE SEGURIDAD

Debido a la necesidad de contar con un estándar de medición de la seguridad en los sistemas informáticos, se han establecido una serie de normas de seguridad o criterios con los que se puede evaluar el grado de confianza que se puede depositar en un sistema. Ejemplos de estas normas de seguridad son los TCSEC (Trusted Computer System Evaluation Criteria o Libro Naranja) del Departamento de Defensa de los Estados Unidos, los criterios de la evaluación de seguridad de la tecnología de información (ITSEC), un estándar europeo, y los criterios comunes para evaluación de seguridad de tecnología de la información (Common Criteria for Information Technology Security - CCIT-SE)[5].

Criterios comunes para evaluación de seguridad de tecnología de la información

Los criterios comunes son una norma internacional para evaluar la seguridad de los productos de tecnología de la información basados en los criterios europeos,

norteamericanos y canadienses existentes para la evaluación de la seguridad de tecnología de la información[6].

Los criterios comunes se originaron con proyectos cooperativos que estaban relacionados con la Organización Internacional de Estándares (ISO). En 1999 fue adoptada por ISO como Tecnología de información –Técnicas de seguridad- Criterios de evaluación para la seguridad de tecnología de la información (ISO/IEC 15408).

La certificación de un producto sobre los criterios comunes se debe hacer por una entidad independiente aprobada. El proceso de evaluación certifica que un producto verifica los aspectos siguientes:

- Los requerimientos del producto están definidos correctamente.
- Los requerimientos están implementados correctamente.
- El proceso de desarrollo y documentación del producto cumple con ciertos requerimientos.

Los Criterios Comunes están formados por tres partes: introducción y modelo general, requerimientos de seguridad funcional y requerimientos de garantía de seguridad.

La introducción y modelo general describen el formato de los criterios comunes y como fueron establecidos, la segunda parte, requerimientos de seguridad funcional define once clases denominadas clases de requerimientos funcionales y cada una de ellas cubre un área particular de la seguridad, a su vez, cada clase cuenta con una o más familias funcionales que tratan aspectos específicos de la totalidad de la clase en cuestión. Las clases antes mencionadas son:

- Auditoría de seguridad
- Comunicación
- Soporte de cifrado
- Protección de datos de usuario
- Identificación y autenticación
- Administración de la seguridad
- Privacía
- Protección de las funciones de seguridad del objeto de evaluación
- Utilización de recursos
- Control de acceso
- Caminos/canales confiables

La tercera parte de los criterios define a las clases de garantía de seguridad y los niveles de garantía. Las clases de garantía son aquellos requerimientos necesarios para garantizar la seguridad que brinda el objeto de evaluación. Estos requerimientos están agrupados en siete

clases y a su vez, cada clase está formada por una o más familias de garantías. Las siete clases de garantía de la seguridad son:

- Administración de la configuración
- Distribución y operación
- Desarrollo
- Documentos guía
- Soporte al ciclo de vida
- Pruebas
- Vulnerabilidad de los bienes

Los niveles de garantía o EAL (Evaluation Assurance Level) definen una escala para garantizar la medición y los criterios para la evaluación de perfiles de protección. Los niveles de garantía son siete van desde el nivel de seguridad más bajo (EAL_1) hasta el más alto (EAL_7):

EAL_1 Funcionalidad probada.
EAL_2 Estructuralmente probado
EAL_3 Probado y verificado metódicamente
EAL_4 Diseñado, probado y revisado metódicamente
EAL_5 Diseñado y probado semiformalmente
EAL_6 Diseño verificado y probado semiformalmente
EAL_7 Diseño verificado y probado formalmente

Los criterios comunes proporcionan ventajas a los usuarios de productos de tecnología de la información como son, tener un medio de comparación para varios productos de diferentes desarrolladores; contar con un esquema que permita describir sus necesidades de seguridad y utilizar productos evaluados por una entidad externa de acuerdo con criterios internacionales reconocidos.

En el caso de los desarrolladores, los criterios comunes brindan un medio para comprobar que su producto es adecuado, está bien diseñado y es seguro, y esto constituye las ventajas competitivas que busca todo desarrollador para posicionar su producto en el mercado.

1.8 TIPOS DE AMENAZAS EN UNA RED

Una amenaza se representa a través de una persona, una circunstancia o evento, un fenómeno o una idea maliciosa las cuales pueden provocar daño cuando existe una violación de la seguridad, así, una amenaza es todo aquello que intenta o pretende destruir[6].

En la actualidad las redes de computadoras están expuestas a múltiples amenazas que atentan contra la seguridad de la información y de los recursos de la red. Existen amenazas

relacionadas con el hardware, con el software, catástrofes naturales, fallas humanas o ataques malintencionados. Las amenazas por catástrofes naturales son inherentes a toda organización, sin embargo se puede disminuir el riesgo por este tipo de amenazas mediante una adecuada planeación de las instalaciones y una correcta protección de los equipos.

Las amenazas humanas son, generalmente debidas a errores u omisiones, o al desconocimiento de las personas que utilizan los recursos de la red. Existe otro tipo de amenaza que se diferencia del resto, básicamente porque no se aprovecha de debilidades y vulnerabilidades propias de un componente informático para la obtención de información y es, la amenaza del tipo ingeniería social, que consiste en utilizar artilugios, tretas y otras técnicas que engañan a las personas para que de manera no intencional revelen información de interés para el atacante. También se encuentra dentro de este tipo de amenazas, el fraude, el robo, el sabotaje y el espionaje.

Las amenazas debidas al hardware son, esencialmente debidas a errores de fabricación y al mal uso de los equipos por parte del personal encargado. Las amenazas al software por otra parte, consideran elementos como la mala configuración de los programas, fallas en las aplicaciones, códigos maliciosos, virus y exploits que pueden aprovechar las vulnerabilidades en el software.

Las amenazas a la red pueden ser debidas a la mala configuración de la misma, la implementación de un sistema operativo inadecuado o con una mala administración, controles de acceso inadecuados o deficientes. Tambien se puede hacer una clasificación de amenazas dependiendo de la zona a la que pueda dirigirse un ataque. De esta manera se tendrían tres tipos de amenazas a valorar: amenazas de red, amenazas al servidor y amenazas de aplicaciones, aunque algo muy importante a tomar en cuenta es que estas tres amenazas están íntimamente ligadas y un ataque puede estar dirigido a las tres zonas de vulnerabilidad posibles.

Dentro de las amenazas de red se encuentran todos los ataques efectuados desde fuera de la red que tienen que ver con el tráfico que pasa a través de los routers, switches y a través del *firewall*. Este tipo de ataques en su mayoría se relacionan con la escucha secreta o *eavesdropping* y tiene múltiples variantes. El sniffing, o captura de información que circula por la red mediante herramientas diseñadas con tal fin es un ejemplo de este tipo de amenazas, el desbordamiento de la tabla de direcciones de un switch para bloquear su capacidad de direccionar paquetes a su destino es otro ejemplo muy significativo de las amenazas de red. *IP spoofing* o *MAC Address spoofing*, método en el que el atacante modifica la dirección *IP* o la dirección MAC de origen de los paquetes de información para

hacerse pasar por un usuario válido también se encuentra dentro de las amenazas dirigidas a explotar vulnerabilidades inherentes al diseño y seguridad de la red.

Las amenazas al servidor representan un peligro constante sobre el sistema operativo y sus elementos de control y administración y en general se caracterizan por explotar todas las vulnerabilidades relacionadas con la configuración del servidor, del software mismo y de los servicios ofrecidos por el servidor. Por ello es muy importante mantener actualizado el software con que trabajamos, aplicar los parches respectivos y tener información constante sobre las vulnerabilidades recientes encontradas en los servicios que se ofrecen en el servidor, a fin prevenir daños graves.

Un aspecto importante es el cuidado de las bases de datos y el correo electrónico, donde los niveles de confidencialidad, integridad y disponibilidad de la información contenida en ambos servicios determinan el nivel de protección y configuración de seguridad. Ataques como la inyección de SQL pueden provocar pérdidas importantes de información en bases de datos y requiere tanto de una buena protección a nivel de aplicación como de una protección a nivel de servidor.

Finalmente los ataques a aplicaciones, como los *exploits*, que buscan explotar las vulnerabilidades del sistema en general; la *denegación de servicios*, ataque en el cuál se busca dejar sin disponibilidad un servicio en la red; el defacement, ataque en el que se modifica el contenido de un sitio web, son ejemplos de este tipo de amenazas dirigidas contra las aplicaciones.

En general los ataques pueden estar dirigidos a más de un objetivo y buscar por todos los medios las vulnerabilidades del sistema. Incluso pueden perpetrarse ataques desde dentro de la red. Este tipo de amenazas puede ser desde el acceso no autorizado a directorios y archivos hasta la manipulación malintencionada de datos, cambios de archivos sin autorización y manejo no autorizado de recursos. Las amenazas de rechazo se asocian con usuarios que niegan haber ejecutado una acción, pero no existe forma de probar lo contrario. Un ejemplo de este tipo de amenaza es cuando un usuario puede realizar una operación prohibida en un sistema que no tiene la capacidad de rastrear dicha operación.

Una vez que el atacante superado de manera eficaz todas las defensas del sistema, es cuando el sistema corre mayores riesgos. La detección de la intrusión es de vital importancia en este punto. Si el atacante ha logrado el ingreso y trata de hacer una

elevación o escalada de privilegios, implementar una *puerta trasera* o utilizar los recursos de la red, es necesario tener un medio de control que permita verificar que acciones extrañas se llevan a cabo en el servidor y que ponen en peligro al sistema. Este es el tipo de amenaza contra las cuáles un sistema de detección de intrusos se enfoca y es el único medio de defensa capaz de detectar, prevenir y llevar a cabo acciones que permitan que dicha intrusión sea lo menos dañina posible al sistema.

Como se puede observar, existen muchas amenazas a las redes, en la tabla 2.1 Tipos de amenazas, se muestra una clasificación de las mismas. Una buena planeación y políticas de seguridad adecuadas son necesarias para tener una red lo más segura posible. Las medidas a implementar comienzan con la seguridad física del equipo, pero en cuanto al software se refiere, se determinan de acuerdo con las necesidades de confidencialidad, integridad y disponibilidad de la información que se requiere para la red, de acuerdo con el nivel de protección y configuración de seguridad que se requiere para los recursos de la red y los servicios que la misma ofrecerá a sus usuarios. Esta determinación de requerimientos es una tarea que requiere un planeación previa del tipo de sistema operativo que se requiere, los servicios a ofrecer, las características de seguridad de la red, del *firewall* y del sistema de detección de intrusos, aspectos que se analizan en el siguiente capítulo.

Tabla 1.2 Tipos de amenazas

Amenazas	Descripción
Humanas	Errores, omisiones, robo, fraude, ingeniería social, intrusos, espionaje.
Red	Sistemas mal diseñados, mala administración, accesos mal configurados.
Software	Fallos en las aplicaciones, códigos maliciosos, virus, exploits.
Hardware	Errores de fabricación, uso inadecuado.
Naturales	Terremotos, inundaciones, incendios, etc.

CAPÍTULO 2

ANÁLISIS DE REQUERIMIENTOS DE SEGURIDAD DE LA RED

ANÁLISIS DE REQUERIMIENTOS DE SEGURIDAD DE LA RED

En toda red existe la necesidad de proteger la integridad y confidencialidad de la información y el uso de sus activos, para determinar el grado de confianza que se puede depositar en un sistema informático existen criterios y normas de seguridad, pero, si se requiere mejorar el nivel de seguridad que existe en una red, se tiene que realizar una evaluación de seguridad, con lo cuál se puede determinar el estado de un sistema y los cambios que se pueden realizar para mejorar dicho estado. Para realizar una evaluación de seguridad se pueden hacer pruebas de vulnerabilidad que incluyen el análisis de una red y sus políticas y controles de seguridad; pruebas de seguridad, en las que se realizan auditorías de seguridad, escaneo de vulnerabilidades y pruebas de penetración, y finalmente, el reporte de las vulnerabilidades encontradas y las sugerencias para la implementación de mejoras.

A continuación se describe una norma de evaluación reconocida para la evaluación de la seguridad en un sistema informático. El presente trabajo se desarrolla en la parte técnica de dicho proceso de evaluación, que es en sí, la parte práctica en la que se implementan herramientas para el descubrimiento de vulnerabilidades. Cabe mencionar que el proceso de documentación y el descubrimiento de hallazgos y reporte de recomendaciones es también un elemento importante en este tipo de análisis, pero que no se trata a fondo en el presente documento.

2.1 Normas de evaluación reconocidas

Para la evaluación de la seguridad de una red existen diferentes modelos que permiten tener una mayor certeza sobre evaluación, calificación y mejora de la misma, estos modelos permiten cuantificar de alguna manera, los resultados de las acciones emprendidas sobre una red, tanto si es un resultado tangible como si se trata de un intangible. Estándares de evaluación reconocidos como NSA IAM de la agencia de seguridad de Estados Unidos[7], CESG CHECK (Council of Registered Ethical Security Testers) y SPD (Site Data Protection) de Mastercard, nos brindan una metodología estructurada que permite desarrollar la evaluación de seguridad de una manera organizada.

NSA IAM

La metodología de evaluación de la red NSA IAM (National Security Agency -Information Security-Assessment Methodology) es un método detallado y sistemático para la evaluación de vulnerabilidades desde una perspectiva organizacional, en oposición a una perpectiva técnica. Generalmente se pasan por alto los procesos, documentación y actividades informales que impactan directamente en la postura de una organización de seguridad y no necesariamente de manera técnica. El IAM fue desarrollado por la Agencia

de Seguridad Nacional de los Estados Unidos y asesores de INFOSEC y ha estado en práctica dentro del gobierno de ese país desde 1997.

Modelo NSA

- Avalúo (“Assessment”)
- Evaluación (“Evaluation”)
- Penetración (“Red Team”)
- Informe de Hallazgos
- Recomendaciones

Fase de avalúo: En esta fase se realiza la recopilación y examen de las políticas de seguridad, procedimientos, de arquitectura de seguridad y de flujo de la información. También se hace un análisis colaborativo de alto nivel y se evalúan las funciones críticas de la organización.

Fase de Evaluación: Se realizan pruebas de seguridad del sistema, firewalls, routers, equipos. Se realiza un escaneado de la red y se utilizan herramientas de penetración.

Fase de Penetración: Una evaluación de nivel 3 es no cooperativa y externa a la red de destino, con la participación de pruebas de penetración para simular el adversario adecuado. La evaluación es no intrusiva, por lo que dentro de este marco, una evaluación de nivel 3 implica una calificación completa de las vulnerabilidades. Se realizan pruebas de ingeniería social, simulación de ataques y hackeo de sistemas.

- Informe modelo de hallazgos: Se realiza un informe técnico, operacional y gerencial de los hallazgos determinados durante las fases previas.
- Modelo análisis de hallazgos y recomendaciones: En esta etapa se prepara un informe con las recomendaciones para mitigar las vulnerabilidades.

El presente trabajo se sitúa en el nivel 2, “Fase de evaluación” y nivel 3 “Fase de Penetración”, en la que se realizan pruebas de seguridad del sistema, escaneado de la red y uso de herramientas de penetración, para determinar el estado de seguridad del sistema e implementar mecanismos de corrección y mejora de la seguridad.

2.2 GENERALIDADES DE LA RED Y SERVICIOS A PROTEGER

De acuerdo con las necesidades de comunicación de los usuarios, los servicios mínimos que se requieren proteger en toda red son los siguientes:

2.2.1 Servicios Web

Las aplicaciones web son, por lo general, una colección de varios scripts de lenguajes como Javascript, PHP y HTML, que residen en un servidor Web e interactúan con una base de datos para la creación de una página Web con contenido dinámico[4]. Estas aplicaciones se implementan por la facilidad con que permiten la interacción entre clientes y proveedores de productos y servidico y porque permiten compartir y manipular información entre sí, mediante una interfaz Web que es usualmente independiente del sistema que se usa.

El servicio Web sigue el modelo cliente-servidor, por lo que se tiene un programa cliente Web instalado en el equipo del usuario que establece una conexión con el programa servidor que administra la presentación de la información en cada servicio Web. El programa cliente utiliza un número de puerto aleatorio con un valor superior a 1023 (entre 1024 y 65 535). El programa servidor usa por defecto el número de puerto 80 para el protocolo HTTP y el puerto 443 para el protocolo HTTPS. Pero un servidor puede configurarse para que utilice cualquier otro número de puerto. En este caso, el cliente tendría que introducir dicho puerto al indicar la dirección de destino. La gran mayoría usa el puerto 80 por lo que las reglas de filtrado a configurar en los *firewall* no consideran el tráfico Web a un número de puerto distinto.

2.2.2 Correo electrónico

El correo electrónico, junto con el Web, es el servicio más importante de los que ofrece Internet. La finalidad de este servicio es la de permitir el intercambio de mensajes entre usuarios. Para hacer uso del correo electrónico se utilizan unos programas conocidos como clientes de correo, aunque también existe la posibilidad de hacer uso de este servicio desde un servidor Web. A este servicio se le conoce como correo Web o Web mail.

Todos los mensajes de correo electrónico se mueven por Internet utilizando un formato especial llamado SMTP (Simple Mail Transfer Protocol, Protocolo Simple de Transferencia de Correo). Cuando se usa un programa cliente de correo, los mensajes que van recibiendo son guardados en el buzón de correo del servidor a la espera de que el usuario se conecte a Internet, ejecute su programa cliente y descargue los mensajes desde su buzón. La comunicación entre el programa cliente y servidor de correo para la descarga de mensajes se lleva a cabo con el protocolo POP (Post Office Protocol, Protocolo de la Oficina de

Correos, puerto 110). No obstante, cuando el programa cliente envía correos salientes a Internet lo hace directamente con el protocolo SMTP (puerto 25).

Aunque éstos son los protocolos comúnmente utilizados para el uso del correo electrónico, existen otros protocolos que aportan ciertas funcionalidades adicionales. El protocolo IMAP4 (Internet Mail Access Protocol, Protocolo de Acceso al Correo de Internet, versión 4) le da al usuario la posibilidad de acceder a otras carpetas de su buzón distintas a la carpeta Entrada (Inbox). Con POP3 el usuario sólo puede bajar los nuevos mensajes de la carpeta Entrada para posteriormente organizarlos localmente. Con IMAP4, el usuario puede acceder también a otras carpetas como Salida (Outbox), Enviado (Sent), Borrado (Deleted), etc., así como a carpetas de uso compartido. El protocolo IMAP4 utiliza una conexión *TCP* con el puerto 143 del servidor.

Otro protocolo de correo es el LDAP (Lightweight Directory Access Protocol, Protocolo Simple de Acceso al Directorio), que facilita la prestación de un servicio de directorio para que los usuarios puedan buscar las direcciones de correo desconocidas de los destinatarios de sus mensajes. El protocolo LDAP utiliza una conexión *TCP* sobre el puerto 389.

Si la conexión entre el programa cliente y el servidor de correo se realiza con el cifrado SSL, los puertos utilizados por el servidor suelen ser distintos: 995 para POP3, 993 para IMAP, 25 o 465 para SMTP y 636 para LDAP.

2.2.3 Mensajería instantánea

La mensajería instantánea es un servicio de Internet que permite a distintos usuarios estar en contacto permanente mientras están conectados a Internet. Ofrece herramientas de comunicación mediante texto escrito, voz o video. Adicionalmente se pueden realizar transferencias de archivos y compartir aplicaciones. Su relevancia en el entorno laboral es cada vez mayor, ya que permite que los trabajadores dispersos puedan trabajar en un entorno virtual como si todos estuviesen juntos en una sala. El inconveniente de los programas de mensajería es que son incompatibles entre sí al utilizar cada uno de ellos su propio sistema de comunicación.

2.2.4 Servicios de terminal

El servicio de terminal permite que un usuario haga un uso remoto de otra computadora (servicio central de terminal) en modo compartido. De esta manera no es necesario estar físicamente presente en el lugar donde se encuentra el servidor para hacer uso de sus recursos. *Secure shell* (SSH) es una herramienta muy popular y muy segura para comunicación en canales inseguros, que permite realizar conexiones remotas en otro host desde una estación de trabajo. Además de la conexión, SSH permite copiar datos de forma

segura y pasar datos de otra aplicación por un canal seguro tunelizado mediante SSH. *Secure shell* utiliza el puerto 22. Cabe mencionar que la mayoría de los sistemas operativos de libre distribución ya incluyen esta herramienta, por lo que no es necesario realizar su instalación.

2.2.5 Bases de datos

Una base de datos es una entidad que puede almacenar datos de forma estructurada. Se trata de una serie de datos organizados y relacionados entre sí, con la finalidad de ser usados de de una manera específica en beneficio de una persona u organización. Generalmente, una base de datos interactúa con diferentes programas y usuarios, los cuales utilizan los datos almacenados para sus tareas cotidianas, razón por la cual una base de datos es un servicio indispensable en una red que comparte información.

Una base de datos proporciona a sus usuarios el acceso simultáneo a los datos almacenados, quienes pueden hacer consultas de los registros existentes, ingresos o altas, bajas o eliminación de registros y modificaciones a los mismos, todo de acuerdo con los privilegios establecidos para cada usuario. Los puertos utilizados por las bases de datos varían de acuerdo con el sistema manejador de bases de datos.

2.3 MODELADO DE AMENAZAS Y GESTIÓN DE RIESGOS

De una forma simple, un riesgo es la relación entre los activos y las vulnerabilidades que cualquier atacante aprovecharía para adueñarse o interferir sobre estos[8].

Existen dos métodos para llevar a cabo el modelado de amenazas y gestión de riesgos, el método cuantitativo y el cualitativo. En el método cuantitativo se expresa el valor de cada activo en términos monetarios, tanto las medidas a implementar, como el presupuesto destinado a la seguridad está basado en análisis confiables y siempre determinados mediante el análisis de costo-beneficio.

En el método cualitativo no es necesario determinar el valor monetario de la información ni el costo de las medidas a tomar; no requiere una actualización constante de los valores de los activos que se evalúan, en su lugar, se consideran factores generales que afectan el desempeño del sistema, como son, las amenazas, las vulnerabilidades, el nivel de riesgo a que se expone el sistema y el posible tratamiento correctivo. En adelante se implementará este tipo de metodología ya que se adapta a todo tipo de sistema. Debido a que el presente trabajo está orientado a la parte lógica, no se analizarán los aspectos relativos a la parte física o seguridad de hardware, aspecto muy importante pero que no forma parte del

presente trabajo, por lo tanto en el siguiente análisis se consideran solamente las amenazas al software.

Las siguientes tablas muestran un análisis de las posibles amenazas a la red (tabla 2.1), al servidor (tabla 2.2) y a las aplicaciones (tabla 2.3), implementadas en el servidor. Se trata de un análisis global de las amenazas, las vulnerabilidades, nivel de riesgo, dependiendo del grado de daño que puedan provocar en los recursos y su tratamiento. No contiene todas las amenazas existentes o posibles, ya que un estudio de tal magnitud requiere también un tratamiento específico, bastante extenso y que escapa a los alcances de este trabajo, pero permite definir en forma básica cómo crear una respuesta ante un ataque a los recursos de la red. Asimismo, cabe mencionar que las tablas en cuestión son producto de una recopilación, análisis, y selección de las que he considerado las más sobresalientes de los documentos revisados.

Tabla 2.1, Amenazas a la red

Amenaza	Vulnerabilidad	Nivel de riesgo	Tratamiento
Ataque a los dispositivos de acceso a la red	Escaneo de puertos y consultas remotas.	**Alto**	Filtrar el tráfico entrante para aislar los puertos específicos en el host. Implementar un sistema de detección de intrusos. Implementar un *sniffer* de red para verificar la entrada de tráfico autorizado. Deshabilitar los servicios innecesarios con base en las políticas de seguridad de la institución.
Ataque a los dispositivos de monitoreo de la red	Mala configuración del *firewall* y/o del Sistema de Detección de Intrusos.	**Alto**	Verificar la configuración del Sistema de Detección de Intrusos en línea para normalizar el tráfico. Verificar la configuración del *firewall* y realizar pruebas de filtrado. Permitir el mínimo paso posible de paquetes al sistema. Revisar que las políticas de seguridad sean adecuadas para el nivel de seguridad

			requerido.
Banners	Brindar información mediante las pancartas de ingreso al sistema.	**Bajo**	Cambiar las pancartas que ofrecen información del sistema. Realizar las actualizaciones pertinentes al sistema operativo.
Rastreo de puertos	Puertos innecesarios abiertos.	**Alto**	Usar un detector de rastreadores, para descubrir actividades de escaneo. Revisar puertos abiertos en el sistema y determinar si son indispensables en ese momento. Usar protocolos cifrados para todas las comunicaciones que tengan contenido confidencial. Segmentar lo más posible servicios y accesos para evitar la divulgación de información.

Tabla 2.2, Amenazas al servidor

Amenaza	Vulnerabilidad	Nivel de riesgo	Tratamiento
Explotación de errores de configuración del sistema operativo.	Sobrecargas de búfer.	**Alto**	Establecer límites de uso de procesador. Realizar pruebas frecuentes con un software de escaneado de vulnerabilidades. Realizar las actualizaciones de software necesarias.
Administración abierta de usuarios y archivos	Errores en la configuración de cuentas de usuarios y	**Alto**	Establecimiento de cuotas. Acceso sistemas y programas indispensables.

	permisos de archivos.		Privilegio mínimo para usuarios y ejecución de aplicaciones. Realizar pruebas frecuentes con un software de escaneado de vulnerabilidades. Verificar que todas las contraseñas sean robustas. Eliminar cuentas no utilizadas. Establecer adecuados controles de acceso a personal autorizado.
Ataques a cuentas del sistema	Cuentas predeterminadas del fabricante.	**Medio**	Eliminar cuentas predeterminadas y cuentas no utilizadas. Verificar que las contraseñas sean actualizadas periódicamente. En su caso, cambiar contraseñas predeterminadas.
	Contraseñas en blanco o vulnerables.	**Alto**	Implementar un software de detección de contraseñas vulnerables. Establecer políticas de contraseñas para evitar contraseñas vulnerables e implementarlas mediante el uso de un software. Requerir que las contraseñas se cambien regularmente y eliminar las cuentas que no hayan iniciados nunca una sesión en la red. Escaneado de vulnerabilidades.
Puertos abiertos	Ejecución de servicios innecesarios.	**Bajo**	Eliminar aplicaciones que se ejecutan en nuestras máquinas que ya no sirven a ningún propósito útil, para evitar que se tengan puertos abiertos. Configurar el *firewall* para cerrar dichos puertos.
Bitácoras accesibles a	Mala configuración de	**Medio**	Verificar los permisos de las bitácoras y evitar el acceso mediante permisos de

usuarios.	las bitácoras y libre acceso a ellas.		administrador.
Ataques de fuerza bruta	Ataques a contraseñas.	**Alto**	Establecer un número máximo de intentos de autenticación. Revisión periódica a los intentos de acceso monitoreando los archivos /var/log/messages y /var/log/secure. Bloqueo de las direcciones *IP* de las que se reciben intentos de ataque en el archivo /etc/host.deny. Evitar el logeo como administrador en el servicio *secure shell*. Usar protocolos cifrados para todas las comunicaciones que tengan contenido confidencial.
Intrusiones	Acceso a cuentas de usuario no root.		Verificar que los archivos en el servidor no son alterados. Revisión periódica a los intentos de acceso monitoreando los archivos /var/log/messages y /var/log/secure.
	Acceso a cuentas root.		Revisión de los archivos de sistema. Realizar respaldos a las aplicaciones y bases de datos necesarias y considerar reinstalación del sistema. Análisis forense.
	Aumento de privilegios y puertas traseras.		Escaneo para revisar vulnerabilidades aprovechadas. Revisión de las bitácoras para descubrir actividades dañinas. Escaneo para ubicar y eliminar puertas traseras.

Tabla 2.3 Amenazas a las aplicaciones

Amenaza	Vulnerabilidad	Nivel de riesgo	Tratamiento
Ataque a base de datos	Saturación del búfer.	**Medio**	Corregir aplicaciones con vulnerabilidades. Eliminar las cuentas de usuario que no se usen y especialmente las predeterminadas. No aceptar cuentas sin contraseñas. Eliminar en la medida de lo posible los procedimientos almacenados.
Ataque a aplicaciones web	Ataques de inyección a SQL.	**Medio**	Validar entradas para el tamaño y tipo correctos, en particular si hay caracteres especiales. Revisar y corregir aplicaciones basadas en el Web. No permitir que se vean mensajes de error explícitos que muestren la consulta o parte de la consulta de SQL. No aceptar cuentas sin contraseñas. Mantener al mínimo los privilegios de las cuentas.
	Manipulación de URL y cruce de directorios.	**Medio**	Deshabilitar la visualización de los archivos de un directorio que no contiene un archivo índice. Eliminar directorios y archivos inservibles, así como secuencias de comandos innecesarios. Proteger el acceso a directorios que contienen datos importantes. Eliminar las opciones de configuración

			innecesarias. Impedir la visualización HTTP en páginas HTTPS accesibles.
Ataque Dos	DoS múltiple.	**Medio**	Bloquear redifusiones dirigidas. Considerar el bloqueo a los paquetes *ICMP*.
	Privación de recursos.	**Bajo**	Aplicar las últimas actualizaciones al sistema operativo y aplicaciones. Establecer cuotas de disco.
	Interrupción de servicios.	**Medio**	Aplicar actualizaciones constantes al sistema operativo. Probar las actualizaciones antes de aplicarlas a los sistemas de producción. Deshabilitar los servicios innecesarios

Las siguientes páginas Web también son útiles para investigar vulnerabilidades potenciales en los servicios de red:
http://www.securityfocus.com
http://www.milw0rm.com
http://www.packetstormsecurity.com
http://www.frsirt.com
http://www.mitre.org

2.4 DISEÑO DEL PERÍMETRO DE LA RED

El diseño del perímetro de la red permite determinar la posición de los elementos de seguridad implementados en toda la red con respecto a su exterior. Constituye la planeación física del sistema de defensa y se trata de la primera medida de seguridad que va a tener resultados importantes en la forma de administrar la red.

Existen diferentes configuraciones o arquitecturas que varían tanto en elementos como en el nivel de seguridad que pueden proveer a la red, a continuación se describen las características de algunas configuraciones básicas de acuerdo con la descripción hecha por Michael D. Bauer[8].

2.4.1 Arquitectura interna contra externa

La arquitectura más simple y la que se utiliza comúnmente es aquella en que se emplea en primer término un enrutador con filtrado de paquetes, pero no como única línea de defensa. Directamente detrás se sitúa un *firewall*, en este caso una estación ejecutando el sistema operativo Linux con iptables (filtrado de paquetes). No hay conexión directa con Internet o desde el enrutador externo con la red interna; todo el tráfico que entra o sale, pasa por el *firewall*. Una desventaja de esta configuración es que todo el tráfico de servicios públicos como SMTP (email) o HTTP (www) debe ser enviado a través del *firewall* a servidores internos. El paso de este tráfico no expone directamente a los servidores internos a algún ataque, pero si magnifica las consecuencias de servidores internos comprometidos.

El establecer una arquitectura en la que se controlan todos los servicios públicos en un *firewall* no parece tan malo a simple vista, pero resulta fácil deducir que el rendimiento podría resultar diferente al esperado; el *firewall* tiene que emplear todos los recursos disponibles para inspeccionar y mover paquetes, así que la seguridad podría verse afectada cuando algún servicio se ejecute en el *firewall*.

2.4.2 Arquitectura de zona desmilitarizada con tres interfaces de red en el *firewall.*

En esta configuración se tiene una zona desmilitarizada (*DMZ*, DeMilitarized Zone) para acceso a los servicios públicos. Una zona desmilitarizada es aquella red que contiene servicios públicamente accesibles, aislados de la red interna. De preferencia aislada de la red externa.

En este caso se cuenta con un enrutador y detrás del enrutador se coloca el *firewall* protegiendo la zona desmilitarizada, a su vez un switch o un enrutador filtra el acceso a la red local. Si se configura adecuadamente, el *firewall* usa diferentes reglas para evaluar el tráfico:

- De Internet a la zona desmilitarizada
- De la zona desmilitarizada
- De Internet a la red interna
- De la red interna a Internet
- De la zona desmilitarizada a la red interna
- De la red interna a la zona desmilitarizada

En este caso se simplifica la administración de la seguridad, ya que se toma a la zona desmilitarizada como una sola entidad con diferentes servicios internos como SMTP, FTP, Web, DNS.

2.4.3 Arquitectura débil de subred protegida

En esta arquitectura se utilizan dos enrutadores que actúan como *firewalls* filtrando paquetes, uno entre Internet y la zona desmilitarizada y otro enrutador entre la zona desmilitarizada y la red interna. Para tener acceso a la zona desmilitarizada o a la red interna vía el enrutador de protección, se utiliza un switch o un hub.

Este tipo de arquitectura tenía sentido cuando los enrutadores eran meros copiadores de grandes cantidades de datos en lugar de host con varias interfaces de red. Tiene algunos inconvenientes importantes como son: los enrutadores generalmente se encuentran bajo el control de una persona diferente que la encargada del *firewall*; los enrutadores tienden a tener un control administrativo más débil que los *firewalls*; y los enrutadores tienden a ser más vulnerables a ataques que una computadora bien configurada.

2.4.4 Arquitectura fuerte de subred protegida

En esta arquitectura se utilizan dos *firewalls* propiamente hechos, que son más sofisticados que los enrutadores. El *firewall* externo filtra las peticiones desde el exterior hacia el interior de nuestra red, inmediatamente después del *firewall* se encuentra un switch o un hub que redirige el tráfico hacia los servicios públicos o hacia el *firewall* interno. El *firewall* interno filtra el tráfico dirigido hacia la red interna.

Esta arquitectura es útil cuando hay que soportar grandes volúmenes de tráfico y en entornos de *firewalls* heterogéneos.

2.4.5 El Sistema de Detección de Intrusos

No existen reglas fijas para situar el Sistema de Detección de Intrusos, pero dependiendo de la ubicación de éste en la red, serán los resultados que se obtengan y el aprovechamiento que tendremos del mismo. Dependiendo de la estructura de la red, un SDI (Sistema de Detección de Intrusos) puede situarse:

a) Delante del *firewall*
b) Detrás del *firewall*
c) Combinación de las dos anteriores
d) En el *firewall*
e) Configuraciones avanzadas

a) Delante del *firewall*:

El SDI comprobará todos los ataques que se produzcan y le evitará una gran carga al *firewall*, que se encargará de bloquear ataques efectivos, a su vez esto generará una gran

cantidad de logs y un exceso de información que podría resultar contraproducente. Debido a la excesiva carga de información se puede perder de vista información vital sobre ataques importantes a nuestra red.

b) Detrás del *firewall*:

El SDI no analizará todos los ataques se dirijan a la red, sino únicamente el tráfico que haya sido permitido por el *firewall*, es decir, que haya sido previamente filtrado, con lo que se genera una menor cantidad de logs y de información y se enfoca realmente en ataques potencialmente más peligrosos.

c) Combinación de las dos anteriores:

Con esta configuración se tiene un mejor control de los eventos que se monitorizan en ambos lados del *firewall*, se mejora la seguridad porque se puede establecer una correlación entre los ataques detectados en ambos lados del *firewall*. La desventaja es que se requieren dos máquinas para implementar este tipo de configuración.

d) En el *firewall*:

En esta configuración una máquina opera como *firewall* y de SDI a la vez, al igual que en el caso del SDI delante del *firewall*, se monitoriza todo el tráfico de la red, por lo que se genera una gran cantidad de logs y se pueden llegar a perder de vista los ataques efectivos dirigidos hacia nuestra red.

e) Combinaciones avanzadas:

Surgen de la necesidad de tener un mayor control y seguridad en áreas específicas de la red, segmentos o hosts individuales, dependiendo de la arquitectura o diseño del perímetro de red implementado.

2.5 CARACTERÍSTICAS DEL SISTEMA OPERATIVO

La elección del sistema operativo para el sistema de seguridad es una parte fundamental de lo que se conoce como la base informática de confianza, una lista de elementos que proporcionan seguridad, entre los que destacan el sistema operativo, los programas, el hardware de la red, las protecciones físicas e incluso los procedimientos.

Anteriormente un sistema operativo tenía un número limitado de entradas posibles, eran pocos los programas de aplicación implementados. Sin embargo, ahora con Internet y la diversidad de programas que se usan en las computadoras, los sistemas operativos se han vuelto menos seguros y más susceptibles a fallas. Además, la tendencia de los proveedores

de software es intentar que el usuario tenga todo preparado una vez que instala su sistema operativo, esto puede parecer muy bueno pero representa un problema muy grande en cuanto a seguridad; ya que la mayoría de las opciones de seguridad están desactivadas de forma predeterminada, muchos programas y servicios se cargan automáticamente, tanto si las necesita el usuario como si no, y se introducen muchos "extras" en el sistema en un esfuerzo por ganar la competencia entre desarrolladores de software.

El sistema operativo que se va a instalar y sobre el cuál se va a implementar el Sistema de Detección de Intrusos es GNU/Linux. Este sistema, implementación de libre distribución tipo Unix, es usado en una amplia variedad de plataformas de hardware y computadoras, incluyendo las computadoras de escritorio, servidores, supercomputadoras, mainframes y teléfonos celulares. Se encuentra protegido con la licencia GNU.

Linux contiene todas las funciones y características necesarias de cualquier sistema operativo; pero en especial implementa características que lo definen como un sistema operativo de alto rendimiento en servidores; entre ellas: es multitarea, es decir, puede ejecutar varios programas al mismo tiempo; tiene capacidad multiusuario, varios usuarios pueden estar trabajando al mismo tiempo en la computadora; es multiplataforma, es multiproceso, se puede hacer una implementación de varios procesadores para realizar objetivos comunes; tiene protección de memoria entre procesos; permite usar bibliotecas enlazadas tanto estática como dinámicamente; es compatible con *Posix*, System V y BSD a nivel fuente; y por el amplio desarrollo, tiene cada vez más soporte para nuevo hardware, todo el código fuente está disponible, incluyendo el núcleo completo y todos los drivers, las herramientas de desarrollo y todos los programas de usuario.

Existen numerosas distribuciones Linux, también conocidas como "distros". Entre ellas destacan el proyecto Debian/GNU y Fedora, ambos sistemas de propósitos generales usados frecuentemente en servidores de aplicaciones, ambos cuentan con el respaldo del movimiento de *software libre* y de todas las ventajas de que muchos programadores están trabajando continuamente para mejorar el código fuente continuamente.

2.6 REQUERIMIENTOS DE SEGURIDAD DEL SERVIDOR

Fortalecer un sistema Linux después de su instalación básica no es una tarea trivial, se deben determinar todas las vulnerabilidades ofrecidas por la instalación por default y modificarlas de manera que no signifiquen un peligro para la seguridad del sistema. Una tarea importante es mantener actualizado al sistema con parches de seguridad con cierta regularidad y deshabilitar todos los servicios que no sean necesarios.

La computadora en que estén instaladas las herramientas de seguridad requiere de una serie de medidas de seguridad bien definidas tanto en aplicaciones como en servicios, para

evitar al máximo las vulnerabilidades y amenazas que den lugar a algún ataque y puedan poner en riesgo la seguridad del servidor y por tanto de la red que éste defiende.

2.6.1 Políticas sobre contraseñas

La importancia que tienen las contraseñas en cualquier sistema es muy grande; una contraseña permite identificar a un usuario y saber de acuerdo con sus estatus qué tiene permitido hacer en el sistema. Es de vital importancia por tanto, asegurar que este sistema de identificación básico no pueda ser fácilmente vulnerado por cualquier persona que quiera usurpar una identidad. Para ello se tiene que hacer una revisión contínua de contraseñas, a fin de verificar que son fuertes, deben contener caracteres alfanuméricos, y caracteres especiales; además dichas contraseñas deben estar en uso y tener un tiempo de expiración. El archivo /etc/shadow debe ser legible únicamente por el administrador.

2.6.2 Políticas sobre cuentas de usuario

Se deben crear cuentas de usuario solamente en los casos necesarios y con establecimiento de cuotas, es decir, con cierto espacio de disco y determinados privilegios, tener un control estricto de los usuarios y grupos creados en el sistema y sus privilegios. Desactivar o remover cuentas innecesarias que se crean para servicios que no estarán en uso y limitar el uso del procesador para evitar el desbordamiento de buffer.

Hacer la asignación adecuada de permisos a cada directorio para asegurar que los usuarios únicamente puedan tener acceso a los archivos que les pertenecen y puedan ejecutar los programas permitidos.

2.6.3 Políticas de acceso remoto

El acceso remoto tiene que realizarse mediante una herramienta segura como *secure shell*, que nos permitirá realizar conexiones remotas hacia el servidor con la finalidad de administrar los recursos instalados en el mismo. No debe permitirse el acceso como usuario root y debe estipularse un número máximo de intentos de autenticación para evitar el ataque por fuerza bruta. Además se debe estar muy atentos a los intentos de acceso monitoreando permanentemente los archivos /var/log/messages y /var/log/secure y bloquear definitivamente las direcciones *IP* de las que se reciben intentos de ataque en el archivo /etc/host.deny.

2.6.4 Políticas de respaldos

Es indispensable crear, conservar y proteger los recursos de información de la red, del sistema y de las aplicaciones; para ello se requieren políticas para crear respaldos de todos los elementos participantes en el desarrollo fundamental de la red.

Estos respaldos proporcionan la seguridad de recuperación ante algún incidente dañino, imprevisto o poco usual que modifique alguna característica de nuestra red perjudicando su funcionamiento. La información a respaldar incluye a los archivos de sistema de los equipos de la red, bitácoras del Sistema de Detección de Intrusos, resultados de escaneo de vulnerabilidades, archivos de información sobre conexión remota a las máquinas, respaldo de bases de datos, de aplicaciones y de programas desarrollados para el servicio de la red.

2.7 HERRAMIENTAS DE FILTRADO Y MONITOREO

Existen muchos elementos que permiten mantener un sistema seguro, entre ellos el *firewall* y el Sistema de Detección de Intrusos, que aunque son elementos importantes en el ámbito de la seguridad, no constituyen una garantía de fiabilidad si no se implementan otras herramientas, componentes y medidas de seguridad que forman parte de las estrategias de protección de un sistema.

Algunas herramientas de seguridad para filtrado y monitoreo son distribuidas con los sistemas operativos, otras se producen especialmente para reforzar la seguridad en una parte específica de la red o del servidor. El control de acceso, los mecanismos de identificación y autenticación, el cifrado de la comunicación, y diversos comandos de monitoreo del sistema, son elementos que se distribuyen comúnmente con el sistema operativo.

Ciertas herramientas se han diseñado para abordar problemas específicos, entre ellas destacan los programas para el reforzamiento del sistema operativo, los escáners de vulnerabilidades, los *sniffers* de red, los programas seguros de conexiones remotas, descifradores de contraseñas, los programas antivirus, herramientas de filtrado de paquetes y de detección de intrusiones. Se trata de herramientas para diversos problemas de seguridad de la red, muchas de ellas de libre distribución y generalmente disponibles para los sistemas Linux.

Como se dijo anteriormente, un elemento importante es el *firewall*, un dispositivo que actúa en la primera línea de defensa frente a cualquier ataque entrante, que puede desviar o suavizar el efecto de muchos tipos de ataques y proteger los servidores y estaciones de trabajo. Un *firewall* también puede evitar el acceso a las máquinas internas desde fuera de

la red. Correctamente configurado, un *firewall* nos ayuda a estar más seguros frente a los ataques exteriores.

Existen dos formas de configurar un *firewall*, una de ellas es permitir todo el tráfico y después añadir el comportamiento que deseamos bloquear. La otra forma es denegar todo y añadir después lo que deseamos permitir. Este último método es más fácil de mantener con seguridad que la otra solución, ya que nos permite tener cerrado el sistema a ataques desconocidos e identificar y abrir el sistema únicamente a los servicios inofensivos y conocidos.

El *firewall* debe respetar el mapa de servicios internos y externos de la red, es decir, permitir la conexión a los servidores que necesitan hacerlo desde el exterior, en los puertos correspondientes, por ejemplo, abrir los puertos 80 y 443 para web, el puerto 22 para *Secure Shell*, y así sucesivamente, de acuerdo con la arquitectura definida para la red y las políticas de seguridad de la institución.

El núcleo de Linux incluye la facilidad del filtrado de paquetes desde la versión 1.1.x. A mediados de 1999 apareció una nueva generación de *firewall* de Linux desarrollada por Rusty Russel, y que tiene como nombre iptables. Esta aplicación implementa la inspección dinámica de paquetes y utiliza de forma más eficiente la cadena de reglas que maneja el tráfico que se enruta a otras redes. El *firewall* de Linux forma parte del núcleo del sistema operativo, y por tanto, está siempre presente en la mayoría de las distribuciones, aunque puede que no esté instalado. *Iptables* es una herramienta muy eficaz pero compleja, y normalmente se recomienda para usuarios que están familiarizados con los *firewalls* y la forma de configurarlos.

La otra herramienta importante en la seguridad de una red es el Sistema de Detección de Intrusos, una herramienta de seguridad que ayuda a supervisar los eventos ocurridos en una red comparándolos con patrones previamente definidos que impliquen cualquier tipo de actividad sospechosa o maliciosa para la red. Un SDI alerta y previene de manera anticipada sobre cualquier actividad sospechosa en la red pues está diseñado para detectar las primeras etapas de un ataque como es el barrido de puertos.

El SDI configurado correctamente sirve para encontrar usos indebidos de los recursos de la red, esto se hace comparando las firmas con la información recogida en busca de coincidencias; además puede detectar anomalías mediante el uso de técnicas estadísticas que definen de forma aproximada lo que debería ser el comportamiento normal o usual.

Uno de los Sistemas de Detección de Intrusos más populares y efectivo es Snort, un detector de intrusos disponible bajo la licencia GPL, de libre distribución que funciona bajo

plataformas Unix/Linux y Windows. Snort contiene una gran cantidad de patrones definidos y actualizaciones ante ataques, barridos y vulnerabilidades detectadas en boletines de seguridad. Incluye un lenguaje flexible y potente para la creación de reglas e incluye filtros predefinidos contra ataques frecuentes; se puede utilizar como *sniffer* de red, registro de paquetes o como un SDI normal.

En el siguiente capítulo se explica la funcionalidad de algunas herramientas de libre distribución que se implementan en sistemas Linux y el uso de dichas herramientas para obtener el sistema de seguridad tanto de un servidor como de la red. Entre las herramientas descritas adelante, se encuentran herramientas para el escaneo de vulnerabilidades, el análisis de paquetes, descifrado de contraseñas, filtrado de paquetes y monitoreo de la red.

CAPÍTULO 3

HERRAMIENTAS DE SOFTWARE LIBRE A IMPLEMENTAR EN LA RED

HERRAMIENTAS DE SOFTWARE LIBRE A IMPLEMENTAR EN LA RED

Existe un gran número de herramientas de software libre para implementar ya sea en una evaluación de seguridad para una red o bien, para implementar acciones de defensa que permitan tener certeza de la seguridad en un sistema. Las principales ventajas que ofrecen estos programas son, su efectividad, ya que se trata de herramientas que han sido probadas por un gran número de administradores de redes; su constante desarrollo, pues muchas personas trabajan para mejorar el código de cada programa; el bajo o nulo costo y la facilidad de acceso a estos programas.

Existen varias encuestas en Internet sobre las herramientas de software libre más populares en el área de seguridad, entre ellas destaca la encuesta hecha por insecure.org en su sitio http://sectools.org/ en el año 2006, denominada "Top 100 Network Security Tools". Muchas de estas herramientas siguen siendo muy populares y permanecen en uso en el 2010, además se pueden encontrar otras encuestas publicadas en otros sitios y blogs, entre las cuales destacan las siguientes:

http://agilworld.com/freeware-download/networking-and-admin-download/special-top-14-networks-internet-security-tools

http://www.seguridaddigital.info/index.php?option=com_content&task=view&id=128&Itemid=26

En este capítulo se enumeran ciertas herramientas de software libre que se han elegido por ser las más eficaces de acuerdo con las encuestas antes mencionadas, de acuerdo con la bibliografía consultada y con base en el sistema defensa que se pretende desarrollar a fin de que funcione correctamente, el cual considera:

- Sistema operativo con seguridad reforzada
- Seguridad en una terminal remota
- Scanner de vulnerabilidades
- Scanner de vulnerabilidades web
- Cortafuegos
- Protección de contraseñas
- Sistema detector de intrusos

3.1 HERRAMIENTAS PARA REFORZAR EL SISTEMA OPERATIVO

Con la finalidad de que el sistema de seguridad de una red sea lo más confiable posible, es necesario implementar ciertas herramientas adicionales para garantizar que el sistema en su totalidad carezca de vulnerabilidades que signifiquen brechas de seguridad importantes. En

el caso del sistema operativo, dichas herramientas permiten realizar configuraciones más seguras en los equipos y evitar en la medida de lo posible las vulnerabilidades asociadas a las instalaciones hechas por defecto en los sistemas.

A continuación se describen las herramientas que se implementan en el sistema de defensa de la red.

3.1.1 OpenSSH[9]

SSH (*Secure Shell*) es el nombre de un protocolo y el nombre del programa que implementa dicho protocolo. Se trata de un programa que sirve para iniciar sesiones en servidores remotos, ejecutar comandos remotamente, copiar archivos entre distintos servidores, ejecutar aplicaciones X11 remotamente y realizar túneles *IP* cifrados. SSH utiliza técnicas de cifrado que hacen que la información sea transferida por el canal de comunicación de manera no legible, lo que evita que una tercera persona pueda tener acceso a datos confidenciales e información intercambiada durante la conexión. SSH funciona con una arquitectura cliente-servidor.

OpenSSH es la versión SSH desarrollada por OpenBSD, una alternativa libre y abierta al programa *Secure Shell*, que es software propietario. OpenSSH es una herramienta de libre distribución disponible de manera predefinida en la mayoría de los sistemas operativos basados en Linux.

Debido a que esta herramienta se implementa en la mayoría de los sistemas Linux no se mostrará su instalación, sino, únicamente características básicas sobre su configuración y su uso.

Configuración del servidor sshd

El archivo de configuración del servidor se llama sshd_config y generalmente se encuentra en el directorio /etc/ssh. Este archivo que se debe modificar para obtener un mejor rendimiento de SSH.

La mayoría de los ataques por medio de Secure Shell se realizan a través del puerto 22, por lo que es recomendable cambiar el número de puerto de conexión por un valor mayor a 1024.

La línea a editar es:

#Port 22

Port 2330

No es recomendable que el administrador se logee directamente, ya que se puede cambiar de usuario una vez que se ingresa como usuario normal, se tendría que cambiar la siguiente línea:

#PermitRootLogin yes

PermitRootLogin no

El número máximo de equivocaciones al ingresar el usuario y/o la contraseña se recomienda cambiarse a dos.

MaxAuthTries 2

Maxstartups indica la cantidad de pantallas de login o cantidad de conexiones simultáneas de login que permitirá el sshd por ip que intente conectarse.

MaxStartups 3

Con estas directivas bien configuradas en el archivo /etc/ssh/sshd_config se puede incrementar enormemente la seguridad en este servicio.

SSH puede asegurar un inicio de sesión en un sistema remoto de distintas maneras. La sintaxis SSH básica para un inicio de sesión remota es:

ssh –l login hostname

o

ssh login@hotname

En la siguiente tabla, (tabla 3.1, opciones de SSH) se muestran las opciones de SSH.

Tabla 3.1 Opciones de SSH

Opciones	Descripción
-c protocol	Utiliza un protocolo criptográfico determinado.
-p port #	Se conecta a otro número de puerto, en lugar del predeterminado (22).
-P port #	Usa un puerto específico que no forma parte de la lista estándar de puertos propietarios, lo que normalmente significa un número por encima de 1024. Esto puede ser útil si se tiene un *firewall* que tira las conexiones en números de puertos inferiores.

-v	Muestra la salida larga. Útil para la depuración (verbose).
-q:	Informa de manera silenciosa, contrario del modo largo.
-C:	Utiliza compresión del tráfico cifrado. Puede ser útil para conexiones demasiado lentas, como las telefónicas.
-1	Obliga a SSH a utilizar sólo la versión 1 del protocolo SSH. No está recomendado pero puede ser necesario si el servidor al que estamos conectando no está actualizado a la versión 2.
-2	Obliga a SSH a usar sólo la versión 2 del protocolo SSH. Puede evitar que nos conectemos a algunos servidores.

3.1.2 BASTILLE LINUX[10]

Bastille Linux es una herramienta de seguridad para reforzar el sistema operativo. Se trata de un conjunto de secuencias de comandos que permiten realizar la configuración del sistema de manera simplificada, de manera que podemos hacer que el sistema operativo sea lo menos vulnerable posible. Mediante una buena configuración de Bastille Linux se pueden eliminar un gran número de vulnerabilidades comunes en plataformas Linux/Unix.

Con Bastille se pueden realizar cuatro pasos para asegurar el sistema:

- Aplicar un *firewall* para prevenir el acceso a posibles servicios vulnerables.
- Aplicar actualizaciones para los agujeros de seguridad conocidos.
- Realizar un SUID, root audit.
- Desactivar o restringir servicios innecesarios.

Los anteriores pasos se subdividen en módulos que cubren cada una de las áreas anteriormente descritas.

- Módulo ipchains: permite la configuración de un *firewall* para filtrar el tráfico y así proteger la red.

- Módulo patch download: ayuda a mantener actualizados los servicios que se usan en el servidor. Lo más importante es aplicar actualizaciones a los agujeros de seguridad conocidos.

- Módulo file permissions: este módulo permite realizar una auditoría sobre los permisos de archivos en el sistema. Restringe o habilita permisos para el uso de servicios y archivos binarios.

- Módulo account security: con este módulo se obliga al administrador a tener "buenas prácticas" en el manejo de cuentas, además de que provee algunos trucos para realizar de una manera más efectivo el manejo de cuentas.

- Módulo boot security: en este módulo podemos habilitar ciertas opciones de arranque para hacerlo más seguro y evitar que sean aprovechadas las vulnerabilidades inherentes a los sistemas operativos tipo Linux/Unix.

- Módulo secure inetd: con este módulo podemos deshabilitar servicios que podrían significar una vulnerabilidad grave al poder ser iniciados con privilegios de root. En caso de que no se pueda deshabilitar dichos servicios por ser necesarios para los usuarios o para el sistema, entonces se restringen los privilegios. En este caso se recomienda deshabilitar telnet, ftp, rsh, rlogin y talk. Pop e imap son protocolos de correo que deben ser deshabilitados únicamente si no se requieren en el sistema.

- Módulo disable user tools: en este módulo podemos restringir el uso del compilador para que únicamente sea accesible para el usuario root.

- Módulo configure misc pam: partiendo de la idea de que un servidor debe tener únicamente los servicios y herramientas necesarias, en este módulo se pueden hacer ciertas modificaciones que harán difícil a los usuarios, incluidos los usuarios "nobody", "web" y "ftp", que abusen de los recursos para producir un ataque de *denegación de servicios*.

- Módulo logging: aquí manejamos y configuramos las bitácoras adicionales que contienen información del sistema, mensajes del kernel, mensajes de errores graves, etc.

- Módulo miscellaneous daemons: en este módulo se desactivan todos los demonios del sistema que no sean necesarios, los cuáles se pueden volver a activar con el comando chkconfig.

- Módulo sendmail: permite deshabilitar comandos sendmail que son usados para obtener información acerca del sistema para realizar cracking o spamming.

- Módulo remote access: el acceso remoto se puede configurar de manera que se realice mediante *secure shell* client, un programa seguro de cifrado de información.
- Los módulos dns, *apache*, printing y ftp permiten configurar cada uno de los servicios que previamente se han asegurado en caso de que no se haya deshabilitado y establecer cierta seguridad en la forma de levantar dichos servicios.

Todos los módulos mencionados anteriormente se establecen al momento de responder a las preguntas que el script de Bastille realiza al ejecutarse, por lo que se simplifica la tarea de configuración de esta herramienta, sin embargo, es muy importante contestar dicho cuestionario de acuerdo con las políticas de seguridad establecidas por cada institución.

3.2 PROTECCIÓN DE CONTRASEÑAS Y ATAQUES DE FUERZA BRUTA

Las contraseñas en los sistemas Unix y Windows se almacenan en resúmenes codificados "de un sentido" y estas contraseñas no pueden decodificarse. En su lugar, el inicio de sesión de un usuario sigue un sencillo proceso. Cuando un usuario intenta iniciar una sesión en el sistema, el sistema Unix invoca a su función crypt( ) para generar un resumen codificado temporal. Si el resumen codificado para la contraseña introducida no concuerda con el resumen codificado almacenado, entonces se ha introducido una contraseña errónea. El resumen codificado almacenado no se decodifica. Tomar el resumen codificado de una palabra conocida y compararlo con el resumen codificado de la contraseña objetivo es la base para los ataques de descifrado de contraseñas.

Algunas técnicas de fuerza bruta se aprovechan de las mejoras en el rendimiento del hardware. El intercambio tiempo-memoria significa que en realidad es más sencillo generar todo un diccionario de contraseñas y ejecutar búsquedas de resúmenes codificados de contraseñas. Estos diccionarios generados, también denominados Rainbow Tables, están formados por todas las claves de una determinada longitud y con un determinado contenido. Por ejemplo, un diccionario podría estar formado por todas las combinaciones de siete caracteres alfanuméricos, con mayúsculas y minúsculas, mientras que otro diccionario podría estar formado por combinaciones de nueve caracteres de letras sólo mayúsculas y sólo minúsculas. Estos diccionarios están codificados con *DES*, *MD5* o cualquier algoritmo preferido por el usuario. Estos diccionarios pueden tener un tamaño de cientos de gigabytes; sin embargo ahora es fácil encontrar equipos con un terabyte de capacidad.

Con estos diccionarios a mano, un atacante sólo tiene que esperar a realizar una sola búsqueda en el diccionario. Las ventajas de esta técnica pronto se hacen evidentes, cuando tenemos en cuenta que las búsquedas a través de cientos de contraseñas sólo requieren cientos de iteraciones redundantes a través del conjunto de claves. El tiempo necesario para descifrar una contraseña es muy variado y depende del método o proceso que el atacante usa para crear sus diccionarios. Debido a la posibilidad de tener ataques a las contraseñas en el sistema, es necesario buscar herramientas que limiten o contrarresten este tipo de ataques. Entre las herramientas que se pueden implementar para realizar un ataque a contraseñas se encuentran John the Ripper, Cain and Abel, THC Hydra, Aircrack, L0phtcrack y Airsnort, entre otras. En este caso se usará John the Ripper, una herramienta bastante probada y usada tanto por administradores de red como por hackers.

3.2.1 John The Ripper[11]

Esta herramienta es una de las más populares, rápida y versátil disponible entre las herramientas de fuerza bruta. Admite seis esquemas de resúmenes codificados diferentes, que cubren varias versiones de Unix y los resúmenes codificados LANMan de Windows, también llamado NTLM (usados en NT, 2000 y XP). Puede utilizar listas de palabras especializadas o reglas de contraseña basadas en el tipo de carácter y su ubicación. Funciona en, al menos 13 sistemas operativos diferentes y admite varios procesadores, incluyendo las mejoras de velocidad especiales para chips Pentium y RISC.

Con John the ripper se pueden utilizar complicados archivos de diccionario, usar combinaciones de contraseña específicas o distribuir el proceso entre varias computadoras. Este programa permite realizar ataques específicos, muy importantes si se conocen algunas de las políticas implementadas en el sistema que se va a atacar. El diccionario predeterminado de John es el archivo password.lst, pero se pueden hacer cambios en las palabras del diccionario, usando la opción –rules, y de esta manera mejorar la frecuencia de aciertos al descifrar las contraseñas.

En la tabla 3.2 se describen las reglas comodín que indican dónde debe colocarse el nuevo carácter**:**

Tabla 3.2 Reglas comodín de John the Ripper

Símbolo	Descripción	Ejemplo
^	Antepone el carácter	^[01] 0password

		1password
$	Añade el carácter	$[%"] password% password"
i[n]	Inserta un carácter en la posición n	i[4][AB] passAword passBword

3.3 SEGURIDAD DEL EQUIPO

Por seguridad del equipo se entiende tanto el rastreo de vulnerabilidades del sistema de red como el rastreo de vulnerabilidades en las aplicaciones y servicios implementados en la red. Este es un campo extenso y por lo tanto muy difícil de monitorear. Por ello se han creado herramientas que simplifican la tarea de administrar un sistema y que permiten estar al tanto de vulnerabilidades, ataques y fallos en el sistema, ejemplos de estas herramientas son SARA (Security Auditor´s Research Assistant), SAINT (Security Administrator´s Integrated Network Tool) y Nessus. A continuación se describe Nessus, una herramienta muy completa para reforzar la seguridad del equipo.

3.3.1 Nessus[12]

Nessus es un rastreador remoto de vulnerabilidades que realiza un barrido básico pero eficiente, de los sistemas de nuestra red, buscando fallos en la configuración de la red y vulnerabilidades de aplicaciones.

Se trata de una aplicación cliente/servidor. El servidor ejecuta las pruebas y el cliente configura y controla las sesiones. El hecho de que el cliente y el servidor puedan estar por separado ofrece algunas ventajas, lo que significa que tenemos nuestro servidor de escaneado fuera de nuestra red, aunque accesible a él desde dentro de nuestra red a través del cliente.

Nessus cuenta con un lenguaje de series de comandos denominado lenguaje de creación de series de comandos de ataque para Nessus, *NASL*. En Nessus, cada rastro de vulnerabilidades es una serie de comandos o un complemento diferente, escritos en *NASL*. Esta arquitectura modular permite añadir fácilmente rastros y posibles pruebas de ataques, a

medida que se descubren nuevas vulnerabilidades. Además, puede reconocer los servicios que se están ejecutando en cualquier puerto, no sólo en el número de puerto de la autoridad para la asignación de números en Internet (IANA).

Si tenemos un servidor Web ejecutándose en el puerto *TCP* 8888, Nessus lo encontrará y realizará en él pruebas de Interfaz común de acceso (CGI). Por otro lado, si Nessus no encuentra servidores Web en el sistema que está examinando, no realizará más pruebas de servidor Web o CGI en este sistema.

Nessus es un programa minucioso, muchos de los complementos no sólo buscarán vulnerabilidades, sino también intentará aprovecharlas e informar del resultado. A veces esta actividad puede ser peligrosa, porque conseguir aprovechar una vulnerabilidad puede hacer que se "cuelgue" el sistema que estamos examinando, haciendo que quede inutilizado o que se pierdan datos. Sin embargo, como Nessus nos proporciona descripciones completas de lo que hace cada prueba de vulnerabilidad, podemos decidir qué pruebas podemos realizar sin peligro.

Los informes de Nessus son muy amplios, bien organizados y disponibles en muchos formatos, como texto plano, HTML, LaTeX y PDF (sólo en cliente de Windows). Clasifica los eventos de seguridad desde indicaciones hasta advertencias y agujeros, cada uno con un nivel de riesgo que varía entre bajo y muy alto.

Para instalar Nessus con todas las funciones se requieren los paquetes GIMP Toolkit (GTK) y *OpenSSL*. Nessus se puede descargar en cuatro paquetes diferentes: nessus-libraries, libnasl, nessus-core y nessus-plugins.

Nessus usa SSL para que los clientes Nessus de redes remotas puedan comunicarse con seguridad. También permite a estos clientes Nessus usar certificados SSL para autenticación y realizar comprobaciones sobre servicios basados en SSL, como servidores HTTPS y demonios SSH. Si se compila Nessus en un sistema con *OpenSSL* instalado, Nessus se configurará automáticamente para emplear SSL. Esto se aplica al paquete nessus-libraries.

Sin SSL, las comunicaciones de red entre el cliente y el servidor no serían confidenciales, una persona con acceso a la red local podría interceptar información como contraseñas o estadísticas de vulnerabilidad.

Para usar SSL, es necesario configurar un certificado para el Nessus. Esto se consigue fácilmente ejecutando el comando nessus-mkcert como administrador. A continuación se realizarán una serie de preguntas de certificado SSL estándar, preguntas que deben ser contestadas verazmente ya que a partir de esto se creará una conexión segura entre cliente y servidor. A partir de esta información, nessus-mkcert genera los archivos necesarios para

crear una CA, o Autoridad de Certificado, para el servidor Nessus. Los certificados se utilizan para firmar digitalmente otros certificados y autorizarlos como certificados de confianza.

La utilidad nessus-adduser puede usarse para agregar un usuario a la base de datos de Nessus. Cuando ejecutemos nessus-adduser como usuario raíz, se nos pedirá un nombre de inicio de sesión y un tipo de autenticación. La autenticación por contraseña normalmente será suficiente para las instalaciones de Nessus a pequeña escala. En esta fase se pueden configurar reglas de acceso para el usuario.

La autenticación de usuario basada en certificados necesita algunos pasos más. En lugar de usar el comando nessus-adduser, se ejecuta la herramienta nessus-mkcert-client con privilegios de administrador. Se crearán los archivos de certificado de usuarios y se registrarán en la base de datos de usuario.

Cuando se ejecute Nessus por primera vez, se creará un archivo .nessusrc en el directorio inicial. A la siguiente ejecución del cliente Nessus, la autenticación se realizará mediante certificados. También se puede especificar el certificado y la clave en la interfaz gráfica de usuario.

La autenticación del certificado aumenta significativamente la seguridad del modelo cliente/servidor de Nessus, especialmente si la clave privada está protegida con contraseña.

El archivo nessusd.conf, instalado por defecto en /usr/local/etc/nessus/, contiene varias opciones de examen global, éstas pueden editarse en el archivo nessusd.conf para que se apliquen a todos los usuarios. Estos valores pueden ser complementados, pero no anulados, por valores específicos para cada usuario.

Una vez que se termina de modificar el archivo de configuración nessusd, es posible iniciar el demonio e iniciar el cliente gráfico Nessus. Sin embargo, para continuar se debe iniciar una sesión en el demonio Nessus que por defecto se ejecuta en el equipo local, en el puerto 1241.

La primera vez que se ejecuta Nessus, probablemente debemos deshabilitar todas las selecciones de la pestaña Nessus Plugins y recorrer cada categoría, para conocer exactamente lo que hace cada comprobación (complemento). Nessus divide los complementos en grupos o categorías. En la tabla 3.3 se muestran algunas categorías y breves descripciones de las mismas.

Tabla 3.3 Complementos de Nessus por categorías

Categoría de complemento Nessus	Descripción
Backdoors	Comprueba la existencia de puertas traseras como Trinity, Netbus, Back Orifice, SubSeven y similares, además de infecciones como CodeRed y Bugbear.
Brute Force Attacks	Busca credenciales comunes en servicios basados en autenticación.
CGI Abuses	Busca vulnerabilidades CGI para servidores Web y aplicaciones como IIS, Lotus, Domino, *Apache*, PHP, Cold Fusion, FrontPage y más.
CGI Abuses:XSS	Realiza ataques de secuencia de comandos entre sitios (XSS) sobre una aplicación web.
Denial-of-Service	Comprueba las vulnerabilidades ante DoS para varias aplicaciones y servicios de Unix y Windows.
Firewall **Gain a shell remotely**	Aprovecha esos servicios basados en red objetivo con vulnerabilidades de desbordamiento de búfer.
Gain root remotely	Aprovecha esos archivos locales o servicios objetivo con vulnerabilidades de desbordamiento de búfer.
Peer-to-peer File Sharing	Detecta la presencia de utilidades para compartir archives que se estén ejecutando como LimeWire, Trillian, Kazaa, ICQ, etc.
Remote File Access	Comprueba la existencia de métodos no autorizados de obtener archivos mediante servicios como NFS (Sistemas de archivos de red), TFTP (protocolo intrascendente para el traspaso de archivos), HTTP (protocolo para la transferencia de hipertexto), y Napster, además de bases de datos a las que se puede acceder y no están bien protegidas, como MySQL y PostgreSQL.
Service Detection	Identifica el tipo y versión de los servicios que se están ejecutando en un puerto.

Web Servers	Comprueba la existencia de vulnerabilidades y aplicaciones obsoletas que estén relacionadas con servidores Web, como IIS o *Apache*.

Nessus resume los problemas que encuentra y divide los puertos y problemas según el equipo. Además ordena las entidades objetivo por la gravedad de sus descubrimientos. El icono circular en rojo indica que existe al menos un agujero de seguridad, el icono de advertencia triangular naranja indica que existe al menos un peligro para la seguridad y el icono del foco indica que existe al menos una nota sobre la seguridad. Si no aparece ningún icono es porque el puerto está abierto, pero no se ha encontrado nada anómalo.

3.4 HERRAMIENTAS PARA WEB

La seguridad de un servidor Web se puede dividir en dos grandes categorías: probar en el servidor las vulnerabilidades más comunes y probar la aplicación Web. Antes de conectar un servidor Web a Internet, debería estar configurado según la siguiente lista:

- Configuración de red segura: un *firewall* u otro dispositivo limita el tráfico entrante para que llegue sólo a los puertos necesarios (probablemente, los puertos 80 y 443).
- Configuración de equipo seguro: el sistema operativo tiene parches de seguridad actualizados, se ha activado la auditoría y sólo los administradores pueden acceder al sistema.
- Configuración de servidor Web seguro: se ha revisado al configuración predeterminada del servidor Web, se han eliminado los archivos de muestra y el servidor se ejecuta en una cuenta se usuario restringida.

Por supuesto, una lista tan corta no cubre los detalles específicos de una combinación *Apache*/PHP o los detalles de todas las opciones de instalación de Servicios de información de Internet (IIS), pero sirve como base para una política de creación se servidores Web seguros. Se recomienda un rastreador de vulnerabilidades para verificar la política de instalación y de la seguridad de la aplicación Web.

Rastreadores de vulnerabilidades

Un rastreador de vulnerabilidades Web consiste, básicamente, en un motor de rastreo y un catálogo que contiene una lista de archivos comunes, archivos con vulnerabilidades conocidas y métodos para aprovechar vulnerabilidades para un grupo de servidores.

Un rastreador de vulnerabilidades, por ejemplo, busca archivos de copia de seguridad (como cambiar el nombre de default.asp por default.asp.bak) o intenta aprovechar vulnerabilidades de salto de directorio. El motor de rastreo utiliza la lógica para leer el catálogo de vulnerabilidades, enviando las solicitudes al servidor Web e interpretando las solicitudes para determinar si el servidor es vulnerable. Estas herramientas tienen como objetivo vulnerabilidades que se pueden solucionar fácilmente con configuraciones de equipo seguras, parches de seguridad actualizados, y una raíz de documentos Web limpia.

Ejemplos de estos rastreadores de vulnerabilidades Web son, Paros Proxy, WebScarab, Whisker y Nikto. A continuación se describe Nikto, un escáner de vulnerabilidades muy completo y funciona sobre una gran variedad de software Web.

3.4.1 Nikto[13]

Nikto de Chris Sullo, se basa en la biblioteca LibWhisker, es una herramienta compatible con capa de conexión segura (SSL), proxies y rastreos de puertos. Debido a que Nikto es un rastreador basado en *Perl*, funciona en Unix, Windows y Mac OS X. Usa bibliotecas estándar, que se incluyen en las instalaciones *Perl* predeterminadas.

Al iniciar Nikto, se debe especificar un equipo objetivo con la opción –h. A medida que el motor descubre posibles vulnerabilidades, aparecerán notas en el resultado, para explicar por qué algo descubierto puede suponer un riesgo para la seguridad.

En la tabla 3.4 se muestran las opciones básicas para ejecutar Nikto. Las opciones más importantes son establecer el equipo objetivo, el puerto objetivo y el archivo de resultados. Nikto acepta el primer carácter de una opción como si fuera toda la palabra. Por ejemplo, podemos especificar –s o –ssl para usar el protocolo HTTPS, o podemos especificar –w o -web par que el resultado tenga formato HTML.

Tabla 3.4 Opciones básicas de Nikto

Opción	Descripción
-host	Especifica un solo equipo. Nikto no acepta archivos con nombre de equipo, como la opción –H de whisker.
-port	Especifica un puerto al azar. Especificar el puerto 443 no implica necesariamente HTTPS. Debemos acordarnos de incluir –ssl.
-verbose	Resultado detallado. No podemos abreviar este modificador (-v está reservado para la opción de equipos virtuales).

-ssl	Activa la compatibilidad con SSL. Nikto no presupone HTTPS si especificamos el puerto 443.
-generic	Indica a Nikto que no tenga en cuenta el marcador del servidor y ejecuta un rastreo usando toda la base de datos.
-Format	Muestra el resultado con el formato HTML, CSV o texto. Debemos combinarlo con –output. -F htm -F csv -F txt
-output	Guarda el resultado en un archivo. Por ejemplo, -output nikto80_website.html –F htm.
-id	Proporciona credenciales de autenticación básica HTTP. Por ejemplo, -id nombreusuariocontraseña.
-vhost	Usa un equipo virtual como servidor Web objetivo, en lugar de una dirección *IP*. Esto afecta al contenido del encabezado de HTTP Host. Es importante usar esta opción en entornos de servidores compartidos.
-Cgidirs	Examina todos los directorios CGI posibles. Esto omite los errores 404 que nikto recibe para el directorio base.
-evasion	Técnicas de evasión de IDS. Nikto puede usar nueve técnicas diferentes para dar formato a la solicitud de URL, para intentar sobrepasar los sistemas de detección de intrusos menos sofisticados, basados en comparación de cadenas.

Firmas comunes

Los archivos de registros son dispositivos de seguridad reaccionarios, lo que significa que, si vemos una firma de ataque en el archivo, sabremos que ya hemos sido atacados. Si el ataque ha puesto en peligro el servidor, el primer sitio al que se debe acudir para recrear el suceso son los registros Web. Los registros ayudan a los administradores y a los programadores a encontrar fallos o páginas erróneas en un sitio Web (son necesarios para mantener un servidor Web estable). Por lo anterior es necesario tener una política para activar en el servidor Web los registros, la recolección de archivos de registro y la revisión y guarda de los mismos.

3.5 RASTREADORES

Estas herramientas nos permiten conocer el tipo de tráfico de la red, los protocolos utilizados y los servicios que se ejecutan, entre otras cosas. Los datos que se obtienen a través de estos análisis permiten al administrador de una red conocer la carga que significa un servicio en el sistema, el tipo de conexiones que se realizan en la red y los protocolos utilizados. Estos rastreadores generalmente son conocidos como sniffers de red.

Un sniffer es una aplicación que monitorea, filtra y captura paquetes de datos transmitidos por una red[14].

3.5.1 Wireshark[15]

Este proyecto surgió en 1997 con Gerald Combs a la cabeza con el nombre "Ethereal", por varios años este sniffer de red se hizo muy popular debido a las ventajas que ofrecía como herramienta de línea de comandos y una interfaz gráfica cómoda, la posibilidad de analizar múltiples protocolos y generar estadísticas. En el año 2006 el proyecto cambió de nombre y de compañía. El nuevo nombre es "Wireshark" y se sigue desarrollando bajo este nombre junto con la librería pcap.

Wireshark es un *sniffer* de red, que ofrece todas las ventajas de una herramienta de línea de comandos como tcpdump con varias ventajas. Tiene una interfaz gráfica cómoda para el usuario por lo que no tenemos que tratar de aprender todos los parámetros de línea de comandos. También ofrece opciones más analíticas y estadísticas. Otras ventajas de Wireshark son:

- Formato de salida más limpio.
- Se admiten muchos más formatos de protocolo.
- Se admiten muchos más formatos de red física, incluyendo los protocolos más modernos.
- Los datos de red capturados pueden explorarse y ordenarse interactivamente.
- Tiene un modo de filtro de presentación que incluye la capacidad de resaltar determinados paquetes en color.
- La capacidad de seguir un flujo *TCP* y ver el contenido en ASCII, una opción que puede ser muy valiosa cuando necesitamos leer mensajes entre servidores para localizar problemas Web o de correo electrónico.
- La capacidad de trabajar con varios programas y bibliotecas de captura.
- La capacidad de guardar sesiones en múltiples formatos.
- Un modo de línea de comandos para aquellos a los que no les gusta la interfaz gráfica.

Wireshark presenta la información detallada sobre los paquetes capturados ordenados de manera muy parecida al modelo *OSI*, en la tabla 3.5, Datos del flujo de paquetes, podemos ver la información que Wireshark despliega sobre el flujo de paquetes:

Tabla 3.5 Datos del flujo de paquetes

Elemento	Descripción
Packet Number	Asignado por wireshark
Time	La hora en que se ha recibido el paquete, establecida a partir del tiempo transcurrido desde el inicio de la sesión de captura.
Source address	De dónde proviene el paquete. Es una dirección *IP* sobre redes *IP*.
Destination address	Hacia dónde se dirige el paquete, normalmente también una dirección *IP*.
Protocol	El protocolo de nivel 4 que está utilizando el paquete.
Info	Alguna información de resumen sobre el paquete, normalmente un campo de tipo.

La última sección es el contenido real del paquete, tanto en hexadecimales como su traducción a ASCII, cuando es posible. Los archivos binarios siguen teniendo una apariencia desordenada así como el tráfico cifrado, pero todo lo que tenga un texto claro aparecerá, lo que resalta la eficacia y el peligro de tener un *sniffer* en la red.

Existen muchas opciones y filtros que se pueden establecer. Se comienza ejecutando una sesión de captura muy abierta y con las opciones mostradas en la tabla 3.6, Opciones y filtros de Wireshark:

Tabla 3.6 Opciones y filtros de Wireshark

Opción	Descripción
Interface	Wireshark detecta automáticamente todas las interfaces y presenta una lista de ellas. También puede elegir una captura de todas las interfaces a la vez.
Limit each packet	Establece el tamaño máximo para los paquetes capturados.

to x bytes	
Capture packets in promiscuous mode	Esta opción esta activada de forma predeterminada. Se debe desactivar si se desea capturar tráfico sólo en la máquina *sniffer*.
Filter	Permite crear un filtro.
Capture file(s)	Esta opción se activa si se desea leer desde un archivo en lugar de capturar datos activos.
Display options	Estas opciones están deshabilitadas de forma predeterminada pero pueden habilitarse si deseamos ver los paquetes desplazarse en tiempo real.
Capture limits	Existen otras opciones sobre cuándo debe finalizar la sesión. Aparte de detenerla manualmente se puede configurar para que Wireshark se detenga cuando haya capturado x número de paquetes o kilobytes de datos o una vez transcurrido x segundos de tiempo.
Name resolution	Se puede especificar si queremos que Wireshark resuelva nombres en diversos niveles del modelo de red. Puede resolver selectivamente nombres de direcciones MAC, nombres de red y/o nombres de capas de transporte.

Al finalizar la captura y análisis de datos, se pueden guardar en distintos formatos, incluyendo captura de tráfico libpcap, SunSnoop, LANalyser, Sinfeer, Microsoft Network Monitor y Visual Networks.

3.6 HERRAMIENTAS PARA AUDITAR Y DEFENDER LA RED

Básicamente se tienen dos dispositivos para controlar el acceso a la red y para defenderla desde dentro hacia afuera. El *firewall* permite controlar los accesos y el Sistema de Detección de Intrusos, que alerta sobre incursiones no autorizadas al sistema de manera que se pueda crear una respuesta efectiva ante tales incursiones.

3.6.1 El *firewall*

En la mayoría de los sistemas Linux versión de núcleo 2.4 o superiores se ha incorporado una utilidad de filtrado de paquetes llamada *Iptables* que permite crear un *firewall* empleando comandos propios del sistema operativo. *Iptables* es una herramienta compleja y normalmente se recomienda para usuarios que están familiarizados con los *firewalls* y la tarea de configurarlos.

La idea que se esconde detrás de *Iptables* e Ipchains es crear canales de entradas y procesarlas de acuerdo con un conjunto de reglas (la configuración del *firewall*) y enviarlas a continuación en canales de salida. En *Iptables*, estos canales se denominan tablas; en Ipchains se denominan cadenas. Las tablas básicas empleadas en *Iptables* son:

Input (entrada)

Forward (envío)

Prerouting (enrutamiento previo)

Postrouting (enrutamiento posterior)

Output (salida)

El formato que genera una declaración *Iptables* es:

iptables comando especificación_de_regla extensiones

A continuación se muestra la tabla 3.7 con un resumen de los comandos *Iptables*.

Tabla 3.7 Comandos Iptables

Comando	Descripción
-A chain	Añade una o más reglas al final de la declaración.
-I chain rulenum	Inserta una cadena en la ubicación rulenum. Es útil cuando deseamos que una regla reemplace a las anteriores.
-D chain	Elimina la cadena indicada.
-R chain rulenum	Reemplaza la regla en la ubicación rulenum con la chain proporcionada.
-L	Lista todas las reglas en al cadena actual.
-F	Purga todas las reglas en la cadena actual, eliminando básicamente la configuración de nuestro *firewall*. Es recomendable su uso cuando se inicia una configuración para asegurar que ninguna regla existente entrará en conflicto con una regla nueva.
-Z chain	Pone a cero todas las cuentas de paquetes y bytes en la cadena denominada.

-N chain	Crea una nueva cadena con el nombre de chain.
-X chain	Elimina la cadena especificada. Si no se especifica ninguna cadena, se eliminan todas las cadenas.
-P chain policy	Establece la política para la cadena especificada en policy.

Entre los firewalls de libre distribución se encuentran Turtle Firewall, Netfilter e IP Filter.

Turtle *Firewall*[16]

Turtle *Firewall* es un programa de libre distribución creado por Andrea Frigido. Se trata de un conjunto de secuencias de comandos *Perl* que hacen todo el trabajo de configurar un *firewall* Iptables automáticamente. Este programa facilita la observación de las reglas para estar seguros de obtener las declaraciones en el orden correcto.

No se requiere inicializar el *firewall* con una secuencia de comandos shell ya que se ejecuta como un servicio. Turtle *Firewall* utiliza el servicio Linux Webmin, un pequeño servidor Web que permite efectuar cambios de configuración en el servidor a través de un explorador web. Aunque esta tarea puede introducir alguna inseguridad en el sistema al ejecutar un servidor web sobre el *firewall*, puede merecer la pena el riesgo por la facilidad de configuración que conlleva. Muchos suministradores comerciales utilizan ahora una interfaz de explorador web para la configuración. Una extraordinaria ventaja obtenida a partir de esta aplicación es que se puede acceder a la pantalla de configuración desde cualquier máquina Windows o Unix.

Turtle *Firewall* utiliza el concepto de zonas para definir las redes de confianza y las de no confianza. Una zona de confianza conecta a una red con empleados o personas en los que generalmente se puede confiar, como una red interna. Una zona de no confianza es una red que puede tener cualquier cosa, desde empleados a clientes o suministradores o cualquier otra persona con malas intenciones. Turtle los denomina “good” (buenos) y “bad” (malos) pero básicamente es lo mismo que de confianza y de no confianza. Además tiene una entrada para un segmento *DMZ*, que se utiliza para los servidores que necesitan acceder sin restricciones a la zona de no confianza.

Se puede implementar la función Iptables Masquerade (máscaras de Iptables) usando direcciones *IP* privadas para la LAN interna, definiendo la zona que se va a enmascarar, normalmente será nuestra interfaz de confianza o “good”. De igual manera, se pueden establecer anfitriones para NAT. Al establecer el anfitrión como *IP* virtual, actúa en el frente como el anfitrión real, y el *firewall* enviará todos los paquetes a través del anfitrión

virtual hasta el anfitrión real. Así se proporciona un nivel adicional de protección a los servidores internos.

3.6.2 Sistema de detección de intrusos

Las técnicas de detección de intrusos consisten en analizar y determinar las actividades anómalas, incorrectas y, por supuesto, ilegales a las que puede estar sometido un sistema o una red[4]. Para poder descubrir todo este tipo de acciones se pueden utilizar técnicas diferentes, como el análisis del tráfico de paquetes, análisis de las firmas de los ataques, comprobación de integridad de los archivos y el análisis estadístico.

Un Sistema de Detección de Intrusos (SDI) protege contra los ataques que pasan a través del *firewall* hasta la red local interna. Los *firewalls* pueden estar mal configurados, lo que permite la introducción de tráfico no deseado en la red. Aún si funcionan correctamente, los *firewalls* normalmente dejan algún tráfico de aplicación que puede ser peligroso. Un SDI puede comprobar el tráfico entrante y marcar los paquetes potencialmente peligrosos, si está configurado correctamente puede hacer una doble comprobación de las reglas del *firewall* y proporcionar una protección adicional para los servidores de aplicación.

Aunque es útil una protección contra ataques externos, una de las ventajas principales de un Sistema de Detección de Intrusos es que permite detectar los ataques y la actividad sospechosa de orígenes internos. Un *firewall* protege de muchos ataques externos; sin embargo, una vez que el atacante se encuentra en la red local, un *firewall* puede hacer muy poco, sólo puede ver el tráfico que atraviesa desde el exterior. Los *firewalls* son en general ciegos a la actividad de la red interna.

Los sistemas de detección normalmente ofrecen a los administradores varios métodos diferentes de notificación de una alerta, en su nivel más básico, las alertas pueden simplemente enviarse a un archivo de registro para una revisión posterior, algo no muy recomendable ya que requiere que el administrador revise los registros. Para estar al tanto de los intentos de intrusión, se deben supervisar diariamente estos registros, de otra manera pueden pasar días o semanas antes de que se descubra la intrusión. La otra alternativa es enviar un mensaje de correo electrónico o una página a la persona apropiada siempre que se genere un alerta. Sin embargo, puede ser molesto recibir páginas varias veces al día. Asimismo, las alertas de correo electrónico no estarán en un formato en el que se puedan comparar con las alertas pasadas o analizadas de otra forma. La mejor solución para controlar las alertas es portarlas inmediatamente a una base de datos que permita un análisis más profundo.

Ejemplos de sistemas de detección de intrusos son Snort, Sguil y OSSEC HIDS (Open Source Host-based Intrusion Detection System).

Snort[17]

Snort es un Sistema de Detección de Intrusos desarrollado por Martin Roesch, aunque este proyecto ha crecido demasiado y ahora cuenta con 30 desarrolladores más en su equipo principal sin contar con los que escriben reglas y otras partes del software. Existen muchos recursos disponibles para Snort y todos ellos son recursos "online" gratuitos.

Snort es principalmente un SDI basado en firmas, aunque con la adición del módulo Spade, ahora puede realizar una detección de actividad de anomalías.

Opciones únicas de Snort

Libre distribución: Snort es de libre distribución y portable a casi cualquier sistema operativo tipo Unix. También existen versiones disponibles para Windows y otros sistemas operativos.

Ligero: Debido a que el código se ejecuta de una forma tan eficiente, no se requiere mucho hardware para ejecutar Snort, lo que permite que pueda analizar tráfico en una red de 100Mbps a una velocidad cercana al cable.

Snort personaliza reglas: Snort ofrece una forma fácil para ampliar y personalizar el programa brindando la posibilidad de escribir reglas o firmas propias.

Snort se ejecuta desde la línea de comandos. Puede ejecutarse de tres modos diferentes: *sniffer* de paquetes, registro de paquetes y como SDI.

Modo *sniffer* de paquetes

Snort actúa como un *sniffer*, mostrando el contenido sin filtrar en el cable. Evidentemente, si lo único que necesitamos es un sniffer se puede usar un programa dedicado a este propósito como Tcpdump o Wireshark. Sin embargo, el modo *sniffer* de paquetes es bueno para asegurarse de que todo funciona correctamente y Snort está viendo paquetes.

Modo de registro de paquetes

Modo similar al de *sniffer* de paquetes, pero nos permite registrar paquetes "olfateados" al disco para su utilización y análisis futuros. Snort registra paquetes por dirección *IP* y crea un directorio independiente para cada *IP* registrado. También se puede utilizar Snort con la opción –b para registrar todos los datos en un solo archivo binario apropiado para su lectura posterior con un *sniffer* de paquete como Ethereal, Wireshark o Tcpdump.

Modo de detección de intrusos

En este modo Snort registra paquetes sospechosos o que merecen una consideración posterior. Sólo se necesita el modificador adicional en línea de comandos "-c configfile", que le indica a Snort que utilice un archivo de configuración para dirigir los paquetes que registra. El archivo de configuración determina todas las configuraciones para Snort y es un archivo muy importante. Snort incluye un archivo de configuración predeterminado.

La siguiente tabla (tabla 3.8) muestra las opciones de modos de alerta de Snort.

Tabla 3.8 Opciones de modos de alerta de Snort

Opciones	Descripción
-A full	Información completa de la alerta, incluyendo datos de aplicación. Es el modo predeterminado de alerta y se utilizará cuando no se especifique nada.
-A fast	Modo rápido. Registra sólo la información de encabezado del paquete y el tipo de alerta. Es útil para redes muy rápidas pero si se necesita más información forense hay que utilizar el modificador full.
-A unsock	Envía la alerta a un número de zócalo Unix que otro programa puede estar escuchando.
-A none	Desactiva las alertas.

Snort activa algunas reglas en su configuración base, que se deben deshabilitar en caso de que no se apliquen a nuestro sistema. En el directorio de instalación de Snort se encuentran los archivos de reglas, archivos con la extensión .rules. Cada uno de ellos contiene muchas reglas agrupadas por categoría. Se puede deshabilitar toda una clase de reglas comentándola en el archivo de configuración o se pueden deshabilitar reglas individuales si se quiere la protección de otras reglas de la clase. Para comentar una regla, se busca en los archivos .rules apropiados y se inserta el signo # delante de la línea de dicha regla. Normalmente es mejor deshabilitar una sola regla que toda una clase de reglas, a no ser que ésta no se aplique a la configuración deseada.

BASE[18]

BASE (Basic Analysis and Security Engine), el Motor de Seguridad de Análisis Básico, es una aplicación basada en ACID (Analysis Console for Intrusion Databases) la consola de análisis para bases de datos de intrusión.

Se trata de un programa diseñado para hacer un mejor uso de los dispositivos de detección de intrusiones. En sus inicios fue desarrollado para el programa AirCERT llevado a cabo por la Carnegie Mellon University, que forma parte de la organización CERT (Computer Emergency Response Team), es decir, el equipo para respuestas informáticas de emergencia. El CERT registra los incidentes del crimen informático y envía noticias a una lista de correo siempre que se produce un incidente importante. La lista de correos del CERT es el primer sistema de avisos para cualquier estallido o ataque que se esté produciendo en Internet y, como tal, puede ser muy útil para un administrador de sistemas.

La idea básica de BASE es trasladar todos los datos de detección de intrusión a una base de datos donde se pueden ordenar y organizar por prioridades. BASE proporciona un panel de control basado en la Web para visualizar y manipular estos resultados; utiliza una base de datos SQL y un servidor Web y admite múltiples sensores para los datos de entrada, también acepta alertas Snort y archivo de registros ajustados a los registros del sistema.

BASE es una aplicación Web hecha en PHP que permite analizar los logs y las alertas generadas por Snort de una forma muy amigable.

Entre las facilidades que ofrece BASE se encuentran las siguientes:

- La opción de hacer búsquedas de alertas específicas.
- Observar de una forma gráfica los paquetes capturados.
- Generación de gráficas estadísticas.

Este programa utiliza un sistema basado en la autenticación de usuarios y la definición de roles, de modo que el administrador de seguridad puede decidir qué información de cada usuario ver, además de tener una configuración simple y fácil de usar, basada en la Web.

En el siguiente capítulo se realiza la instalación y la configuración de los programas de *software libre* y las librerías necesarias para tener un sistema de defensa que permita auditar, prevenir y detectar incidentes, así como investigar y responder a los mismos de una manera aceptable y segura.

CAPÍTULO 4
CONFIGURACIÓN E IMPLEMENTACIÓN DE LAS HERRAMIENTAS DE DEFENSA

CONFIGURACIÓN E IMPLEMENTACIÓN DE LAS HERRAMIENTAS DE DEFENSA

En este capítulo se lleva a cabo la instalación y configuración de las herramientas de defensa para la red, se describe paso a paso el proceso de instalación y se hace una breve descripción de las configuraciones realizadas a cada herramienta y en qué área se va a implentar.

4.1 HERRAMIENTAS PARA REFORZAR EL SISTEMA OPERATIVO

4.1.1 *OpenSSH*

OpenSSH es una herramienta que se implementa en la mayoría de los sistemas operativos basados en Linux por lo que únicamente se mostrarán características básicas sobre su configuración.

El archivo de configuración del servidor se llama sshd_config y generalmente se encuentra en el directorio /etc/ssh. Para obtener un mejor rendimiento de SSH este archivo que se debe modificar.

Especificamos el protocolo a utilizar:

Protocol 2

Cambiamos el número de puerto estándar (22) por un puerto diferente (opcional):

Port 2220

No se permitirá el logueo del administrador directamente, ya que se puede cambiar de usuario una vez que se ingresa como usuario normal:

PermitRootLogin no

El número máximo de equivocaciones al ingresar el usuario y/o la contraseña se cambiará a dos:

MaxAuthTries 2

La cantidad de pantallas de login o cantidad de conexiones simultáneas de login que permitirá el sshd por ip se establecerá en 3:

MaxStartups 3

El servidor desconecta al cliente si no se ha logueado correctamente en 2 minutos:

LoginGraceTime 120

No usaremos el método de autenticación por rhosts:

RhostsAuthentication no

IgnoreRhosts yes

Establecemos que la conexión termine después de 3 minutos, para lo cual cada 60 segundos el servidor enviará un mensaje esperando respuesta del cliente, y el número de intentos para obtener respuesta será 3:

ClientAliveInterval 60

ClientAliveCountMax 3

No se permitirán passwords vacíos:

PermitEmptyPasswords no

4.1.2 BASTILLE LINUX

Para llevar a cabo el proceso correspondiente se debe abrir una interfaz de línea de comandos en la que se introducen los siguientes comandos:

yum install perl-Curses*

yum install perl-Tk*

tar jvxf Bastile-x-x.tar.bz2

cd Bastille

sh Install.sh

bastille –x

accept

bastille -x

Usaremos Bastille para aplicar actualizaciones para los agujeros de seguridad conocidos y para restringir y/o desactivar los servicios innecesarios en el sistema.

Al iniciar Bastille aparece la interfaz de configuración y una serie de preguntas con una explicación sobre cada área que se va a configurar, con una posible respuesta, SI o NO, como se muestra en la figura 4.1, Interfaz de configuración de Bastille.

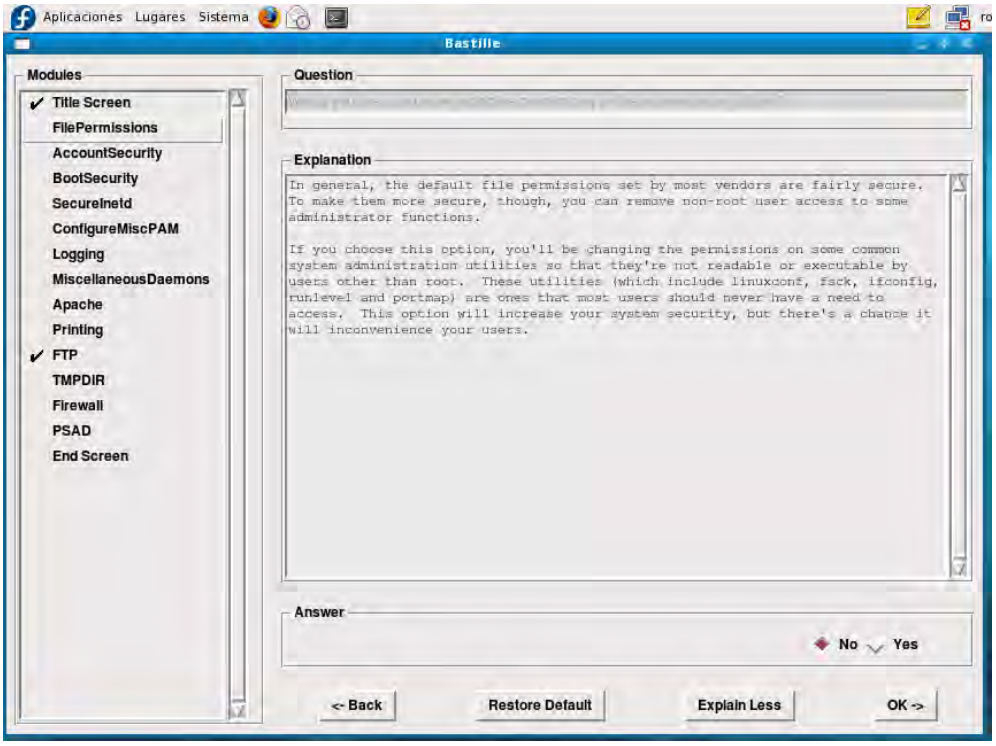


Figura 4.1, Interfaz de configuración de Bastille

- ***Would you like to set more restrictive permissions on the administration utilities? [N]***

  Útil en máquinas con múltiples cuentas de usuario pero en este caso se trata de una máquina dedicada para establecer el sistema de seguridad.

- ***Would you like to disable SUID status for mount/umount? [Y]***
- ***Would you like to disable SUID status for ping? [Y]***
- ***Would you like to disable SUID status for at? [Y]***

  Se deshabilitarán los programas SUID para evitar que usuarios sin privilegios ejecuten programas con permisos de administrador.

- ***Should Bastille disable clear-text r-protocols that use IP-based authentication? [Y]***

r-tools son un conjunto de utilidades para la administración remota de equipos que usan las direcciones *IP* como método de autenticación y no usan encriptación para el intercambio de información.

- ***Would you like to enforce password aging? [Y]***

  Se habilitará un tiempo de caducidad para las contraseñas, por default 180 días. Se deberá cambiar la contraseña antes de este plazo, de otra forma la cuenta será bloqueada.

- ***Would you like to restrict the use of cron to administrative accounts? [Y]***

  El uso de cron estará restringido sólo para la cuenta de administrador.

- ***Do you want to set the default umask? [Y]***

  El umask son los permisos por defecto que se le ponen a los archivos que se van creando.

- ***What umask would you like to set for users on the system? [077]***

  Continuación de la pregunta anterior, lo mejor es usar la opción 077 para que únicamente el dueño de los archivos pueda escribir o leer sobre ellos.

- ***Should we disallow root login on all ttys? [N]***

  Esta opción es extremadamente útil con equipos a los que se pueda acceder por SSH sin limitación en cuanto a la *IP* de origen. En este caso ya se limitó el acceso del administrador en el archivo de configuración de *OpenSSH*.

- ***Would you like to password-protect the GRUB prompt? [Y]***

  Si se tiene acceso físico al equipo cualquier persona podría reiniciar el equipo y obtener una consola de administrador pasándole ciertos parámetros al GRUB, por lo que se protegerá con una contraseña.

- ***Would you like to disable CTRL-ALT-DELETE rebooting? [N]***

  No se recomienda modificar este parámetro ya que si se tiene acceso físico al equipo también se puede tener un reinicio forzado del mismo.

- ***Would you like to password protect single-user mode? [Y]***

Este modo (mono-usuario) permite arrancar el sistema de manera que solamente puede acceder el administrador del equipo. Generalmente no solicita autenticación por lo que protegeremos este modo mediante una contraseña.

- ***Would you like to set a default-deny on TCP Wrappers and xinetd? [N]***

  Se permitirá que se ejecuten estos servicios de conectividad a internet.

- ***Should Bastille ensure the telnet service does not run on this system? [y]***

  Se desactivará el servicio telnet ya que es obsoleto y supone un riesgo grave de seguridad del sistema al transmitir datos sin cifrarlos.

- ***Should Bastille ensure inetd's FTP service does not run on this system? [y]***

  Lo mismo que telnet, ftp es un servicio inseguro, por lo que se deshabilitará.

- ***Would you like to display "Authorized Use" messages at log-in time? [N]***

  No se mostrará ningún mensaje al momento de loguearse. Esta opción se puede activar para mostrar un mensaje con las restricciones para uso del sistema.

- ***Would you like to disable the gcc compiler? [N]***

  Se mantendrá activo el compilador de C ya que podría usarse para desarrollo de aplicaciones o scripts.

- ***Would you like to put limits on system resource usage? [Y]***

  Con esta medida establecemos un límite de 150 procesos por usuario, suficiente para trabajar y evita un ataque por denegación de servicio.

AL DAR CLICK EN OK => System rosource limits have been set in the file /etc/security/limits.conf,which you can edit later as necessary.

- ***Should we restrict console access to a small group of user accounts? [N]***

  Permite denegar el acceso a la consola excepto a un grupo determinado de cuentas.

- ***Would you like to disable printing? [Y]***

  El dejar activo un servicio que no se utilizará supone un riesgo de seguridad.

- ***Would you like to run the packet filtering script? [N]***

Esta opción activaría el *firewall* nativo de Linux, en este caso no se activará porque posteriormente se implementa un *firewall* con la herramienta Turtle *Firewall*.

- ***Are you finished answering the questions, i.e. may we make the changes? [Y]***

  Respondemos afirmativamente para que se apliquen los cambios en el sistema.

Como se aprecia Bastille es una herramienta muy útil para el reforzamiento de un sistema operativo, que si bien no está muy actualizada, tiene como ventaja el poder realizar el reforzamiento de un sistema operativo tipo Linux de una manera sencilla y eficiente, con una interfaz intuitiva y esto permite conocer los puntos más importantes a reforzar en un sistema operativo.

4.2 PROTECCIÓN DE CONTRASEÑAS Y ATAQUES DE FUERZA BRUTA

4.2.1 John The Ripper

John the Ripper es una herramienta de mucha importancia para la administración de sistemas ya que nos permite encontrar de manera rápida y versátil las contraseñas débiles que se crean en el sistema.

Para la instalación de John the Ripper, únicamente se requiere el programa. A continuación se muestra la secuencia de comandos necesarios para realizar su instalación:

tar -zxvf john-1.7.5.tar.gz

cd john-1.7.5

cd src

make

make clean linux-x86-any

cd ../run

Ahora ya está instalado John the ripper, con la opción -- test se prueba el funcionamiento de dicho programa.

```
[root@localhost run]# ./john --test
Benchmarking: Traditional DES [24/32 4K]... DONE
Many salts:         77849 c/s real, 153852 c/s virtual
Only one salt:      61465 c/s real, 150650 c/s virtual
Benchmarking: BSDI DES (x725) [24/32 4K]... DONE
Many salts:         2785 c/s real, 5336 c/s virtual
Only one salt:      2798 c/s real, 5182 c/s virtual
```

```
Benchmarking: FreeBSD MD5 [32/32]... DONE
Raw:    2343 c/s real, 4922 c/s virtual
Benchmarking: OpenBSD Blowfish (x32) [32/32]... DONE
Raw:    145 c/s real, 382 c/s virtual
Benchmarking: Kerberos AFS DES [24/32 4K]... DONE
Short:  57958 c/s real, 139323 c/s virtual
Long:   181035 c/s real, 449900 c/s virtual
Benchmarking: LM DES [32/32 BS]... DONE
Raw:    1605K c/s real, 3536K c/s virtual
```

4.2.2 Cracklib

Cracklib es una biblioteca de comprobación de contraseñas y forma parte de la instalación predeterminada de las distribuciones Debian, Mandrake, RedHat y SuSE. Cracklib permite a los administradores de sistemas establecer reglas para la creación de contraseñas.

Para realizar la configuración se debe editar uno de los dos posibles archivos: /etc/pam.conf o /etc/pam.d/passwd

password required /lib/security/pam_cracklib.so minlen=10 dcredit=2 ocredit=2

password required /lib/security/pam_unix.so nullok use_authok md5

La primera columna de ambas instrucciones actualiza la información de autenticación, como cambiar la contraseña de un usuario, lo que le permite a dicho usuario el acceso al sistema de seguridad que controla las credenciales.

La siguiente columna determina el control de un servicio, o cómo debe administrarse su ejecución. El argumento “required” indica que si el servicio falla, las siguientes acciones se procesarán pero fallarán.

minlen=N, es la longitud mínima de la contraseña, igual a la cantidad de créditos, que deben obtenerse. Un crédito por unidad de longitud. La longitud rel de la nueva contraseña nunca puede ser menor que 6.

dcredit=N, indica la cantidad máxima de créditos por incluir dígitos (0-9).

ocredit=N, indica la cantidad máxima de créditos por incluir caracteres que no son letras ni números.

Use_authok se utiliza para apilar módulos en un servicio. En este caso, se añadió *md5* a la biblioteca pam_unix.so. Esto permite que las contraseñas se codifiquen con el algoritmo *MD5*.

4.3 SEGURIDAD DEL EQUIPO

4.3.1 Nessus

Para llevar a cabo la instalación de Nessus se recomienda seguir las indicaciones que se dan a continuación, en las cuales se presentan los comandos correspondientes y la respuesta del sistema sombreada para su fácil distinción:

Se descarga el archivo fuente de Nessus.
Se desempaqueta y descomprime el archivo, se ingresa al directorio y se ejecuta el archivo binario de instalación:

tar –zxvf Nessus-4.0.2.tar.gz
cd Nessus-4.0.2
./install.sh

This installation program will install or upgrade Nessus under /opt/nessus
Note that you're attempting to install the GENERIC version of Nessus 4
This setup is not supported in production. Try to use a specific RPM instead

Press [ENTER] to start the installation

[root@localhost Nessus-4.0.2]#
<enter>

.opt/nessus/var/nessus/nessus_org.pem
opt/nessus/var/nessus/users/
opt/nessus/var/nessus/nessus-services
opt/nessus/var/nessus/logs/
[Done]
- Please run /opt/nessus/sbin/nessus-adduser to add a user
- Register your Nessus scanner at http://www.nessus.org/register/ to obtain
  all the newest plugins
- You can start nessusd by typing /sbin/service nessusd start

Se crea un usuario para Nessus:

[root@localhost Nessus-4.0.2]# /opt/nessus/sbin/nessus-adduser

Login : lsdpo
Authentication (pass/cert) : [pass]   <enter>lo dejamos en blanco
Login password : berliner&&
Login password (again) : berliner&&
Do you want this user to be a Nessus 'admin' user ? (can upload plugins, etc...) (y/n) [n]: y
User rules
----
nessusd has a rules system which allows you to restrict the hosts that nessus_user has the right to test. For instance, you may want him to be able to scan his own host only.
Please see the nessus-adduser manual for the rules syntax
Enter the rules for this user, and enter a BLANK LINE once you are done :
(the user can have an empty rules set)

```
Login           : lsdpo
Password        : **********
This user will have 'admin' privileges within the Nessus server
Rules           :
Is that ok ? (y/n) [y]
User added
```

Ahora registramos el código de activación para usar los plugins de Nessus.
[root@localhost Nessus-4.0.2]# /opt/nessus/bin/nessus-fetch --register 475F-7F48-7D5B-240D-B222

```
Your activation code has been registered properly - thank you.
Now fetching the newest plugin set from plugins.nessus.org...
Your Nessus installation is now up-to-date.
If auto_update is set to 'yes' in nessusd.conf, Nessus will

update the plugins by itself.
```

Se debe crear ahora un certificado SSL para el servidor:
[root@localhost Nessus-4.0.2]# /opt/nessus/sbin/nessus-mkcert

```
-------------------------------------------------------------------------------
                        Creation of the Nessus SSL Certificate
-------------------------------------------------------------------------------
This script will now ask you the relevant information to create the SSL
certificate of Nessus. Note that this information will *NOT* be sent to
anybody (everything stays local), but anyone with the ability to connect to your
Nessus daemon will be able to retrieve this information.
CA certificate life time in days [1460]:
Server certificate life time in days [365]:
Your country (two letter code) [FR]: MX
Your state or province name [none]: DF
Your location (e.g. town) [Paris]: DF
Your organization [Nessus Users United]: UNAM
Congratulations. Your server certificate was properly created.
The following files were created :
Certification authority :
Certificate = /opt/nessus/com/nessus/CA/cacert.pem
 Private key = /opt/nessus/var/nessus/CA/cakey.pem
 Nessus Server :
 Certificate = /opt/nessus//com/nessus/CA/servercert.pem
 Private key = /opt/nessus//var/nessus/CA/serverkey.pem
```

Para que el servidor comience a usar los nuevos certificados, se deberán agregar al archivo de configuración (nessusd.conf) las siguientes líneas y luego reiniciar el servidor:

```
 echo "cert_file=/opt/nessus//com/nessus/CA/servercert.pem" >>
/opt/nessus/etc/nessus/nessusd.conf

echo "key_file=/opt/nessus//com/nessus/CA/serverkey.pem" >>
/opt/nessus/etc/nessus/nessusd.conf
```

echo "ca_file=/opt/nessus//com/nessus/CA/cacert.pem" >>
/opt/nessus/etc/nessus/nessusd.conf

echo "#force_pubkey_auth = yes" >> /opt/nessus/etc/nessus/nessusd.conf

La última línea (#force_pubkey_auth = yes) es una configuración que le indica al servidor que para permitir que cualquier usuario se conecte, deberá presentar sus certificados. En el último comando estamos pasando esa configuración pero de forma comentada (no se utilizará) lo que significa que los usuarios podrán conectarse al servidor de Nessus usando su clave normal sin necesidad de usar certificados, si se descomenta esa opción, los usuarios no solo deberán presentar su clave sino que también tendrán que usar sus certificados para que Nessus lo pueda autorizar.

Parar Nessus:

/sbin/service nessusd stop

killall nessus-service

Reiniciar Nessus para que cargue los plugins:
/opt/nessus/sbin/nessus-service -D

Ahora se crea el certificado del cliente:
[root@localhost Nessus-4.0.2]# /opt/nessus/sbin/nessus-mkcert-client

Do you want to register the users in the Nessus server as soon as you create their certificates ? [n]: y

-[root@Kakaroto software]# /opt/nessus/sbin/nessus-mkcert-client

Do you want to register the users in the Nessus server

as soon as you create their certificates ? [n]: y

Creation Nessus SSL client Certificate

This script will now ask you the relevant information to create the SSL
client certificates for Nessus.
Client certificate life time in days [365]:
Your country (two letter code) [FR]: MX
Your state or province name []: DF

Your location (e.g. town) [Paris]: DF
Your organization []: UNAM
Your organizational unit []: ING
****
We are going to ask you some question for each client certificate
If some question have a default answer, you can force an empty answer by

```
entering a single dot '.'
*********
User #1 name (e.g. Nessus username) []: lsdpo
User lsdpo already exist
Do you want to go on and overwite the credentials?  [y]: y^C
[root@Kakaroto software]# /opt/nessus/sbin/nessus-mkcert-client
Do you want to register the users in the Nessus server
as soon as you create their certificates ? [n]: y
-------------------------------------------------------------------------------
                        Creation Nessus SSL client Certificate
-------------------------------------------------------------------------------
This script will now ask you the relevant information to create the SSL
client certificates for Nessus.
Client certificate life time in days [365]:
Your country (two letter code) [FR]: MX
Your state or province name []: DF
Your location (e.g. town) [Paris]: DF
Your organization []: UNAM
Your organizational unit []: ING
*********
We are going to ask you some question for each client certificate
If some question have a default answer, you can force an empty answer by
entering a single dot '.'
*********
User #1 name (e.g. Nessus username) []: lsdpo
User lsdpo already exist
Do you want to go on and overwite the credentials?  [y]: y
Should this user be administrator?  [n]: y
Country (two letter code) [MX]:
State or province name [DF]:
Location (e.g. town) [DF]:
Organization [UNAM]:
Organizational unit [ING]:
e-mail []: lsdpo@yahoo.com.mx
User rules
----------
nessusd has a rules system which allows you to restrict the hosts
that $login has the right to test. For instance, you may want
him to be able to scan his own host only.
Please see the nessus-adduser(8) man page for the rules syntax
Enter the rules for this user, and enter a BLANK LINE once you are done:
(the user can have an empty rules set)
User added to Nessus.
Another client certificate?  [n]: n
Your client certificates are in /tmp/nessus-136e615c

You will have to copy them by hand
```

[root@localhost Nessus-4.0.2]# cp /tmp/nessus-136e615c/* /usr/local/software/certs/

Archivos creados

```
Certification authority :
/opt/nessus/com/nessus/CA/cacert.pem
/opt/nessus/var/nessus/CA/cakey.pem
Nessus Server :
 /opt/nessus/com/nessus/CA/servercert.pem
/opt/nessus/var/nessus/CA/serverkey.pem
```

Iniciar Nessus:

/opt/nessus/sbin/nessus-service -D

Para concluir el proceso Nessus:
killall nessus-service

Actualizamos los plugins:
/opt/nessus/sbin/nessus-update-plugins

Ahora se hace la instalación y configuración del cliente de Nessus:

cd /usr/local/software

rpm -Uvh NessusClient-4.0.2-fc10.i386.rpm

Se lanza el cliente de Nessus (figura 4.2, Cliente de Nessus).

/opt/nessus/bin/NessusClient

Figura 4.2, Cliente de Nessus

En esta parte se configura la conexión a través de SSL y mediante los certificados que creamos anteriormente, se hace click en Connect... y se abre la ventana de edición de la conexión.

El nombre de la conexión sirve para identificar dicha conexión. Host name puede ser el nombre del host o su dirección *IP*, por ejemplo "localhost". El login y el password deben ser las credenciales del servidor Nessus.

Una alternativa a la autenticación basada en credenciales es el uso del certificado SSL. Esto se configura haciendo click en el botón SSL Setup y proporcionando las rutas a los archivos cacert.pem, cakey.pem y servercert.pem como se muestra en la figura 4.3, Certificado SSL.

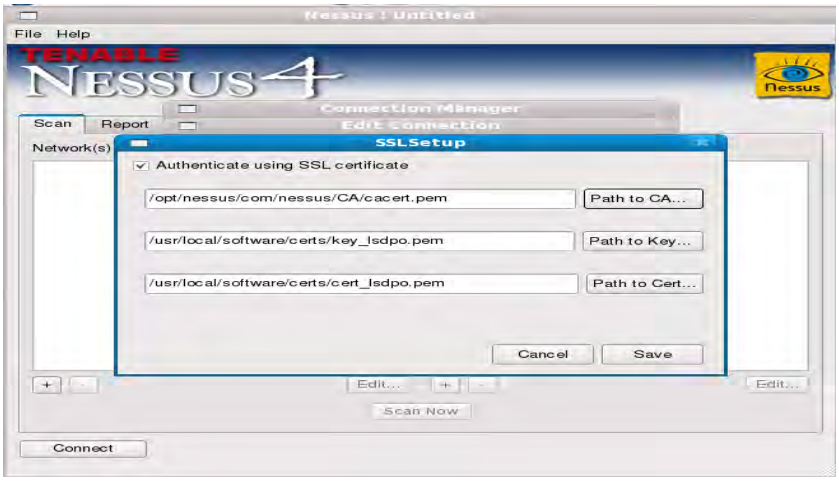


Figura 4.3 Certificado SSL.

Ahora ya se realizó la conexión entre el cliente y el servidor de Nessus y se pueden agregar políticas de escaneo a la medida haciendo click en el signo "+"(figura 4.4, Políticas de escaneo).

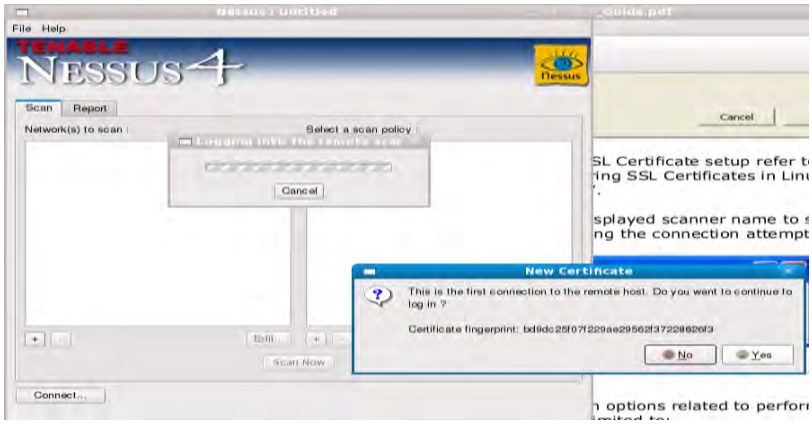


Figura 4.4, Políticas de escaneo.

4.4 HERRAMIENTAS PARA WEB

4.4.1 Nikto

Nikto se basa en la biblioteca LibWhisker, es una herramienta compatible con Capa de conexión segura (SSL), proxies y rastreos de puertos. Debido a que Nikto es un rastreador basado en *Perl*, funciona en Unix, Windows y Mac OS X.

Para llevar a cabo el proceso de instalación de la herramienta Nikto se requieren *OpenSSL* y *Apache*, por lo que después de hacer la descarga de dichos programas se ejecutan los siguientes comandos para hacer su instalación.

Instalación de *OpenSSL*:

```
tar -zxvf cd openssl-0.9.8e
cd openssl-0.9.8e
./config --prefix=/usr/local/ssl
make
make install
```

Instalación de *Apache*:

```
tar -zxvf httpd-2.0.61.tar.gz
cd httpd-2.0.61
./configure --prefix=/usr/local/apache2 --enable-so --enable-ssl --with-ssl=/usr/local/ssl
make
make install
```

Se inicia *Apache* con:
/usr/local/apache2/bin/apachectl start
Se detiene apache con
/usr/local/apache2/bin/apachectl stop

Ahora se desempaqueta y descomprime Nikto:
tar -zxvf nikto-2.1.0.tar.gz
cd nikto-2.1.0

Ya se puede ejecutar Nikto desde la línea de comandos para realizar un escaneo de los servicios web:

```
perl nikto.pl -h localhost:8080
```

4.5 RASTREADORES

Al proceso de escoger un lugar dentro de la red para ubicar ahí el sniffer generalmente es conocido por los analistas como "tapping the network" o "tapping into the wire", literalmente se traduce como "escuchar la red"[19]. Esto es esencial ya que va a determinar el alcance de la "escucha" del sniffer.

Escuchando en un entorno de hubs

Cuando se tiene una red con hubs instalados, simplemente se tiene que conectar el sniffer en un puerto disponible del hub. Esto es el sueño de todo analista de paquetes ya que un entorno así facilita mucho la tarea de hacer la escucha en la red. Como se sabe, el tráfico enviado a través de un hub es también enviado a todos los puertos conectados a dicho hub.

Aun así, es raro encontrar una red que utilice hubs. Los hubs alentan mucho el tráfico en la red, ya que solo un dispositivo a la vez puede hacer uso del hub, por ello los dispositivos tienen que competir entre si por el ancho de banda. Cuando dos dispositivos se comunican al mismo tiempo hay una colisión de paquetes, éstos se pierden y tiene que ser retransmitidos.
En esta situación, por lo único que se tiene que preocupar el analista de paquetes es por el volumen de tráfico de la captura, ya que se captura todo el tráfico que va de y hacia los dispositivos conectados al hub y esto puede generar una gran cantidad de datos, muchos de ellos irrelevantes.

Escucha en un entorno con switches

Este tipo de entorno es más común, como se sabe, el switch tiene la ventaja de transmitir datos a través de tráfico broadcast, unicast y multicast. También permiten hacer una transmisión full dúplex, esto es, se puede transmitir y recibir información de forma simultánea. Esto supone una gran ventaja para la red pero agrega complejidad a la escucha de la misma, ya que el único tráfico que se puede ver cuando se conecta un sniffer a un puerto libre del switch, es el tráfico broadcast y el tráfico enviado y recibido por la propia máquina en la que se encuentra instalado el sniffer.

Hay tres formas de hacer una escucha en un dispositivo conectado a un switch:

Port mirroring (copia de puerto), envenenamiento de ARP y hubbing out.

Port mirroring

Este es quizá el modo más sencillo de escuchar en la red un dispositivo en particular. Se requiere un switch que permita la copia de puertos. A su vez se requiere el acceso a la interfaz de línea de comandos del switch para habilitar esta característica. Si se requiere,

por ejemplo, hacer la escucha de un dispositivo conectado al puerto 3, se conecta el sniffer en el puerto 4 y se habilita al switch para que copie al puerto 4, todo el tráfico que pasa por el puerto 3.

Hubbing out

En caso de que el switch no soporte port mirroring, lo que se hace habitualmente es conectar un hub entre el dispositivo objetivo y el switch y conectar el sniffer a dicho hub, con lo que el dispositivo objetivo y el sniffer se encontrarían en mismo dominio de broadcast. Esto no se considera una práctica muy "limpia", ya que reduce el tráfico de full dúplex a half dúplex, pero se considera una buena alternativa cuando el switch no permite la copia de puertos.

Envenenamiento de ARP

El envenenamiento de ARP o ARP spoofing es el proceso mediante el cual se envían mensajes ARP a un switch o router con una dirección MAC falsa para interceptar o detener tráfico de algún dispositivo conectado. Esta técnica requiere de herramientas como Cain & Abel para ser implementada en una red y se debe ser muy cuidadoso pues puede saturar la computadora a la que se redirecciona el tráfico creando problemas de rendimiento en la red y cuellos de botella.

Escucha en un medio con routers

Cuando se requiere hacer la escucha en redes con uno o varios routers para solucionar algún problema de la red, se debe estar consciente del alcance de cada segmento de la red. Como se sabe, un dominio de broadcast se extiende hasta que alcanza un router, por ello es importante definir en que segmento se va a hacer la escucha. Para este propósito son muy importantes los mapas o diagramas de red, ya que permiten definir una mejor ubicación del sniffer, donde se pueda sacar el mejor provecho.

4.5.1 Wireshark

Para realizar la instalación correspondiente a esta herramienta es necesario realizar las siguientes actividades:

Wireshark requiere varias dependencias para la creación de la interfaz gráfica, éstas son:
glib, atk, gtk, pango y cairo:
yum install glib
yum install pango
yum install cairo

yum install atk
yum install gtk+
Ahora se instala la dependencia libpcap para la captura de datos:
cd libpcap-1.0.0
./configure
make
make install

Finalmente se instala Wireshark:
tar –jvxf wireshark-1.4.1.tar.bz2
cd wireshark-1.4.1
./configure
make
make install

Para iniciar Wireshark con su interfaz gráfica solamente se ejecuta el comando "wireshark".

4.6 HERRAMIENTAS PARA AUDITAR Y DEFENDER LA RED

4.6.1 Turtle *Firewall*

Turtle *Firewall* es un conjunto de secuencias de comandos *Perl* que hacen todo el trabajo de configurar un *firewall* Iptables automáticamente. Este programa facilita la observación de las reglas para estar seguros de obtener las declaraciones en el orden correcto.

Para la instalación de Turtle *Firewall* requerimos de una herramienta llamada Webmin. Esta herramienta permite hacer la configuración de un sistema por medio de la web y controlar y modificar aplicaciones como *Apache*, Mysql, PHP y *DHCP*, entre otras.

Se descarga el archivo rpm y lo instalamos:
rpm -Uvh webmin-1.500-1.noarch.rpm

```
advertencia:webmin-1.500-1.noarch.rpm: CabeceraV3 DSA signature: NOKEY, key ID 11f63c51
Preparando...                ########################################### [100%]
Operating system is Redhat Linux
   1:webmin                  ########################################### [100%]
Webmin install complete. You can now login to http://localhost.localdomain:10000/
as root with your root password.
```

Ahora se inicia el servidor *Apache*:
/usr/local/apache2/bin/apachectl start

Y en un navegador se ingresa el URL http://localhost.localdomain:10000/ y aparecerá la ventana para autenticarse y establecer una contraseña, el nombre de usuario debe ser un nombre de usuario administrador del sistema, al igual que la contraseña, figura 4.5, Interfaz Webmin y finalmente la interfaz de configuración de Webmin, figura 4.6.

Figura 4.5, Interfaz Webmin.

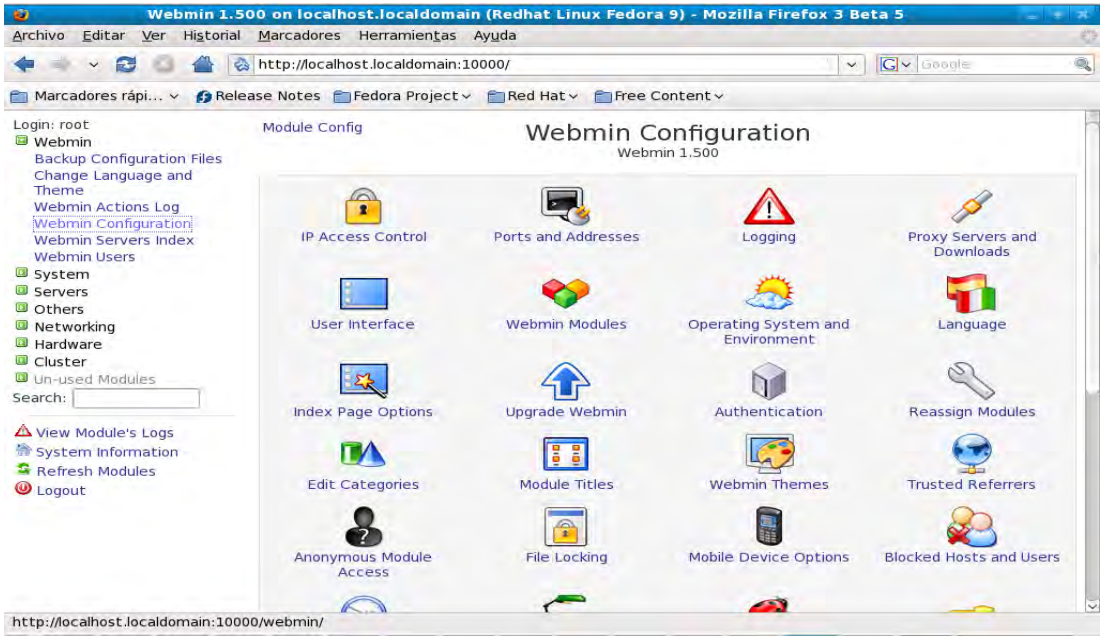


Figura 4.6, Interfaz de configuración de Webmin

Ahora se descarga el archivo de Turtle *Firewall* y se descomprime:

gzip -d turtlefirewall-1.37.wbm.gz

En el navegador nos dirigimos al link “configuration” y después al link “webmin modules”, en install module elegir “from uploaded file” y darle la ruta completa al archivo turtlefirewall-1.37.wbm, figura 4.7, Módulos Webmin.

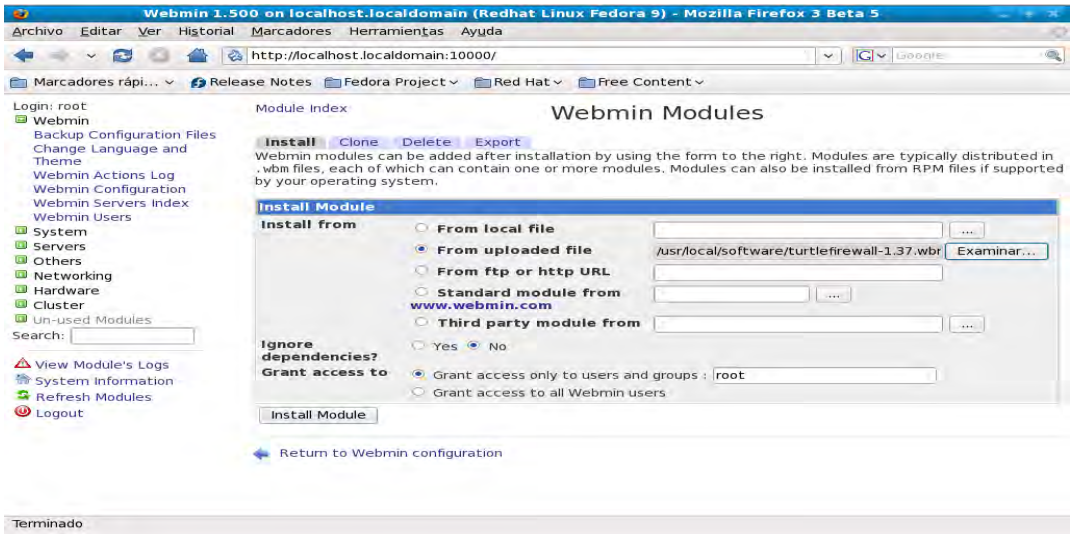


Figura 4.7, Módulos Webmin

Se hace click en instalar los scripts de inicio de Turtle *Firewall* y aparece la ventana de configuración de las reglas de Turtle *Firewall*, figura 4.8, Turtle *Firewall*.

Figura 4.8, Turtle *Firewall*

4.6.2 BASE

BASE (Basic Analysis and Security Engine), el Motor de Seguridad de Análisis Básico, es una aplicación basada en ACID (Analysis Console for Intrusion Databases) la consola de análisis para bases de datos de intrusión.

Antes de instalar BASE se requiere tener instalada una base de datos y php, A continuación se describe paso a paso el proceso de instalación.

OpenSSL

Bajar openssl-0.9.8e.tar.gz
tar -zxvf openssl-0.9.8e.tar.gz
cd openssl-0.9.8e
./config --prefix=/usr/local/ssl
make
make install
Apache
tar -zxvf httpd-2.0.63.tar.gz
cd httpd-2.0.63
./configure --prefix=/usr/local/apache2 --enable-so --enable-ssl --with-ssl=/usr/local/ssl
make
make install
/usr/local/apache2/bin/apachectl start
 /usr/local/apache2/bin/apachectl stop

Librerías

libpng es una librería del formato de imágenes PNG, es dependiente de zlib.

tar -zxvf libpng-1.2.23.tar.gz
cd libpng-1.2.23
make prefix=/usr ZLIBINC=/usr/include ZLIBLIB=/usr/lib -f scripts/makefile.linux
make -f scripts/makefile.linux test
make prefix=/usr install -f scripts/makefile.linux

Librería grafica GD
tar -zxvf gd-2.0.33.tar.gz
cd gd-2.0.33
./configure --prefix=/usr/local/gd
make
make install

Libpcap es una librería para la captura de paquetes.
tar -zxvf libpcap-1.0.0.tar.gz
cd libpcap-1.0.0
./configure
make
make install

Readline le permite a los usuarios editar líneas de comando, esto facilita usar las teclas de flechas para insertar caracteres o desplazarse a través del historial de comandos.
tar -zxvf readline-5.2.tar.gz
cd readline-5.2
./configure
make
make install

Libxml2 es una librería de "parsing" xml para el lenguaje C, desarrollada por el proyecto Gnome.
tar -zxvf libxml2-2.6.29.tar.gz
cd libxml2-2.6.29
./configure
make
make install

Zlib es una biblioteca de compresión de datos que provee una implementación del algoritmo Deflate usado en el programa de compresión gzip.
tar -zxvf zlib-1.2.3.tar.gz
cd zlib
cd 1.2.3
./configure
make
make install
Jpeg es una librería que permite la compresión de archivos de imagen basándose en el estándar del Joint Photographic Experts Group.
tar -zxvf jpeg-6b.tar.gz
cd jpeg/src
./configure
make
make install
o en su caso rpm -Uvh libjpeg-devel-6b-26.i386.rpm

Instalación de la base de datos Mysql
groupadd mysql
useradd -g mysql mysql

```
tar -zxvf mysql-5.1.46.tar.gz
cd  mysql-5.1.46
./configure --prefix=/usr/local/mysql
make
make install
cp support-files/my-medium.cnf /etc/my.cnf
cp: ¿sobreescribir «/etc/my.cnf»? (s/n) s
cd /usr/local/mysql
chown -R mysql .
chgrp -R mysql .
.bin/mysql_install_db --user=mysql
chown -R root .
chown -R mysql var
.bin/mysqld_safe --user=mysql &

su – mysql
cd /usr/local/mysql/bin
./mysql -u root
mysql>show databases;
mysql>exit
exit
y como usuario root

para detener mysql:

/usr/local/mysql/bin/mysqladmin shutdown

 ./mysqladmin -u root password "my5q1&"

Después como usuario mysql podemos iniciar la base con
/usr/local/mysql/bin/mysqld_safe --user=mysql &
cd /usr/local/mysql/bin
y pararlo:
./mysqladmin -u root shutdown –p

Se establece un password para mysql
mysql -u root mysql
mysql>update user set password=PASSWORD('my5q1&') where user='root'
;
Query OK, 3 rows affected (0,00 sec)
Rows matched: 3  Changed: 3  Warnings: 0
mysql>flush privileges;

Ahora nos ingresamos con:
```

./mysql -u root -p
y el password

Instalación de PHP
tar -zxvf php php-5.3.2.tar.gz
cd php-5.3.2
./configure --prefix=/usr/local/php --with-apxs2=/usr/local/apache2/bin/apxs --with-config-file-path=/usr/local/php --enable-sockets --with-mysql=/usr/local/mysql --with-zlib-dir=/usr/local --with-gd=/usr/local/gd --with-png-dir=/usr/include/libpng12 --with-jpeg-dir=/usr/local/jpeg
make
make install
ahora se configura php
cp php.ini-dist/usr/local/php/php.ini
en el caso de php 5.3.2
cp php.ini-production /usr/local/php/php.ini

Se modifica el archivo httpd.conf

vi /usr/local/apache2/conf/httpd.conf
DirectoryIndex index.html
queda:
DirectoryIndex index.html index.html.var index.php

Y finalmente se agregan las siguientes lineas:

AddType application/x-httpd-php .php .php4 .php5 .phtml .html
AddType application/x-httpd.php-source .phps
AddType image/x-icon .ico

Instalación de Snort
Instalación de libpcap 1.0.0
tar –zxvf libpcap-1.0.0.tar.gz
cd libpcap-1.0.0
./configure
make
make install

Instalación de pcre
tar -zxvf pcre-8.02.tar.gz
cd pcre-8.02
./configure

make
make install

Instalación de libnet
tar -zxvf libnet-1.0.2a.tar.gz
cd Libnet-1.0.2a/
En caso de ser necesario, también se instalan flex y bison.

./configure&&make&&make install

tar -zxvf snort-2.8.5.2.tar.gz
cd snort-2.8.5.2
./configure --enable-targetbased &&make&&make install
mkdir /etc/snort
mkdir /var/log/snort
cd /etc/snort
se copia el archivo snortrules-snapshot a /etc/snort
tar -zxvf snortrules-snapshot-2.8.tar.gz
cp etc/* /etc/snort
ln -s /usr/local/bin/snort /usr/sbin/snort
groupadd snort
useradd -g snort snort
chown snort:snort /var/log/snort
touch /var/log/snort/alert
chown snort:snort /var/log/snort/alert
chmod 600 /var/log/snort/alert
cp /etc/snort/so_rules/precompiled/FC-9/i386/2.8.4/*.so /usr/local/lib/snort_dynamicrules
mv /usr/local/lib/snort_dynamicrules /usr/local/lib/snort_dynamicrule

cp snort.conf /etc/snort/
cp: ¿sobreescribir «/etc/snort/snort.conf»? (s/n) s

Editamos el archivo /etc/snort

vi /etc/snort/snort.conf
Cambiamos la variable RULE_PATH por /etc/snort/rules
Comentamos las líneas con la frase output modules on.
Localizamos la frase "output log_unified". Debajo de esta línea insertamos lo siguiente:
output unified2: filename snort.log, limit=128

Probamos snort
[root@localhost snort]# snort -c /etc/snort/ -T

```
        --== Initialization Complete ==--
   ,,_     -*> Snort! <*-
  o"  )~   Version 2.8.5.2 (Build 121)
```

```
  ""    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
        Copyright (C) 1998-2009 Sourcefire, Inc., et al.
        Using PCRE version: 8.01 2010-01-19
Snort successfully loaded all rules and checked all rule chains!
Snort exiting
```

Se levanta la base de datos
su – mysql
cd /usr/local/mysql/bin/

levantamos la base de datos
./mysqld_safe --user=mysql &
./mysql --user=root -p

Si aun no tiene password mysql se lo creamos
mysql> UPDATE user SET Password=PASSWORD('my5q1&') WHERE user='root';

password de mysql my5q1&

Ahora como usuario root creamos la base de datos snort

mysql>create database snort;
y le asignamos un password a snort
mysql> grant CREATE, INSERT, SELECT, DELETE, UPDATE on snort.*to
snort@localhost;
SET PASSWORD FOR snort@localhost=PASSWORD('5n0rt&');
exit

Se crea las tablas de snort
cd /usr/local/mysql/bin/
./mysql --user=root -p < /usr/local/software/snort-2.8.5.2/schemas/create_mysql snort
y se verifica que se hayan creado las tablas correctamente

mysql -p
SHOW DATABASES; Debe haber 4 bases de datos
use snort
SHOW TABLES; Debe haber 16 tablas
exit
exit
ahora como usuario root

Instalación de BASE y ADODB

cd /var/www/html
 cp /root/Escritorio/snort/adodb511.tgz .
 cp /root/Escritorio/snort/base-1.4.5.tar.gz .
 tar zxvf adodb511.tgz

tar -zxvf base-1.4.5.tar.gz
chown apache base-1.4.5
chgrp apache base-1.4.5
chmod 777 /var/www/html/base-1.4.5
vi /etc/php.ini

Verificamos que la variable error reporting tenga el siguiente parámetro:

error_reporting = E_ALL & ~E_NOTICE
Instalamos mail y mail_mime, dos funciones de php que se requieren para la interfaz gráfica.
cd /usr/local/php/bin
y ejecutamos
./pear install Mail
./pear install Mail_Mime

Ahora se reinicia el servicio.
service httpd restart
El servidor debe estar configurado para que lea /var/www/html
en el archivo /usr/local/apache/conf/httpd.conf
#DocumentRoot "/usr/local/apache2/htdocs"

DocumetRoot "/var/www/html"
Ahora se abre un navegador y se ingresa la siguiente dirección:
http://localhost/base-1.4.5/setup/index.php
La salida se muestra en la figura 4.9, Instalacion de BASE 1.

4.9 Instalación de BASE 1

click en continue

Ingresamos la ruta a adodb /var/www/html/adodb5, figura 4.10, Ruta a Adodb.

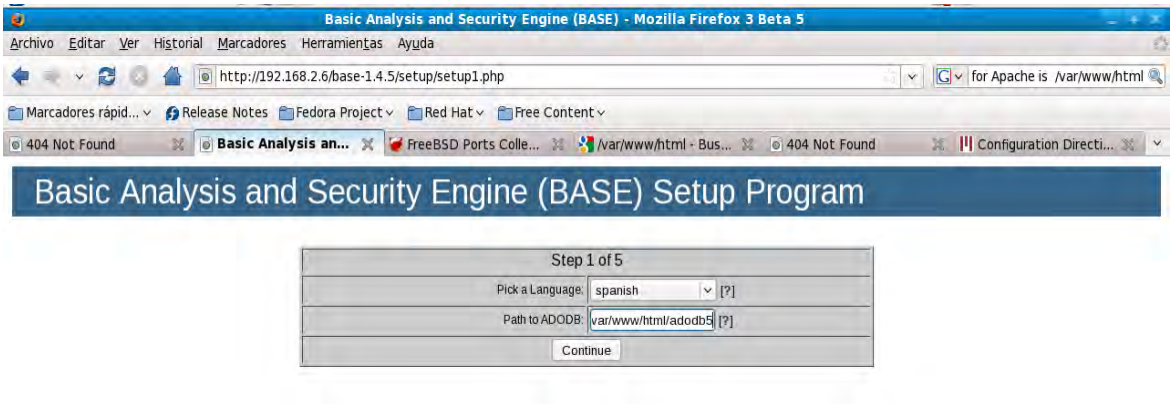


Figura 4.10, Ruta a Adodb

Ahora se ingresan los parámetros del usuario snort en la base de datos. Figura 4.11, Parámetros de Snort.
Database Name=snort, Database Host=localhost, Database User=snort, Database Password=5n0rt&

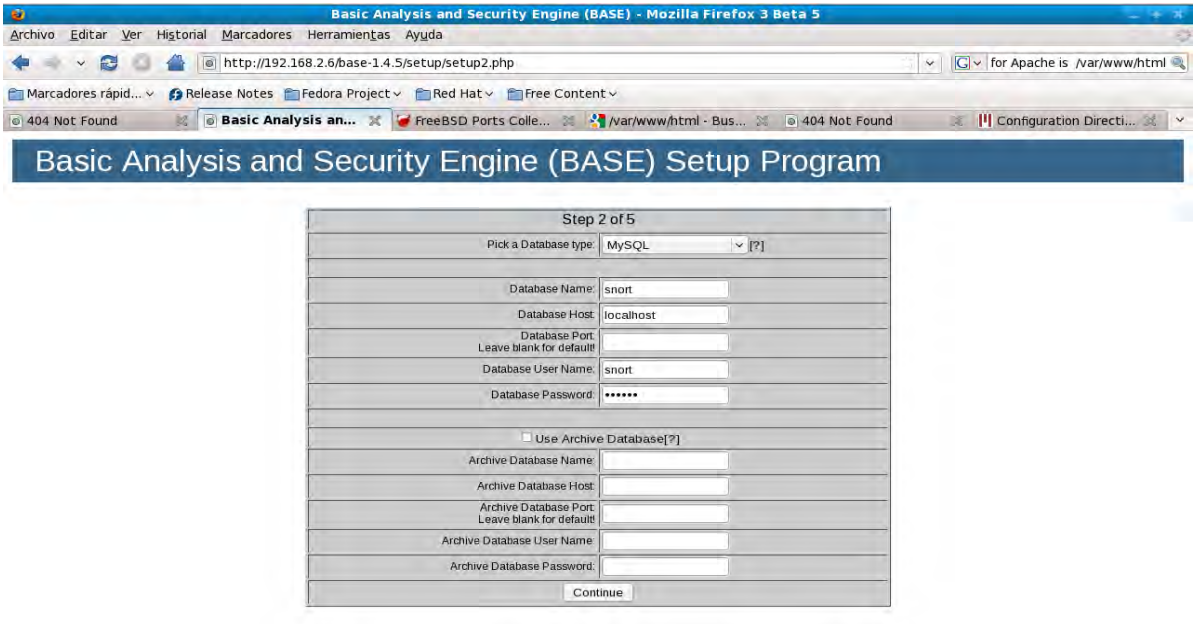


Figura 4.11, Parámetros de Snort

En la siguiente pantalla se ingresa el nombre de usuario del administrador de Snort así como su password.
Admin User Name=snort, Password=5n0rt&, Full Name=snort

El siguiente paso es hacer click en Create BASE AG para agregar las tablas en la base de datos de Snort que dan soporte a BASE. Figura 4.12, Create BASE AG.

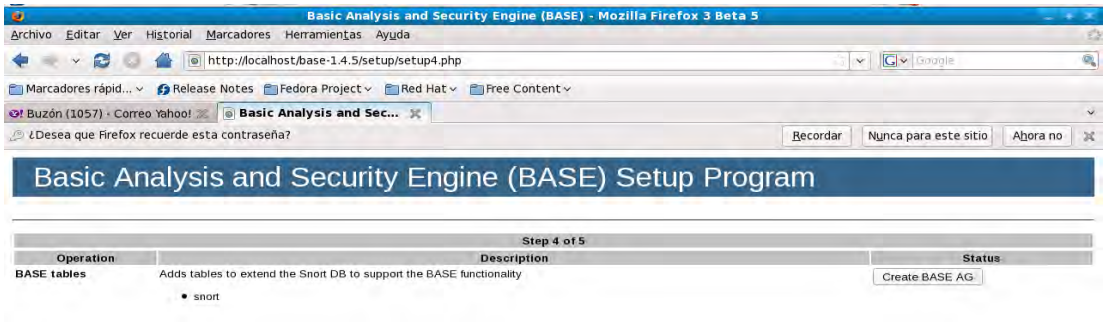


Figura 4.12, Create BASE AG.

Después se hace click en "ir al paso 5" y aparece lo siguiente, Figura 4.13, Instalación final BASE:

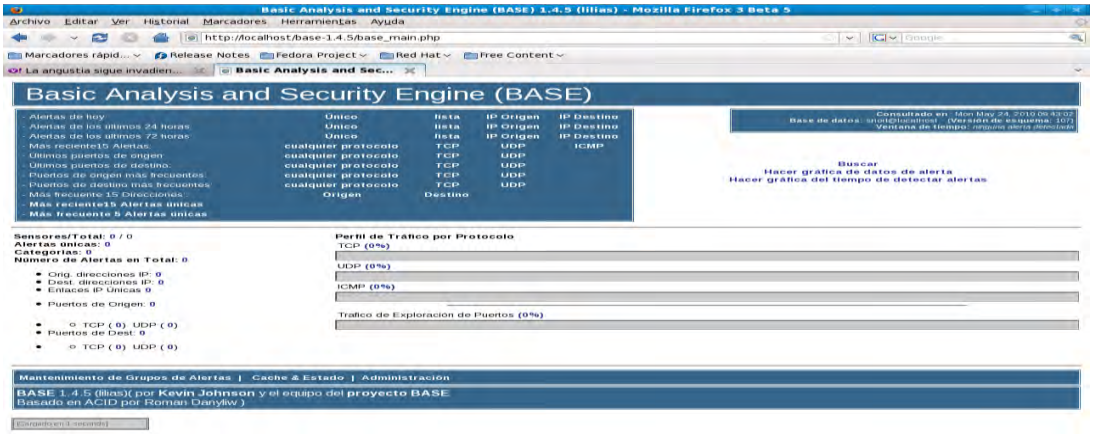


Figura 4.13 Instalación final BASE

Instalación de Barnyard

Barnyard es un intérprete de información de las bitácoras de Snort a MYSQL, su instalación se realiza de la siguiente manera:

tar -zxvf barnyard2-1.8.tar.gz
Primero se debe detener mysql porque cuando está levantado no se pueden leer sus archivos y es necesario leerlos para la instalacion de barnyard.
cd barnyard2-1.8
./configure --with-mysql=/usr/local/mysql
make
make install
cp etc/barnyard2.conf /etc/snort
vi /etc/snort/barnyard2.conf
Ahora se localiza la frase config hostname y se modifica
Se cambia "thor" por "localhost"

Se busca "config interface" en el mismo archivo, en esta línea debe aparecer eth0
En la línea en que aparece la frase output database se debe editar y asignar los siguientes parámetros:
alert, mysql, user=snort password=5n0rt& dbname=snort host=localhost
Se levantan nuevamente mysql y snort
Se ingresa el siguiente comando en una terminal:

snort -c /etc/snort/snort.conf -i eth0

Se abre una nueva terminal y se inserta el siguiente comando
ls -la /var/log/snort
el resultado es:

```
total 12
drwxr-xr-x  2 snort snort 4096 may 12 18:59 .
drwxr-xr-x 22 root  root  4096 may 12 18:27 ..
-rw-------  1 snort snort    0 abr 20 19:20 alert
-rw-r--r--  1 root  root    39 abr 20 20:11 barnyard.waldo
-rw-------  1 root  root     0 may 12 18:59 snort.log.1273708766
```

Ahora se tiene que copiar el último numero de snort.log
En este caso 1273708766
cd /var/log/snort
Si no existe el archivo barnyard.waldo, lo creamos
vi barnyard.waldo
Se insertan las siguientes líneas y se guardan los cambios:

/var/log/snort
snort.log
<el numero de 10 digitos arriba mencionado (1273708766
)>
0

Si tratamos de iniciar barnyard nos mandara el siguiente error
ERROR: Stat check on log dir (/var/log/barnyard2) failed: No such file or directory.
Fatal Error, Quitting..
Para evitar dicho error creamos el directorio donde se guardara la bitacora de barnyard
mkdir /var/log/barnyard2
iniciamos barnyard:

/usr/local/bin/barnyard2 -c /etc/snort/barnyard2.conf -G /etc/snort/gen-msg.map -S /etc/snort/sid-msg.map -d /var/log/snort/ -f snort.log -w /var/log/snort/barnyard.waldo

Ahora probamos Snort
Se tiene que crear una regla para probar Snort en el archivo local.rules
Las reglas locales son reglas que el administrador de snort escribe él mismo y se tiene la convención de iniciar con SID (Snort ID) de 1,000,000-1,999,999.

Se tiene que abrir una tercera terminal y editar el archivo local.rules
vi /etc/snort/rules/local.rules

Se agrega la siguiente línea:
alert tcp any any <> any 80 (msg: "Test web activity"; sid:1000001;)

Se guardan los cambios y se debe reiniciar Snort.
Ahora abrimos un navegador de Internet, visitamos cualquier sitio y en la interfaz oprimimos Ctrl + c para detener snort y ver lo que ha monitoreado.

Ahora se tiene que abrir BASE
El directorio a abrir por default debe ser /var/www/html, para ello se debe modificar la siguiente línea en el archivo httpd.conf:

DocumentRoot "/var/www/html"
Después se ingresa la siguiente dirección en un navegador web: http://localhost/base-1.4.5 y vemos los eventos detectados, si vemos eventos con un número SID 1000001 snort funciona correctamente, figura 4.14, Alertas en BASE.
Ahora se tiene que deshabilitar la regla que acabamos de crear.

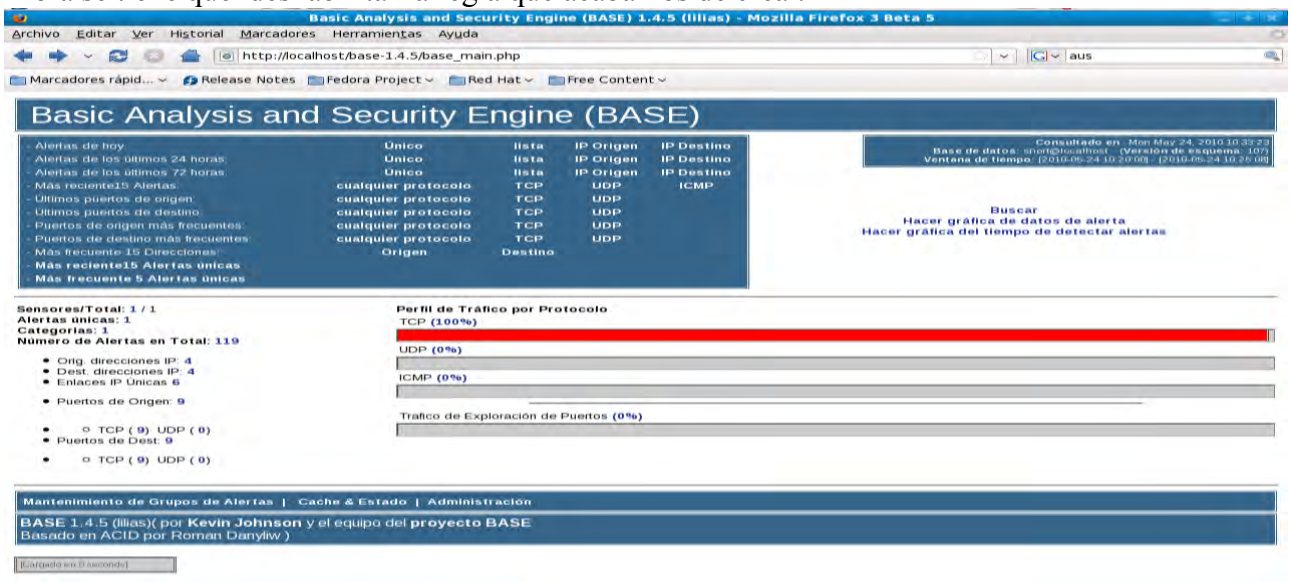


Figura 4.14, Alertas en BASE.

Hasta este momento se ha logrado hacer la instalación y configuración de las herramientas del sistema de defensa que se emplearán para monitorear todos los eventos en la red, en el siguiente capítulo, denominado, pruebas del sistema de defensa, se realizan una serie de pruebas sobre diferentes elementos de dicho sistema para verificar que la instalación y configuración de las herramientas sean las óptimas para obtener el máximo rendimientos de las mismas.

CAPÍTULO 5
PRUEBAS DEL SISTEMA

DE DEFENSA

PRUEBAS DEL SISTEMA DE DEFENSA

5.1 PLAN DE PRUEBAS

En este capítulo se hace una serie de pruebas dirigidas a ciertas áreas del sistema de defensa con los siguientes objetivos en mente:

- Detectar un error.
- Tener un buen caso de prueba, es decir que se tenga más probabilidad de mostrar un error no descubierto antes.
- Descubrir un error no descubierto antes.

La pruebas no pueden asegurar que no existen errores en la configuración total del sistema sólo pueden mostrar que existen defectos en las áreas que se están analizando, dichos defectos pueden ser tanto de instalación como de configuración y uso del software analizado.

Los elementos del sistema de defensa que se someterán a prueba son: John the Ripper, el analizador de vulnerabilidades web Nikto, Wireshark, Nessus, Turtle *Firewall* y el Sistema de Detección de Intrusos.

Todos estos elementos se probarán por separado, ya que el funcionamiento de dos o más elementos al mismo tiempo podría alterar los resultados de las pruebas, por ejemplo, un análisis de vulnerabilidades con Nikto se registraría en el detector de intrusos y se podría perder de vista un ataque real al sistema.

Los elementos que no se probarán son: *OpenSSH* y Cracklib por ser elementos que ya se integran por defecto en la mayoría de los sistemas Linux y únicamente requieren una correcta configuración. La instalación, configuración y prueba de Bastille Linux se realiza en el capítulo 6. A continuación se presentan varios casos de uso de los programas instalados y configurados en el sistema de defensa, en cada uno de ellos se determina la estrategia a seguir para la prueba, los participantes en la prueba, y se analizan los resultados obtenidos a partir de las condiciones dadas.

5.2 JOHN THE RIPPER

Nombre:	**Prueba de John the Rippper.**
Descripción: se utiliza John de Ripper para encontrar contraseñas débiles que pueden ser una vulnerabilidad en el sistema.	
Actores: usuario con privilegios.	
Precondiciones: El usuario con privilegios puede ejecutar la herramienta John the Ripper y leer los archivos /etc/passwd y /etc/shadow. Se crea una contraseña débil para root y se espera que la herramienta la descubra.	
Flujo normal: El usuario con privilegios ejecutará John the Ripper con un archivo de lista de palabras, la herramienta obtendrá las contraseñas inseguras o débiles.	
Flujo alternativo: la herramienta no detecta contraseñas inseguras y continúa con su ejecución hasta que el usuario con privilegios decide detener la prueba.	
Poscondiciones: se detecta la contraseña débil y se muestra el resultado.	

Los sistemas Linux recientes utilizan passwords encriptados con *SHA-256* o *SHA-512* (passwords que empiezan con 5 o 6 respectivamente, en el archivo shadow), estos algoritmos tienen ventajas ante los algoritmos utilizados en sistemas Linux viejos, como *MD5* (1), pero John the Ripper no tiene soporte para los nuevos algoritmos implementados.

Al ver la primera línea del archivo /etc/shadow podemos cerciorarnos de que este sistema en particular utiliza *SHA-512*,

root**6**VdinK4PM$ZZlLNW76hm0oWfkmU9W4I3xDpbw9Ou31s7D2FidFQpilvQUEF4
4HfuOzI47CykLWy5I2fuyG2k6.RXVCdk5Jg0:14756:0:99999:7:::

por lo que John the Ripper no puede crackear este tipo de contraseñas.

Sin embargo, se puede hacer uso de un script en *perl* que utiliza la función crypt y la lista de palabras de John the Ripper. Aunque de rendimiento muy lento, este script es una opción viable mientras no exista soporte para John the Ripper con el algoritmo SHA-512.

Primero se cambia la contraseña; en este caso y únicamente como prueba se usará el usuario root.

[kakaroto@localhost run]# passwd root

Cambiando la contraseña del usuario root.

Nueva UNIX contraseña:
CONTRASEÑA INCORRECTA: Es demasiado corta.
Vuelva a escribir la nueva UNIX contraseña:
passwd: todos los tokens de autenticación se actualizaron exitosamente.
You have new mail in /var/spool/mail/root

Ahora se utiliza el comando unshadow para combinar los archivos /etc/passwd y /etc/shadow, para que puedan ser usados por John the Ripper.

[kakaroto@localhost run]# ./unshadow /etc/passwd /etc/shadow > crack.db

Y finalmente se usa el script en *perl* y una lista de palabras para desencriptar las contraseñas.

[kakaroto@localhost run]# cat password.lst | perl cryptcrack.pl -f crack.db

Read 49 hashes from file
Spawning 4 threads
11.071 keys per second.
12.201 keys per second.
11.001 keys per second.
11.801 keys per second.
12.001 keys per second.
12.001 keys per second.
12.201 keys per second.
.
.11.201 keys per second.
12.201 keys per second.
11.801 keys per second.
18.001 keys per second.

RESULTADOS:

FOUND: hola
(6VdinK4PM$ZZlLNW76hm0oWfkmU9W4I3xDpbw9Ou31s7D2FidFQpilvQUEF44HfuOzI47
CykLWy5I2fuyG2k6.RXVCdk5Jg0)

Cracked passwords:

John the Ripper muestra la contraseña encontrada, y la línea del archivo crack.db que fue descifrada, con lo que se tiene un resultado satisfactorio ya que se pudo verificar que el programa utilizado es adecuado y funciona correctamente.

5.3 NESSUS

Nombre:	**Prueba de Nessus.**
Descripción: Se ejecutará Nessus para encontrar las vulnerabilidades del sistema operativo, de la red y de las aplicaciones en el sistema operativo con una instalación base, sin actualizar.	
Actores: usuario con privilegios.	
Precondiciones: el sistema operativo no se ha actualizado por lo que Nessus deberá detectar todos los paquetes y programas que requieren actualización.	
Flujo normal: El usuario con privilegios ejecutará Nessus y este programa detectará las vulnerabilidades por falta de actualización de software.	
Flujo alternativo: la herramienta no detecta ninguna de las vulnerabilidades o sólo detecta algunas de las posibles vulnerabilidades.	
Poscondiciones: se detectan las vulnerabilidades del sistema y se crea el informe correspondiente.	

El primer paso consiste en ejecutar el cliente de Nessus, para ellos se abre una terminal y se ingresa el siguiente comando:

[root@kakaroto]#/opt/nessus/bin/NessusClient

Ahora en la interfaz de Nessus se ejecuta un escaneo al servidor local, con todos los plugins por default activados, figura 5.1 Plugins de Nessus.

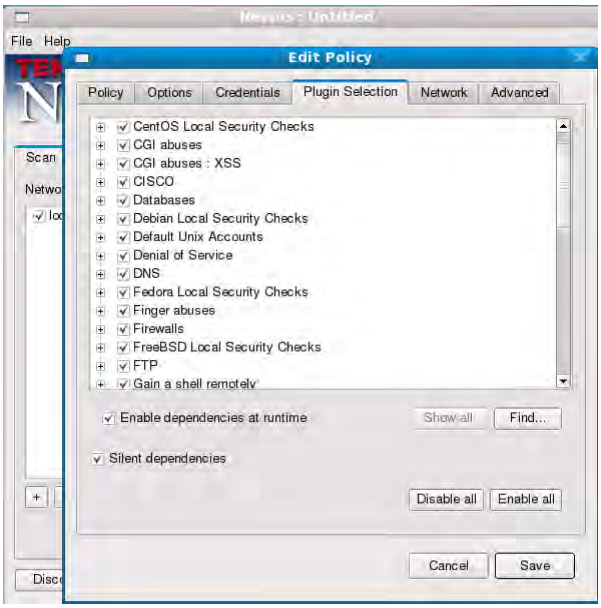


Figura 5.1, Plugins de Nessus.

Nessus permite generar un archivo html con los resultados del escaneo, parte de dicho archivo se muestra a continuación, en él se detallan las vulnerabilidades del sistema y la manera de corregirlas, figura 5.2, Resultados de Nessus.

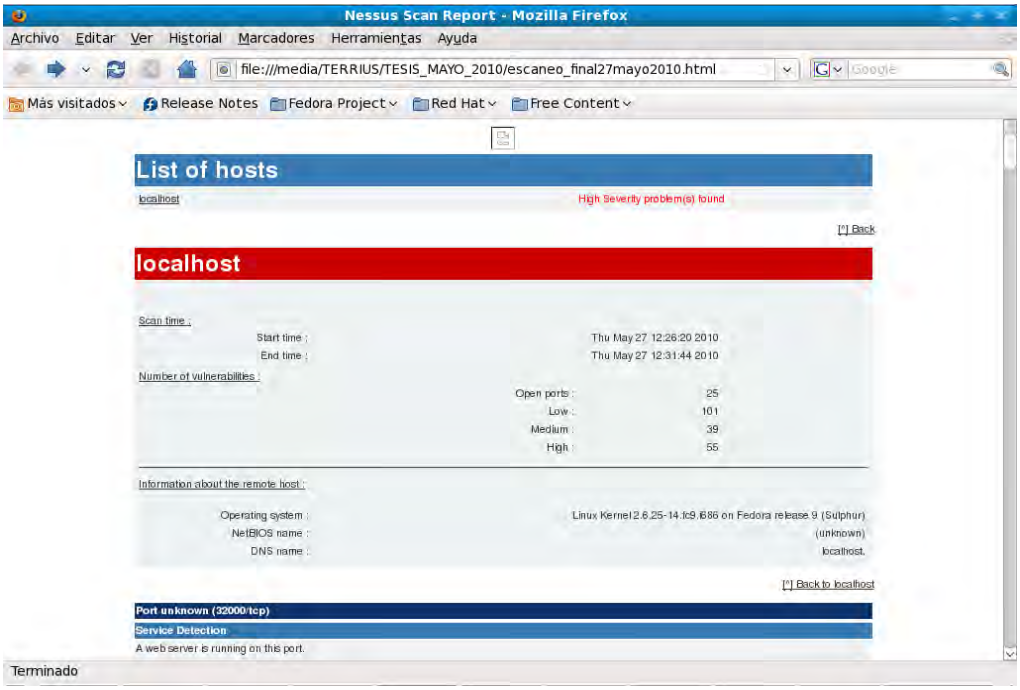


Figura 5.2, Resultados de Nessus.

RESULTADOS:

En este caso Nessus detectó 101 vulnerabilidades de riesgo bajo, 39 de riesgo medio y 55 de alto riesgo, éstas últimas relacionadas en su totalidad con la falta de actualización del sistema operativo y programas de aplicación. Las vulnerabilidades de riesgo medio y bajo fueron en su mayoría vulnerabilidades del servidor web y de actualización de librerías.

La prueba de Nessus se considera exitosa al detectar más vulnerabilidades de las que se esperaban encontrar en un principio. Esto muestra que Nessus está funcionando correctamente de acuerdo con los plugins activados.

5.4 NIKTO

Nombre:	**Prueba de Nikto.**
Descripción: Se ejecutará el programa Nikto en el servidor web con el puerto a verificar, en este caso el puerto 80.	
Actores: usuario con privilegios.	
Precondiciones: El servidor web ha sido instalado pero sin hacer una configuración segura, se ejecutará el programa Nikto y se espera que detecte las vulnerabilidades a que está expuesto.	
Flujo normal: El usuario con privilegios ejecutará Nikto y la herramienta detectará las vulnerabilidades existentes en el servidor web.	
Flujo alternativo: la herramienta no detecta ninguna vulnerabilidad.	
Poscondiciones: se detectan vulnerabilidades web y se muestran en pantalla.	

Se ejecuta Nikto en el servidor local y el puerto 80.

[root@localhost nikto-2.1.0]# perl nikto.pl -h localhost:80

```
- ***** SSL support not available (see docs for SSL install instructions) *****
- Nikto v2.1.0/2.1.0
---------------------------------------------------------------------------
+ Target IP:          127.0.0.1
+ Target Hostname:    localhost
+ Target Port:        80
+ Start Time:         2010-05-28 17:56:50
---------------------------------------------------------------------------
+ Server: Apache/2.2.9 (Fedora)
+ OSVDB-0: Apache/2.2.9 appears to be outdated (current is at least Apache/2.2.14). Apache 1.3.41 and
2.0.63 are also current.
+ OSVDB-0: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-682: /usage/: Webalizer may be installed. Versions lower than 2.01-09 vulnerable to Cross Site
Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-3092: /manual/: Web server manual found.
+ OSVDB-3268: /icons/: Directory indexing is enabled: /icons
+ OSVDB-3268: /manual/images/: Directory indexing is enabled: /manual/images
+ OSVDB-3233: /icons/README: Apache default file found.
+ 3582 items checked: 8 item(s) reported on remote host
+ End Time:           2010-05-28 17:57:14 (24 seconds)

---------------------------------------------------------------------------
```

- 1 host(s) tested

RESULTADOS:

Se encontraron 8 vulnerabilidades en el servidor web, entre ellas, destacan por su severidad, la falta de actualización del servidor web y la vulnerabilidad conocida como Cross Site Tracing (XST), vulnerabilidad que explota controles ActiveX, Flash, Java y otros que permiten la ejecución de una llamada http trace para obtener el valor de las cookies de un navegador.

Esta prueba se considera exitosa ya que Nikto fue capaz de detectar las vulnerabilidades a que está expuesto el servidor web, lo que permite en su caso generar un informe con las alertas para que el área web corrija la configuración del servidor analizado.

5.5 WIRESHARK

Nombre:	**Prueba de Wireshark.**
Descripción: Se ejecutará Wireshark y después se creará cierta actividad en la red que pueda ser detectada, en este caso, se establecerá una comunicación a través de mensajería instantánea para hacer el seguimiento TCP stream y se abrirá un sitio web.	
Actores: usuario con privilegios.	
Precondiciones: Se debe crear cierta actividad en la red que sea monitoreada por un usuario privilegiado que ejecute Wireshark.	
Flujo normal: El usuario con privilegios ejecutará Wireshark y la herramienta detectará la actividad en la red.	
Flujo alternativo: la herramienta no detecta la actividad creada.	
Poscondiciones: se capturan los paquetes de la actividad creada para posterior análisis.	

El usuario con privilegios debe loguearse y ejecutar Wireshark, para ello se deben teclear los siguientes comandos:

[root@kakaroto]#su – estudiante
[estudiante@kakaroto]#wireshark

Se abre la interfaz de Wireshark, en el menú Capture se hace click en la pestaña Start, para iniciar la captura de paquetes. Ahora se abre un navegador de Internet y para este ejemplo se ingresa la dirección http://www.openwall.com, en la interfaz de Wireshark se muestra la captura de la actividad creada al abrir una página web, figura 5.3, Captura de Wireshark.

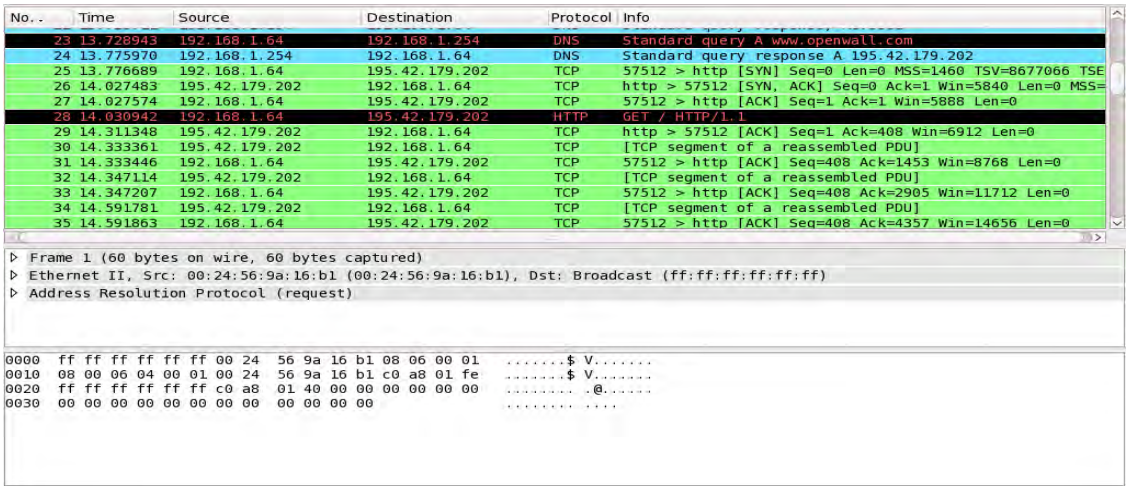


Figura 5.3 Captura de Wireshark

Wireshark muestra los paquetes capturados, la dirección *IP* de origen, la dirección *IP* de destino, el protocolo utilizado e información adicional sobre el paquete, como el estado de la conexión.

Ahora, se crea una conversación con un cliente de mensajería instantánea, al hacer el seguimiento de *TCP* stream se puede observar toda la comunicación que ha habido entre los usuarios del servicio de mensajería, como se muestra en la figura 5.4, Follow *TCP* stream.

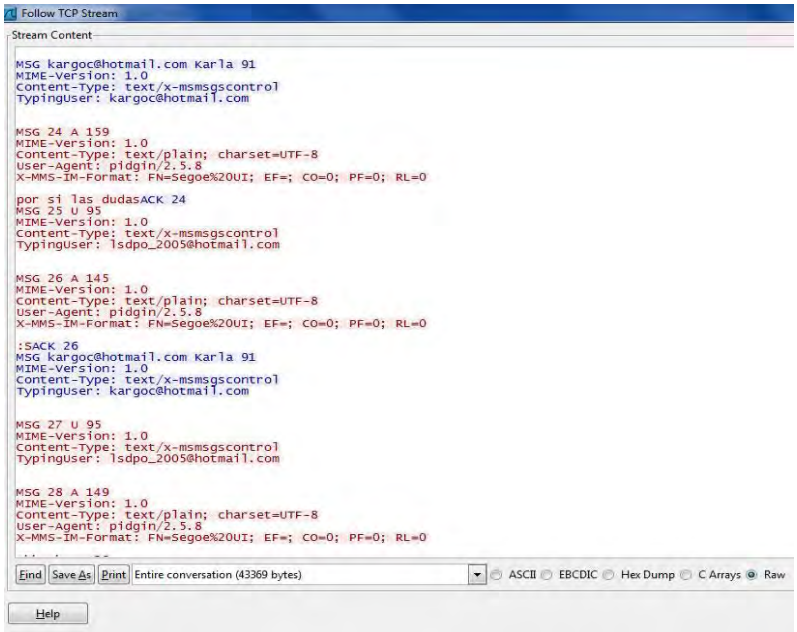


Figura 5.4 Follow TCP stream

Se establece una conexión a la página de Snort, para ello se realiza un proceso denominado Three hand shake (figura 5.5 Three hand shake), en el cuál, la computadora cliente, dirección IP 192.168.2.10 envía un paquete SYN para comunicar que se desea establecer la comunicación, el servidor, en este caso el equipo con dirección IP 68.177.102.20 envía una respuesta SYN/ACK, indicando que recibió un paquete SYN. Finalmente el cliente envía un paquete ACK indicando que recibió el paquete SYN/ACK y de esta manera se establece la comunicación, después de esto se inicia la transmisión de la página web que se está solicitando (paquete número 32).

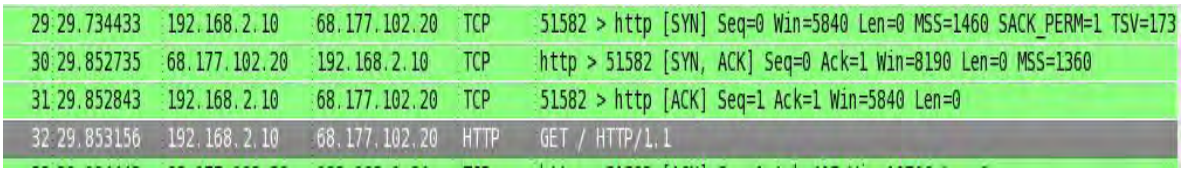


Figura 5.5 Three hand shake

RESULTADOS:

Wireshark ha detectado la actividad en la red correctamente, tanto la captura de paquetes como el seguimiento de *TCP* stream por lo que podemos afirmar que la herramienta tiene un funcionamiento adecuado y nos permite hacer un seguimiento adecuado de todas las actividades en la red y los protocolos utilizados. Además el analizador de protocolos también detectó la actividad de otros sitios que se abrieron posteriormente, por lo que se puede concluir como una prueba exitosa.

5.6 TURTLE *FIREWALL*

Nombre:	**Prueba de Turtle *Firewall*.**
Descripción: Se ejecutará cierta actividad en la red interna para cerciorarnos de que el *Firewall* permite el paso del tráfico estrictamente permitido de acuerdo con la configuración realizada.	
Actores: usuario con privilegios, usuario sin privilegios.	
Precondiciones: El usuario con privilegios realizará la configuración del *Firewall* en un equipo con dos interfaces, en este caso eth0 será la conexión a nuestra red interna y eth1 la conexión al exterior. Además se tiene un servidor web que será protegido por el *firewall* y el mismo *firewall* que se considera como un equipo independiente. Se crearán reglas que permitan el tráfico http y ssh de la red interna hacia el exterior (Internet) y se bloqueará todo el tráfico restante.	
Flujo normal: El usuario sin privilegios tratará de establecer conexiones con el exterior mediante protocolos o servicios permitidos y el *firewall* debe permitir dichas conexiones pero después de haber sido analizadas por las reglas establecidas.	
Flujo alternativo: el *firewall* permite el tráfico que debería estar bloqueado.	
Poscondiciones: se permite el paso del tráfico permitido y se realiza el registro de dicho tráfico.	

Lo primero que se debe hacer es ejecutar un navegador web y mediante la herramienta Webmin hacer la configuración del *Firewall*, para ello se debe ingresar la siguiente dirección en el navegador: http://localhost:10000.

Ahora en el menú Networking, se hace click en Turtle *Firewall*, finalmente click en el vínculo “Firewall Items”.

En este apartado se definen los elementos que estará protegiendo el *firewall*, el propio *firewall* se considera como un elemento independiente en la configuración.

En nuestro caso se define una zona llamada exterior asignada a la interfaz eth1 de la tarjeta de red, que comprende el tráfico que llega desde el exterior hacia el *firewall*, una zona denominada lan, asignada a la interfaz eth0 que representa el tráfico que sale de la red

interna hacia el *firewall* y una zona llamada servidor web, que corresponde a una máquina que brinda dicho servicio y que estará protegida por el *firewall*(*DMZ*), figura 5.6, Elementos del *firewall*.

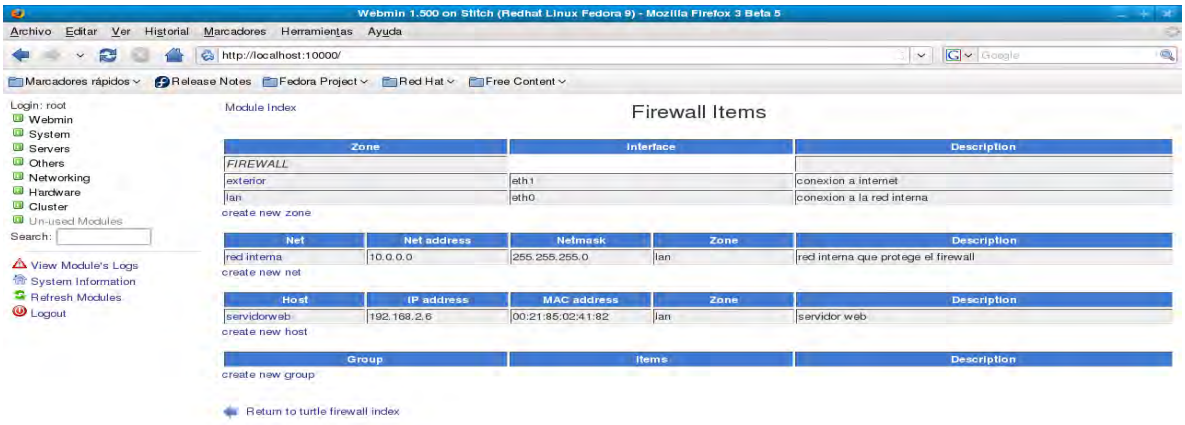


Figura 5.6, Elementos del *firewall*.

Una vez definidos los elementos que serán protegidos por el *firewall*, se tienen que crear las reglas que definen el tipo de tráfico que puede atravesar por el *firewall* y las acciones que el mismo llevará a cabo con dicho tráfico, figura 5.7, Creación de reglas.

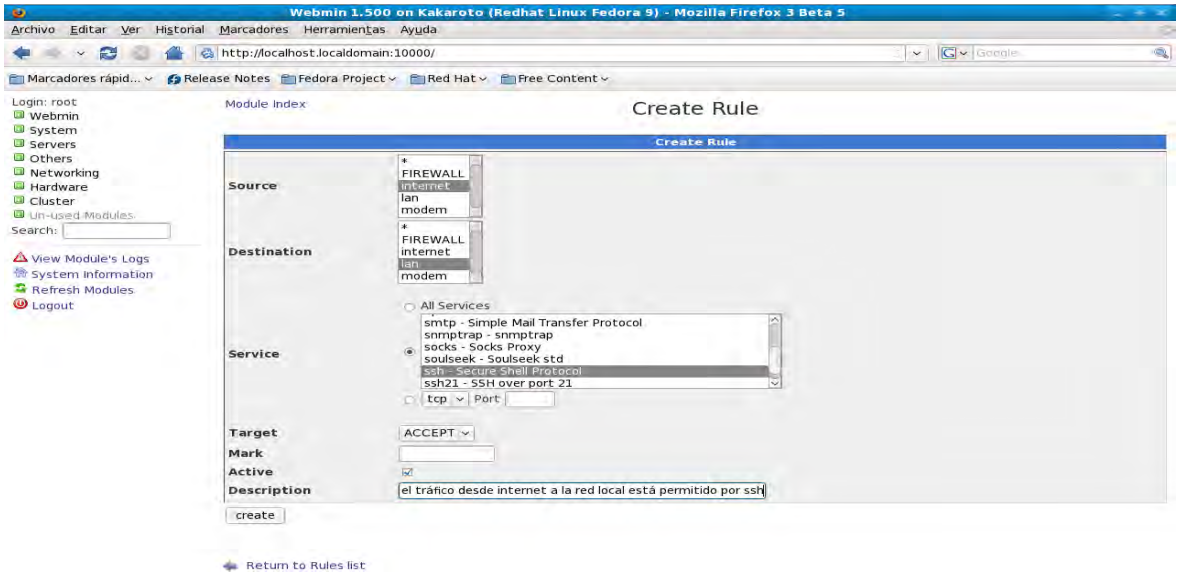


Figura 5.7 Creación de reglas.

Para cuestiones de prueba se crean 7 reglas para el funcionamiento de este *firewall*, se aceptan los protocolos http y https tanto para el tráfico entrante como para el saliente, se

rechaza cualquier actividad FTP que salga de la red y se aceptan las conexiones hechas por medio de *Secure Shell*, figura 5.8, Reglas del *firewall*.

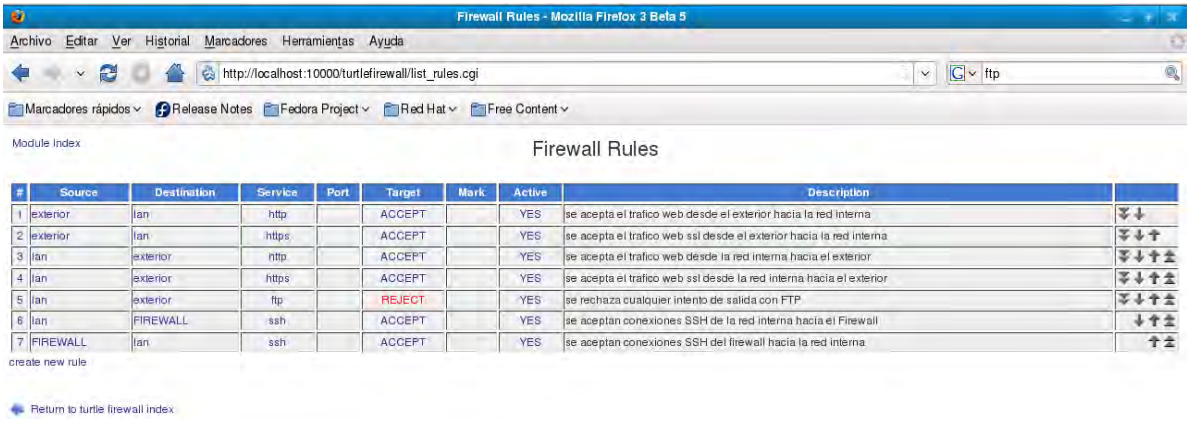


Figura 5.8, Reglas del *firewall*.

Para probar el funcionamiento del *firewall* se abre el navegador web y se hace el intento de salir a Internet desde la red interna.

Desde la red interna se tiene salida a Internet por lo que únicamente falta verificar los registros del *firewall* para determinar que dicho tráfico haya sido analizado por el *firewall* antes de permitir su paso, Figura 5.9, Registro del *firewall*, puerto 80.

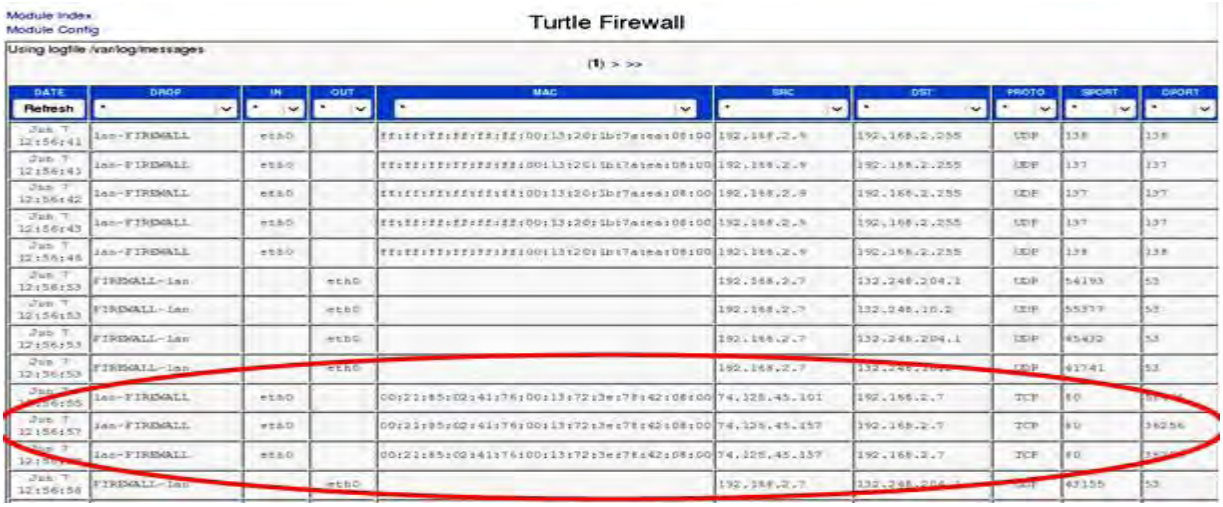


Figura 5.9, Registro del *firewall*, puerto 80.

En los registros se muestran cuatro conexiones en particular establecidas el 7 de junio desde varias direcciones *IP* entre ellas la 74.125.45.101 y la 74.125.45.157, a través de la interfaz eth0 hacia un host de nuestra red interna (192.168.2.7), con puerto de destino 80.

Ahora se verificará que la conexión por medio de SSH sea posible desde un host dentro de la red hacia el *firewall*, esto para poder llevar a cabo la administración del *firewall*.

Se realizó la conexión por medio de SSH con éxito y únicamente revisaremos que la actividad se haya registrado, figura 5.10, Registro del *firewall*, puerto 22.

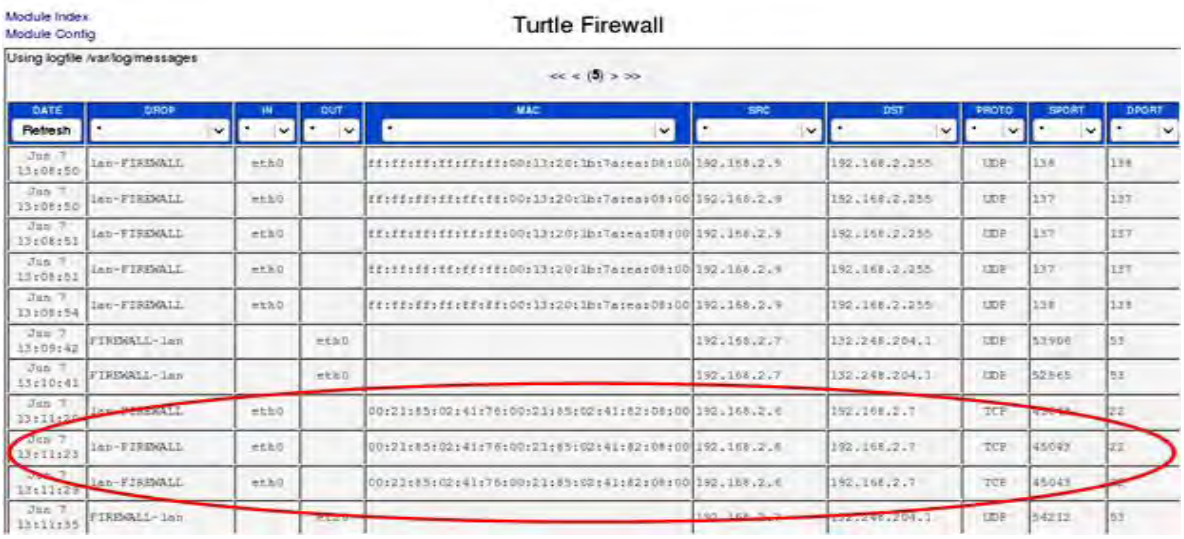


Figura 5.10 Registro del *firewall*, puerto 22.

Se registró la actividad de conexión al *firewall* por medio de SSH el 7 de junio a las 13:22 hrs, a través de la interfaz eth0, desde un equipo que pertenece a la red interna (ip 192.168.2.6) hacia el *firewall* (ip 192.168.2.7), con el protocolo *TCP* y con puerto de destino 22.

Toda la actividad de otros servicios, excepto web, fue bloqueada por el *firewall*, por lo que se considera que para la configuración creada, el *firewall* funcionó correctamente y el registro de las actividades en el *firewall* también se realizó en forma adecuada, la prueba por lo tanto fue exitosa.

5.7 EL SISTEMA DE DETECCIÓN DE INTRUSOS

Nombre:	**Prueba del Sistema de Detección de Intrusos.**
Descripción: Se crearán reglas de detección para Snort, se activará Barnyard para que interprete la actividad registrada en las bitácoras y con la consola de BASE se examinarán los resultados obtenidos.	
Actores: Usuario con privilegios que activará Snort y Barnyard y el usuario sin privilegios que creará cierta actividad en el sistema que debe ser detectada por el Sistema de Detección de Intrusos.	
Precondiciones: Se debe tener la base de datos funcionando, después se crearán un par de reglas para detectar si un usuario está visitando determinados sitios web y una regla para alertar sobre un escaneo de puertos mediante el comando nmap.	
Flujo normal: El usuario sin privilegios visita los sitios web y realiza un escaneo de puertos con nmap, el SDI registra los sucesos y muestra las alertas al administrador, además de registrarlas en sus bitácoras. Finalmente muestra los sucesos mediante la consola de BASE.	
Flujo alternativo: El SDI no detecta la actividad definida en las reglas de Snort.	
Poscondiciones: Si el SDI detectó las actividades definidas en las reglas de Snort, concluye la prueba, de otra manera se deberá verificar si están mal escritas las reglas o si aun estando bien escritas Snort no las detectó como se esperaba.	

Para probar el Sistema de Detección de Intrusos, se crearon dos reglas que permiten verificar si Snort detecta la actividad descrita en las mismas. Dos reglas alertarán sobre la visita de los usuarios a los sitios web www.google.com y www.youtube.com y la tercera alertará sobre un escaneo de puertos con nmap, esta última regla ya está hecha pero se debe activar para que funcione. Para ello, se edita el archivo local.rules que se encuentra en el directorio /etc/snort/rules y se agregan las siguientes líneas:

alert tcp any any -> any any (content:"http://www.youtube.com";msg:"alguien esta visitando youtube";sid:123124;)
alert tcp any any -> any any (content:"www.google.com";msg:"alguien esta visitando google";sid:123123;)

Una vez creadas las reglas, el administrador del SDI debe levantar el servicio Snort y Barnyard, después el usuario sin privilegios debe abrir un navegador de Internet y visitar los sitios descritos en las reglas, una vez hecho lo anterior, se abre una consola de comandos y se ejecuta el comando nmap al servidor local.
Una vez que se creó esta actividad en el sistema, el SID detectó la actividad e inmediatamente mostró los siguientes resultados (figura 5.11, Alertas Snort):

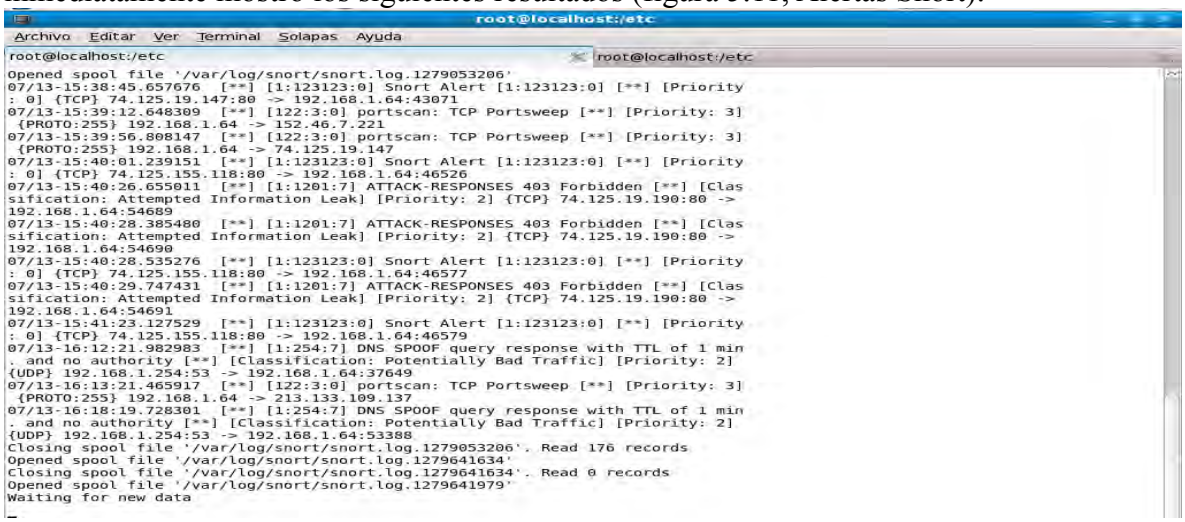


Figura 5.11, Alertas Snort

Como resultado se muestra la fecha en que se efectuó la actividad que generó la alerta, su identificador, el tipo de actividad y la dirección de origen y destino y puerto de origen y destino.

Por otro lado, BASE muestra el tráfico *TCP*, *UDP* y de exploración de puertos, figura 5.12, BASE.

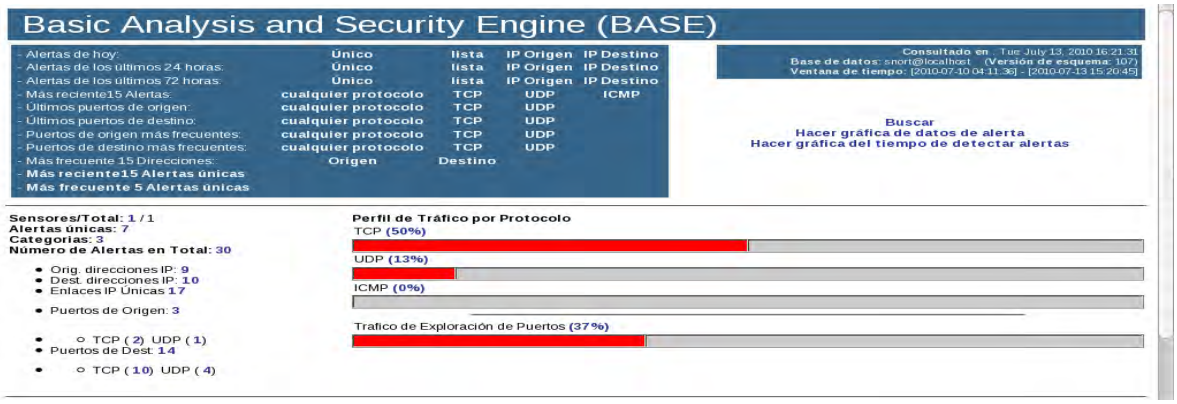


Figura 5.12, BASE.

BASE muestra las alertas generadas por Snort, en este caso, el identificador de la alerta, la firma, la fecha y hora en que se generó la alerta, dirección origen y destino, así como el puerto y el protocolo utilizado, figura 5.13, Alertas BASE.

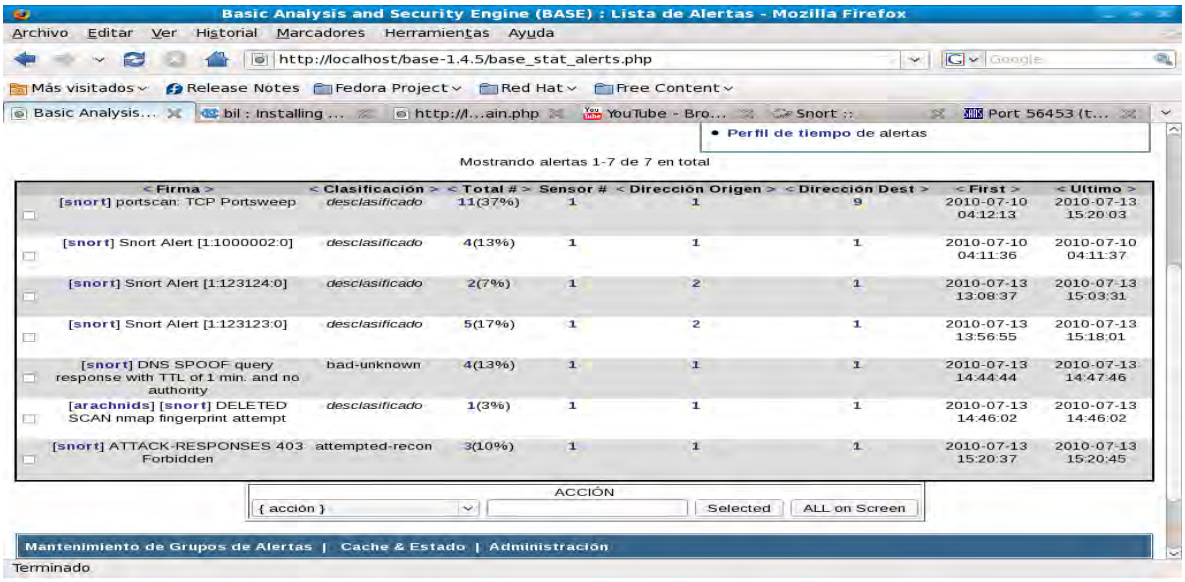


Figura 5.13, Alertas BASE.

Uno de los logs de Barnyard mostró la siguiente actividad:

09/29-15:01:36.045066 [**] [1:123123:0] Snort Alert [1:123123:0] [**] [Priority: 0] {TCP} 66.102.7.104:80 -> 192.168.1.64:49336

09/29-15:02:16.596023 [**] [122:3:0] portscan: TCP Portsweep [**] [Priority: 3] {PROTO:255} 192.168.1.64 -> 87.98.229.71

09/29-15:03:11.480726 [**] [122:3:0] portscan: TCP Portsweep [**] [Priority: 3] {PROTO:255} 192.168.1.64 -> 74.125.127.191

09/29-15:03:36.679786 [**] [122:3:0] portscan: TCP Portsweep [**] [Priority: 3] {PROTO:255} 192.168.1.64 -> 69.63.189.31

09/29-15:15:31.447242 [**] [122:3:0] portscan: TCP Portsweep [**] [Priority: 3] {PROTO:255} 192.168.1.64 -> 76.13.220.49

Las alertas TCP mostraron lo siguiente (figura 5.14, Alertas TCP):

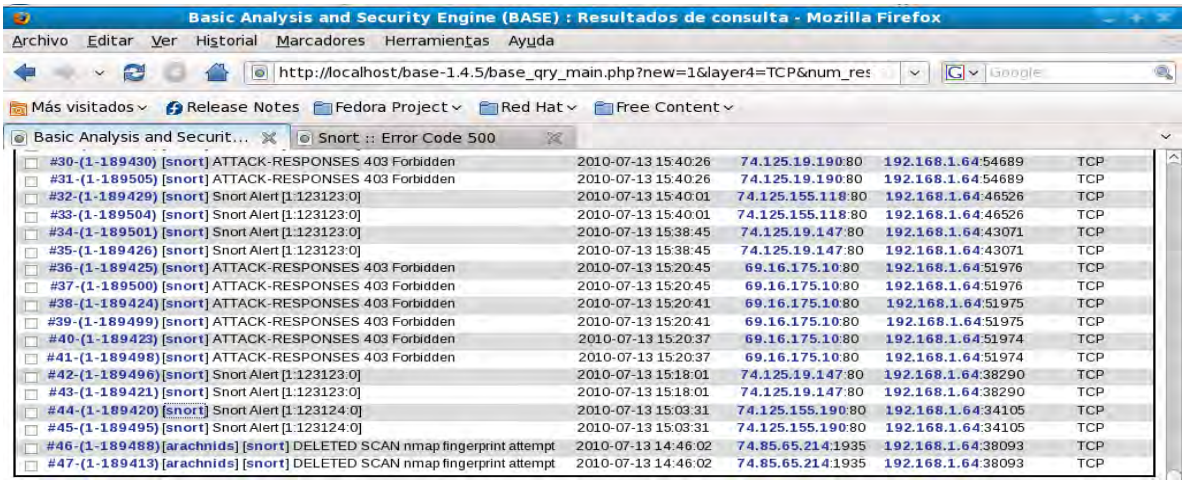
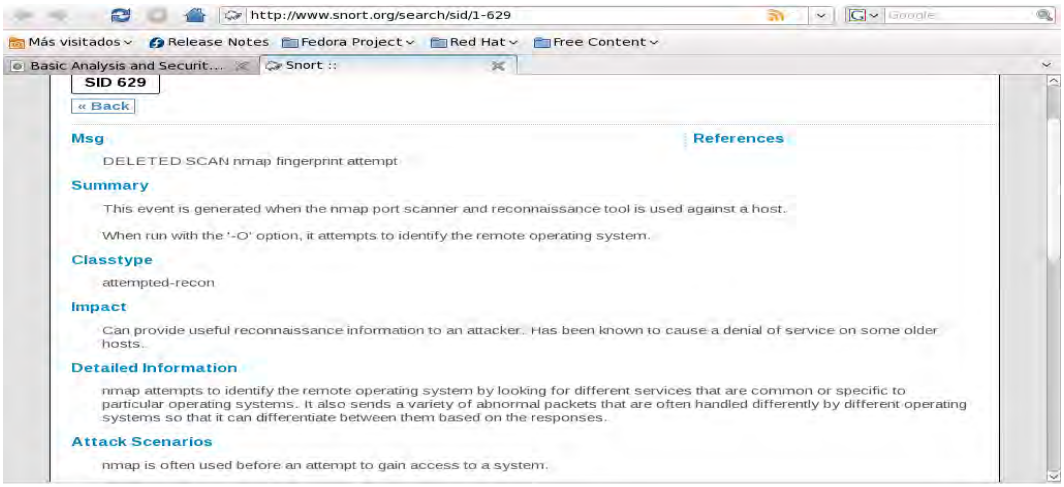


Figura 5.14, Alertas TCP

BASE muestra varias alertas “Atack response”, un intento de acceso a un objeto no autorizado en el servidor, alertas creadas con el SID 123123 que son las generadas por una de las reglas creadas, y alertas generadas por un escaneo de puertos con nmap.
Snort brinda información sobre las alertas incluidas en la configuración por default, como se muestra en la figura 5.15 Snort, alertas en web:



5.15 Snort, alertas en web.

RESULTADOS:
En esta prueba Snort detectó la actividad creada por el usuario y alertó adecuadamente sobre dicho suceso, por una parte Barnyard en modo texto y por otra BASE en modo gráfico mostraron información sobre las páginas visitadas como su dirección ip, los puertos utilizados y la hora exacta en que se visitaron, también se detectó el escaneo de puertos con nmap y un barrido de puertos al servidor de fedora. Esta información es de vital importancia para un administrador de seguridad ya que le permite tomar decisiones sobre eventos casi de manera instantánea.

Como se observó en esta prueba, muchas reglas han sido creadas para trabajar con Snort desde su instalación, en este caso, el escaneo de puertos con nmap fue detectado por una regla prediseñada que se cargó en el momento de la instalación de Snort. Hay vulnerabilidades para las que aun no existen reglas, el administrador debe ser muy cuidadoso y crear reglas que protejan al sistema, como las dos reglas creadas anteriormente, para ello se debe estar al tanto de las últimas noticias en el plano de la seguridad.

Debido a lo anteriormente descrito, se considera que el SDI funciona adecuadamente y que puede detectar firmas de amenazas conocidas y generadas por el administrador de seguridad y alertar oportunamente sobre estos eventos con toda la información necesaria y suficiente para la toma de decisiones de seguridad, por lo tanto se considera que la prueba tuvo éxito.

En la siguiente tabla 5.1, se muestran las pruebas realizadas y los elementos del sistema de defensa que fueron probados:

Tabla 5.1, Pruebas realizadas

Nombre de la prueba	**Elementos del sistema analizado**
John the Ripper	Contraseñas
Nessus	Vulnerabilidades del sistema operativo y la red
Nikto	Vulnerabilidades web
Wireshark	Tráfico de la red
Turtle *Firewall*	*Firewall*
Sistema de Detección de Intrusos	Firmas y reglas de Snort

CAPÍTULO 6

PRÁCTICA

REFORZAMIENTO DE LA SEGURIDAD CON BASTILLE (*HARDENING*)

PRÁCTICA, REFORZAMIENTO DE LA SEGURIDAD CON BASTILLE (*HARDENING*)

1.- OBJETIVOS DE APRENDIZAJE

Al terminar esta práctica, el alumno:

- Comprenderá la importancia de la herramienta de seguridad Bastille Linux, para el reforzamiento del sistema operativo.
- Será capaz de realizar la instalación de Bastille Linux.
- Podrá configurar Bastille Linux para reforzar el sistema operativo de acuerdo con las políticas de seguridad que requiera.

2.- Conceptos teóricos

Bastille Linux es una herramienta de seguridad para el reforzamiento del sistema operativo. Se trata de un conjunto de secuencias de comandos que permiten realizar la configuración del sistema de manera simplificada, de manera que podemos obtener que nuestro sistema operativo sea lo menos vulnerable posible. Mediante una buena configuración de Bastille Linux se pueden eliminar un gran número de vulnerabilidades comunes en plataformas Linux/Unix.

De manera interactiva Bastille Linux crea una configuración segura para el sistema basado en las respuestas del usuario.

Con Bastille se pueden realizar cuatro pasos para asegurar el sistema:

1 Aplicar un *firewall* para prevenir el acceso a posibles servicios vulnerables.

2 Aplicar actualizaciones para los agujeros de seguridad conocidos.

3 Realizar un SUID, root audit (establecer permisos de acceso a archivos y directorios).

4 Desactivar o restringir servicios innecesarios.

Los anteriores pasos se subdividen en módulos que cubren cada una de las áreas anteriormente descritas.

- Módulo ipchains: permite la configuración de un *firewall* para filtrar el tráfico y así proteger la red.

- Módulo patch download: ayuda a mantener actualizados los servicios que se usan en el servidor. Lo más importante es aplicar actualizaciones a los agujeros de seguridad conocidos.

- Módulo file permissions: este módulo permite realizar una auditoría sobre los permisos de archivos en el sistema. Restringe o habilita permisos para el uso de servicios y archivos binarios.

- Módulo account security: con este módulo se obliga al administrador a tener "buenas prácticas" en el manejo de cuentas, además de que provee algunos trucos para realizar de una manera más efectivo el manejo de cuentas.

- Módulo boot security: en este módulo podemos habilitar ciertas opciones de arranque para hacerlo más seguro y evitar que sean aprovechadas las vulnerabilidades inherentes a los sistemas operativos tipo Linux/Unix.

- Módulo secure inetd: con este módulo podemos deshabilitar servicios que podrían significar una vulnerabilidad grave al poder ser iniciados con privilegios de root. En caso de que no se pueda deshabilitar dichos servicios por ser necesarios para los usuarios o para el sistema, entonces se restringen los privilegios. En este caso se recomienda deshabilitar telnet, ftp, rsh, rlogin y talk. Pop e imap son protocolos de correo que deben ser deshabilitados únicamente si no se requieren en el sistema.

- Módulo disable user tools: en este módulo podemos restringir el uso del compilador para que únicamente sea accesible para el usuario root.

- Módulo configure misc pam: partiendo de la idea de que un servidor debe tener únicamente los servicios y herramientas necesarias, en este módulo se pueden hacer ciertas modificaciones que harán difícil a los usuarios, incluidos los usuarios "nobody", "web" y "ftp", que abusen de los recursos para producir un ataque de *denegación de servicios*.

- Módulo logging: aquí manejamos y configuramos las bitácoras adicionales que contienen información del sistema, mensajes del kernel, mensajes de errores graves, etc.

- Módulo miscellaneous daemons: en este módulo se desactivan todos los demonios del sistema que no sean necesarios, los cuáles se pueden volver a activar con el comando chkconfig.

- Módulo sendmail: permite deshabilitar comandos sendmail que son usados para obtener información acerca del sistema para realizar cracking o spamming.

- Módulo remote access: el acceso remoto se puede configurar de manera que se realice mediante *secure shell* client, un programa seguro de cifrado de información.
- Los módulos dns, *apache*, printing y ftp permiten configurar cada uno de los servicios que previamente se han asegurado en caso de que no se haya deshabilitado y establecer cierta seguridad en la forma de levantar dichos servicios.

Todos los módulos mencionados anteriormente se establecen al momento de responder a las preguntas que el script de Bastille realiza al ejecutarse, por lo que se simplifica la tarea de configuración de esta herramienta, sin embargo, es muy importante contestar dicho cuestionario de acuerdo con las políticas de seguridad establecidas por cada institución. En este caso se consideran los aspectos que generalmente se aseguran en un sistema tipo Linux, multiusuario y con características de servidor web.

3.- MATERIAL NECESARIO

Computadora con sistema operativo Linux con alguno de los siguientes sistemas operativos soportados por bastille:

LINUX:
'DB2.2' 'DB3.0' 'RH6.0' 'RH6.1' 'RH6.2'
'RH7.0' 'RH7.1' 'RH7.2' 'RH7.3' 'RH8.0'
'RH9' 'RHEL5' 'RHEL4AS' 'RHEL4ES' 'RHEL4WS'
'RHEL3AS' 'RHEL3ES' 'RHEL3WS' 'RHEL2AS' 'RHEL2ES'
'RHEL2WS' 'RHFC1' 'RHFC2' 'RHFC3' 'RHFC4'
'RHFC5' 'RHFC6' 'RHFC7' 'RHFC8' 'MN6.0'
'MN6.1 ' 'MN7.0' 'MN7.1' 'MN7.2' 'MN8.0'
'MN8.1' 'MN8.2' 'MN10.1' 'SE7.2' 'SE7.3'
'SE8.0' 'SE8.1' 'SE9.0' 'SE9.1' 'SE9.2'
'SE9.3' 'SE10.0' 'SE10.1' 'SE10.2' 'SE10.3'
'SESLES8' 'SESLES9' 'SESLES10' 'TB7.0'

Donde DB significa Debian, RH Red Hat, RHEL Red Hat Enterprise Linux, RHFC Red Hat Fedora Core, MN Mandriva, SE SUSE, SESLES SUSE Linux Enterprise Server.

Privilegios para realizar la instalación en dicho equipo, archivo Bastille-x.x.x.tar.bz2.

4.- DESARROLLO

4.1 Modo de trabajo

Al principio se revisará el estado de algunos elementos importantes del sistema operativo antes de ejecutarse Bastille, posteriormente se ejecutará Bastille y se realizará la configuración recomendada más adelante en esta práctica. Finalmente se comprobará que se hayan realizado los cambios de acuerdo con la configuración realizada.

4.2 Revisión del estado del sistema.

Abra una terminal o Shell (Main Menu, System Tools=> Terminal), y ejecute los siguientes comandos:

```
ls -l /bin/mount
-rwsr-xr-x 1 root root 52012  abr 6      2007        /bin/mount

ls -l /bin/ping
-rwsr-xr-x  1   root     root     36140   abr 6  2007 /bin/ping

ls -l /usr/bin/at
-rwsr-xr-x 1 root    root        45380  mar  27  2007  /usr/bin/at
```

Lo que se puede ver tras la ejecución del comando “ls” en los tres casos es los comandos mount, ping y at tienen el bit SUID activado. Este bit permite a un programa ser ejecutado por un usuario con los permisos de superusuario. Si bien se tiene una mejora por la facilidad de uso, esto reduce la seguridad, ya que algún usuario podría adquirir los derechos de superusuario sin estar autorizado. Se sugiere eliminar este ya que representa un problema de seguridad.

Otros programas que se consideran inseguros son rcp, rlogin, rsh, debido a que no cifran los datos que envían a través de la red y utilizan como método de autenticación únicamente la dirección *IP*, algo que se considera inadecuado. Al listar los comandos de dichos programas se puede observar que también tienen activado el bit SUID.

```
ls –l /usr/bin/rlogin
-rwsr-xr-x  1  root     root     14388  abr      11       2007    /usr/bin/rlogin
ls –l /usr/bin/rsh
-rwsr-xr-x  1   root     root     8940    abr      11       2007    /usr/bin/rsh
ls –l /usr/bin/rcp
-rwsr-xr-x  1   root     root     18640  abr      11       2007    /usr/bin/rcp
ls –l /usr/bin/rexec
-rwsr-xr-x   1  root     root     14300  abr      11       2007    /usr/bin/rexec
```

Otro factor importante es la máscara de usuario (umask), una función que establece los permisos por defecto para los nuevos archivos y directorios creados. Se puede ver la

máscara establecida al ejecutar el comando umask, en este caso la máscara es 0022, lo que nos dice que los nuevos archivos y directorios creados tendrán permisos 755.

```
umask
0022
```

Ahora ejecute el comando ulimit para verificar el número de procesos que puede ejecutar un usuario.

```
ulimit
unlimited
```

Esto también representa una vulnerabilidad ya que se puede ejecutar un ataque de *denegación de servicios* para colapsar el sistema.

Los aspectos analizados anteriormente son una pequeña parte del *hardening* (reforzamiento de la seguridad) que se debe realizar en el sistema. En Linux, esto se realiza mediante el uso de comandos y la edición de los archivos de configuración correspondiente; con Bastille Linux ya no es necesario saber exactamente que comando hay que ejecutar y que archivos modificar para asegurar el sistema, esto representa una gran ventaja si se sabe que realiza Bastille al aplicar la configuración del sistema.

4.3 Instalación de Bastille Linux

Encienda la computadora y elija arrancar con el sistema operativo Linux.

- Abra una terminal o Shell (Main Menu, System Tools=> Terminal)

  Se requieren permisos de administrador para hacer la instalación de Bastille por lo que ejecutaremos el siguiente comando:

  su – root

  ahora ingresaremos la contraseña proporcionada por el profesor.

- A continuación usamos el comando yum (Yellow Dog Updater Modified), una herramienta para el manejo de paquetes en Linux, para instalar perl-Curses y perl-TK, dos librerías que nos permiten crear interfaces basadas en texto e interfaces gráficas. Ejecuté los siguiente comandos:

  *yum install perl-Curses**

  *yum install perl-Tk**

- En seguida copiamos el archivo de Bastille al directorio /usr/local, lo descomprimimos y realizamos su instalación con los siguientes comandos:

bzip2 -d Bastille-x.x.x.tar.bz2

tar xvf Bastille-x.x.x.tar

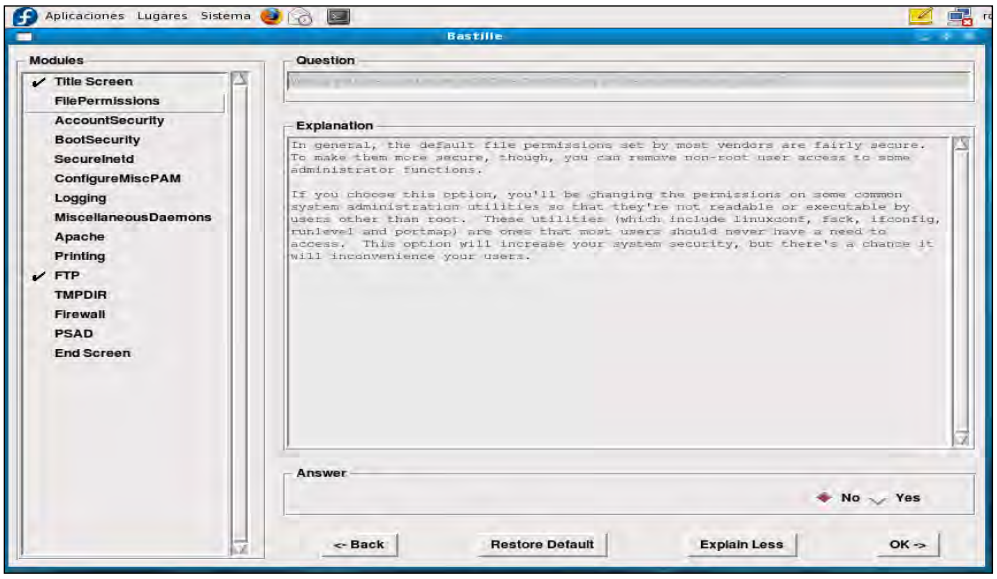
cd Bastille

sh Install.sh

bastille –x

accept

- Al ejecutar el comando "bastille –x" se abre la interfaz de configuración de Bastille Linux y podemos comenzar a configurar Bastille Linux, figura 6.1.



6.1, Configuración de Bastille Linux

En esta práctica se realizará la configuración de Bastille para asegurar el sistema operativo sin incluir la configuración del *firewall*, ya que se considera que el *firewall* es una parte del sistema de defensa tan importante que requiere una instalación y configuración independiente.

Ahora, por cada pregunta que nos presenta Bastille Linux, tenemos que ingresar una respuesta del tipo "Y" (si) o "N" (no), lo cual haremos a continuación, además, se deben completar las razones por las cuáles se responde Y o N, en su caso, donde aparezca la línea contínua:

FILE PERMISSSIONS

- ***Would you like to set more restrictive permissions on the administration utilities? [N]***

  Útil en máquinas con múltiples cuentas de usuario pero en este caso se trata de una máquina dedicada para establecer el sistema de seguridad.

- ***Would you like to disable SUID status for mount/umount? [Y]***
- ***Would you like to disable SUID status for ping? [Y]***
- ***Would you like to disable SUID status for at? [Y]***

  Se deshabilitarán los programas SUID para evitar que usuarios sin privilegios ejecuten programas con permisos de administrador. El atributo setuid es un permiso que se asigna a archivos o directorios, para ser ejecutados por los usuarios del sistema con privilegios elevados temporalmente, para hacer una tarea específica.

- ***Would you like to disable the r-tools? [Y]***

  r-tools son un conjunto de utilidades para la administración remota de equipos que usan las direcciones *IP* como método de autenticación y no usan encriptación para el intercambio de información.

- ***Would you like to disable SUID status for usernetctl? [Y]***

ACCOUNT SECURITY

- ***Should Bastille disable clear-text r-protocols that use IP-based authentication? [Y]***

  Las llamadas r-tools son un conjunto de utilidades para la administración remota de equipos. El problema con ellas es que no usan encriptación para el intercambio de datos y usan las direcciones IP como método de autentificación. Esta carencia de confidencialidad y la facilidad para falsificar las direcciones *IP* de origen han llevado a que se desaconseje el uso de las r-tools y se usen en su lugar alternativas más seguras.

- ***Would you like to enforce password aging? [Y]***

  Se habilitará un tiempo de caducidad para las contraseñas, por default 180 días. Se deberá cambiar la contraseña antes de este plazo, de otra forma la cuenta será bloqueada.

- ***Do you want to set the default umask? [Y]***

El umask son los permisos por defecto que el sistema asigna a los archivos que se van creando.

- ***What umask would you like to set for users on the system? [077]***

  Continuación de la pregunta anterior, lo mejor es usar la opción **077** para que únicamente el dueño de los archivos y nadie más pueda escribir o leer sobre ellos.

- ***Should we disallow root login on ttys 1-6 ? [N] -***

  Esta opción es extremadamente útil con equipos a los que se pueda acceder por SSH sin limitación en cuanto a la IP de origen. En este caso no se limitará el acceso ya que se considera que la configuración del SSH lo contempla.

BOOT SECURITY

- ***Would you like to password-protect the GRUB prompt? [N]***

  Si se tiene acceso físico al equipo cualquier persona podría reiniciar el equipo y obtener una consola de administrador pasándole ciertos parámetros al GRUB, por lo que se debe proteger con una contraseña. Para efectos de la práctica responderemos con [N].

- ***Would you like to password protect single-user mode? [N]***

  Este modo (mono-usuario) permite arrancar el sistema de manera que solamente puede acceder el administrador del equipo. Generalmente no solicita autenticación por lo que se recomienda proteger este modo mediante una contraseña.

  En esta práctica responderemos con [N].

SECURE INETD

- ***Would you like to set a default-deny on TCP Wrappers and xinetd? [N]***

  Se permitirá que se ejecuten estos servicios de conectividad a internet.

  TCP Wrappers es un sistema que se usa para filtrar el acceso de red a servicios de protocolos de Internet. Xinetd es un servicio de Linux que sirve para administrar la conectividad basada en Internet.

- ***Should Bastille ensure the telnet service does not run on this system? [Y]***

Se desactivará el servicio telnet ya que es obsoleto y supone un riesgo grave de seguridad del sistema al transmitir datos sin cifrarlos.

- ***Should Bastille ensure inetd's FTP service does not run on this system? [Y]***

  Lo mismo que telnet, ftp es un servicio inseguro, por lo que se deshabilitará.

- ***Would you like to display "Authorized Use" messages at log-in time? [N]***

  Esta opción se puede activar para mostrar un mensaje con las restricciones para uso del sistema.

  Si la respuesta fuera [Y] Bastille preguntará el nombre del responsable del uso del equipo, con lo que creará un mensaje que se alojará en /etc/issue, mensaje que se puede modificar de acuerdo con las especificaciones de la organización a la que pertenece el equipo que se está configurando.

CONFIGURE MISCPAM

- ***Would you like to put limits on system resource usage?[Y]***

  Con esta medida establecemos un límite de 150 procesos por usuario, suficiente para trabajar y evita un ataque por denegación de servicio.

  AL DAR CLICK EN OK => System rosource limits have been set in the file /etc/security/limits.conf,which you can edit later as necessary.

- ***Should we restrict console access to a small group of user accounts? [N]***

  Permite denegar el acceso a la consola excepto a un grupo determinado de cuentas.

- ***Would you like to add aditional logging? [N]***

  Aunque no representa un peligro tener dos registros adicionales, no se considera necesario en este equipo.

LOGGING

- ***Do you have a remote logging host? [N]***

  Solo se activa en caso de tener un host remoto.

- ***Would you like to set up process accounting? [N]***

  Esta opción permite establecer la contabilidad de procesos y saber quién ejecutó cierto proceso y cuando, aunque parece útil esto puede consumir muchos recursos del sistema y crear bitácoras muy grandes en poco tiempo.

MISCELLANEOUS DAEMONS

- ***Would you like to deactivate NFS and Samba? [Y]***

  El Network File System *(*Sistema de archivos de red*)*, o NFS, es un protocolo utilizado para sistemas de archivos distribuidos. Posibilita que distintos sistemas conectados a una misma red accedan archivos remotos como si se tratara de locales. Samba es una implementación libre del protocolo de archivos compartidos de Windows, antiguamente llamado SMB, fue renombrado recientemente para sistemas de tipo Unix.

- ***Would you like to deactivate the HP OfficeJet (hpoj) script on this machine? [Y]***

  Este script inicia el sistema de soporte en Linux para los dispositivos HP all in one.

- ***Would you like to deactivate ISDN script on this machine? [Y]***

  ISDN es un método para conectar equipos a Internet, con una velocidad de 128 kbps ha sido sustituido por IDSL otro tipo de conexión más rápida.

APACHE

- ***Would you like to deactivate the Apache web server? [N]***

  En este responderemos con N, pero si el equipo no se utiliza como servidor web debe deshabilitarse.

  Se puede habilitar nuevamente con el comando /sbin/chkconfig httpd on

- ***Would you like to bind the Web server to listen only to the localhost? [N]***

  Esto es muy útil cuando se hace desarrollo web, ya que permite editar un sitio web localmente antes de que se cargue en otro servidor.

- ***Would you like to bind the Web server to a particular interface? [N]***

En caso de que se permita el acceso al servidor web únicamente a la red interna se tiene que contestar [Y] e ingresar la ip del equipo que se quiere asociar y el puerto que debe escuchar.

- ***Would you like to deactivate the following of symbolic links? [Y]***

  En general trataremos de limitar la información del servidor web que puede ser vista por los usuarios. Al desactivar la opción de seguimiento de ligas simbólicas se previene el que un visitante web pueda leer o modificar archivos que no se encuentran en el directorio de publicación web.

TMPDIR

- ***Would you like to install TMPDIR/TMP scripts? [N]***

  Esto nos permitiría crear directorios temporales alternativos al directorio /tmp con la ventaja de ejecutar scripts cuando los usuarios accedieran a dichos directorios.

  Activar esta opción hará que Bastille instale unos scripts en las cuentas de los usuarios que configuren las variables TMPDIR y TMP de tal manera que utilicen directorios de ficheros temporales completamente individuales, en vez de que todos usen el */tmp* lo que puede ser extremadamente peligroso en entornos multiusuarios.

PRINTING

- ***Would you like to disable printing? [Y]***

  No usaremos impresoras en este equipo por lo tanto deshabilitamos las funciones de impresión, ya que dejarlas activas supone un riesgo de seguridad.

  Esto se puede revertir con los siguientes comandos:

  /bin/chmod 06555 /usr/bin/lpr /usr/bin/lprm

  /sbin/chkconfig lpd on

- ***Would you like to disable CUPS'legacy LPD support? [N]***

  Para efectos de la práctica contestaremos [N], pero si el equipo realizará funciones de impresión, se puede deshabilitar el soporte para el protocolo LPD del sistema de impresión común de Unix.

FTP

FIREWALL

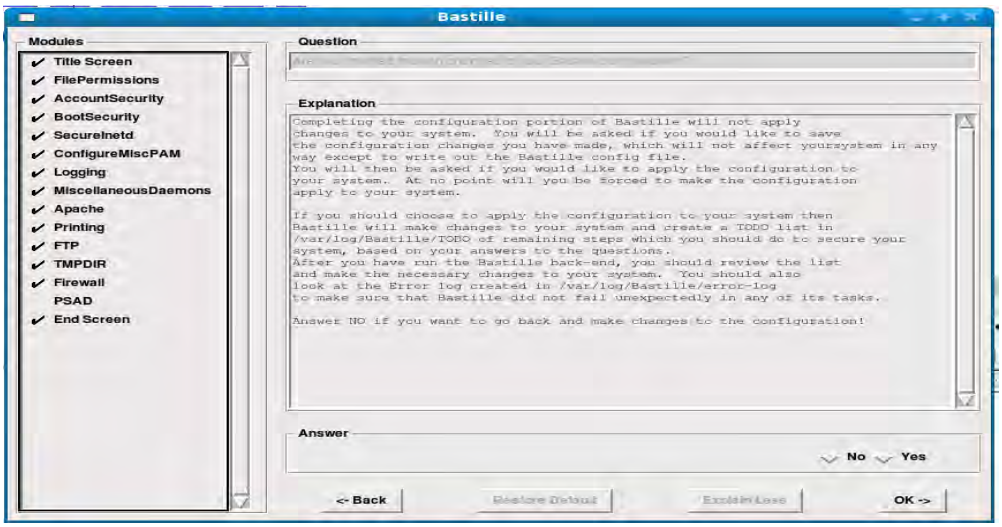
- ***Would you like to run the packet filtering script? [N]***

  Esta opción activaría el *firewall* nativo de Linux, en este caso no se activará porque se recomienda que el *firewall* se configure por separado y con otras herramientas específicas para la configuración del mismo.

- ***Are you finished answering the questions, i.e. may we make the changes? [Y]***

  Respondemos afirmativamente para que se apliquen los cambios en el sistema.

  Guardar la configuración y después aplicarla al sistema, figura 6.2.



6.2, Guardar configuración.

Se ha terminado de configurar Bastille Linux, además de aplicarse los cambios en el sistema operativo.

Ahora con una cuenta de usuario sin privilegios probaremos que la configuración anterior funciona correctamente, en este caso probaremos ciertos puntos en el sistema operativo, que deben estar protegidos por la configuración hecha con Bastille Linux:

Los comandos mount, ping y at no deben tener activo el bit SUID

ls -l /bin/mount
-rwr-xr-x 1 root root 52012 abr 6 2007 /bin/mount

```
ls -l /bin/ping
-rwr-xr-x  1    root    root    36140   abr 6  2007 /bin/ping

ls -l /usr/bin/at
-rwr-xr-x 1 root    root       45380  mar  27  2007  /usr/bin/at
```

Otros programas que inseguros como rcp, rlogin, rsh también deben tener desactivado el bit SUID.

```
ls –l /usr/bin/rlogin
-rwr-xr-x  1  root      root    14388  abr     11      2007    /usr/bin/rlogin
ls –l /usr/bin/rsh
-rwr-xr-x  1    root    root    8940    abr     11      2007    /usr/bin/rsh
ls –l /usr/bin/rcp
-rwr-xr-x  1    root    root    18640  abr     11      2007    /usr/bin/rcp
ls –l /usr/bin/rexec
-rwsr-xr-x   1  root    root    14300  abr     11      2007    /usr/bin/rexec
```

La máscara de usuario (umask), función que establece los permisos por defecto para los nuevos archivos y directorios creados debe tener el nuevo valor (077)

```
umask
0077
```

El número de procesos por usuario debe estar restringido a 150, y eso se comprueba con el comando ulimit.

```
ulimit -a
max user processes    (-u) 150
```

Por lo que el sistema se encuentra protegido si un ataque de saturación de buffer o *denegación de servicios* intenta hacer una replicación infinita de procesos.

El reforzamiento de la seguridad con Bastille Linux se debe verificar en otros aspectos que en este caso no fueron analizados, para ello es necesario saber que archivos se van a modificar o que comandos se ejecutarán mediante la configuración hecha con este programa, pero al revisar estos aspectos básicos podemos darnos una idea de las ventajas de utilizar un software de este tipo.

5.- CUESTIONARIO

1.- ¿Cuáles son los cuatro pasos que se pueden hacer con Bastille para asegurar el sistema operativo?

2.- ¿Qué son las r-tools y qué método de autenticación utilizan?

3.- ¿Qué problema de seguridad presenta el servicio Telnet?

4.- ¿Por qué es importante establecer un límite de procesos que puede ejecutar un usuario?

5.- ¿Cuál es la ventaja de utilizar Bastille para asegurar el sistema operativo?

6.- CUESTIONARIO PREVIO

1.- Investigue ¿Qué medidas para reforzar un sistema operativo tipo Linux existen?

2.- ¿Qué función tiene el comando “umask”?

¿qué permisos se deben establecer en un sistema para que se considere seguro?

3.- ¿Qué servicios remotos se consideran inseguros en un sistema tipo Linux y por qué?

4.- ¿Qué es la denegación de servicios?

5.- Investigue ¿Qué herramientas para el reforzamiento del sistema operativo conoce y cuáles son sus características?

CONCLUSIONES

A través de cada uno de los capítulos del presente documento se han enumerado una serie herramientas de seguridad que forman parte del sistema de defensa de una red y que, controlan su acceso, protegen los servicios y los recursos compartidos de la misma y permiten tener un entorno más confiable y seguro.

En la actualidad existen muchos programas que protegen los activos de las organizaciones; antivirus, escáneres de vulnerabilidades, *firewalls* y sistemas de detección de intrusos son algunos de los programas que permiten tener un mejor control sobre la disponibilidad y uso de los recursos de un sistema, todos estos programas de protección responden a una creciente necesidad de saber que está pasando en el sistema y del estado de la información que se resguarda. La simple instalación de dichos programas no garantiza un control adecuado de los recursos de una red; para garantizar un nivel de seguridad aceptable, los encargados de la seguridad deben tener amplios conocimientos sobre redes, sobre los puntos vulnerables de las redes, y de las herramientas que puede utilizar y las ventajas que puede obtener de las mismas.

En el presente trabajo, se hizo la descripción de las herramientas más usadas debido a su eficacia en la protección de ciertas áreas de la red, desde el sistema operativo hasta una arquitectura de múltiples computadoras en red.

Para el reforzamiento de la seguridad del sistema operativo se utilizó *OpenSSH* y Bastille Linux con muy buenos resultados, *OpenSSH* permitió crear conexiones cifradas tanto al equipo en que se implementó el *firewall*, como al equipo en que se implementó el detector de intrusos. El servidor SSH se configuró de tal manera que no se permite el logueo directamente como root, con lo cual se protege dicha cuenta, además se limitaron los intentos fallidos al momento de ingresar la contraseña y el número de terminales que se pueden obtener por sesión. Con esto se logró un mejor control del acceso a los equipos en que se van a desempeñar las labores de monitoreo y administración de la red.

Con Bastille Linux se hizo todo el proceso de reforzamiento del sistema operativo o *hardening*, se deshabilitaron comandos como mount, ping y at, que por defecto están habilitados en los sistemas tipo Linux y que representan vulnerabilidades para la seguridad del sistema, también fueron deshabilitados servicios como telnet y ftp por tratarse de servicios inseguros, se deshabilitaron los servicios de impresión, se limitó el número de procesos a ejecutar por usuario y se estableció una máscara de usuario adecuada, entre otras acciones que Bastille permitió configurar mediante una interfaz muy intuitiva y fácil de usar. Se obtuvieron muchas ventajas con el uso de este software, pero la más importante fue el reforzar el sistema sin la necesidad de ejecutar comandos o editar archivos como generalmente se realiza en los sistemas Linux.

En el área de protección de contraseñas y ataques de fuerza bruta la librería Cracklib y Johh the Ripper son las dos herramientas que se utilizaron para defender el sistema. Cracklib es una librería que forma parte de la instalación por defecto en los sistemas Linux actuales y únicamente se realizó la configuración de la misma para tener una forma de asegurar que las contraseñas de los usuarios de la red fuesen fuertes. Por otro lado John the Ripper, un programa muy eficiente para descifrar contraseñas, permitió detectar contraseñas débiles, lo que brindó una mayor confiabilidad en el sistema, ya que se cubrió el mismo objetivo, la protección de contraseñas, desde dos puntos de vista diferentes, uno como defensor y el otro como atacante.

Para detectar y corregir los errores en configuración del sistema operativo y las vulnerabilidades de las aplicaciones implementadas en el equipo que se utilizó como detector de intrusos, se utilizó Nessus, un escáner de vulnerabilidades muy efectivo que aprovecha todos los fallos en la configuración del sistema, los huecos de seguridad y las ventajas que proporcionan las instalaciones por defecto. Con el uso de Nessus en el equipo se detectaron más de 190 vulnerabilidades, 55 de alto riesgo, que se relacionaron con la falta de actualización del sistema operativo, las vulnerabilidades de riesgo medio y bajo se relacionaron con la actualización de librerías y huecos de seguridad en la configuración del servidor web, esto permitió conocer los puntos débiles del equipo y las aplicaciones y hacer las correcciones necesarias para proteger el sistema.

En la configuración de la red se consideró una zona desmilitarizada con un servidor web; para detectar las vulnerabilidades web en esta zona, se utilizó Nikto, herramienta basada en *Perl* que ayudó a detectar ciertos problemas que tuvo el servidor web al estar expuesto en Internet, como la vulnerabilidad Cross Site Scripting y el problema de tener la indexación de directorios activa. Los resultados obtenidos fueron bastante buenos ya que esta herramienta permitió detectar vulnerabilidades específicas que no fueron detectadas por Nessus.

Una parte muy importante en el sistema de defensa es Wireshark, herramienta con la que se pudo observar el tráfico en la red, origen y destino de dicho tráfico, protocolos utilizados, las conexiones hechas, y se pudo dar el seguimiento a las conexiones establecidas. Toda esta información permite saber que uso le dan los usuarios a los recursos disponibles en la misma, además de determinar actividades que ocupan un ancho de banda muy grande en la red y en general, tener conocimiento de lo que ocurre en la red para implementar un mejor control.

El *firewall* y el Sistema de Detección de Intrusos constituyen dos elementos importantes en el sistema de defensa de la red. Para la creación del *cortafuegos* se utilizó Turtle *Firewall*, programa que facilitó la tarea de crear las reglas de filtrado en el orden correcto, con una interfaz web amigable. Como política se estableció el denegar todo por defecto y se crearon ciertas reglas para permitir el tráfico necesario, como *Secure Shell*, http y https. Con la correcta implementación del *firewall* se logró un mejor control sobre las conexiones entrantes y salientes del entorno de red y se obtuvieron los conocimientos necesarios para crear una configuración adecuada y acorde con los requerimientos de la red.

El Sistema de Detección de Intrusos se implementó con Snort en modo de detección de intrusos, además se utilizó BASE, con lo cuál se logró tener los datos de detección en una base de datos en forma ordenada y organizada y con una interfaz web fácil de usar. Con Snort se logró un control más eficiente sobre las actividades de los usuarios dentro de la red y se pudo aseguró la detección de actividades peligrosas comparándolas con las firmas de ataques conocidos. También se realizaron pruebas sobre la creación de reglas que, aunque no se incluyen como ataques en la base de firmas, se pueden implementar para alertar sobre ciertas actividades en la red, como evitar que se visiten algunos sitios web; es decir, el detector de intrusos también sirve para alertar sobre actividades que no son consideradas como ataques.

Es claro que la correcta implementación de un *firewall* y un Sistema de Detección de Intrusos toma cierto tiempo y práctica y generalmente se pone a prueba su eficacia en ataques reales, cuando la red está expuesta a amenazas constantes, pero durante este trabajo, con base en las pruebas realizadas, se logró verificar el correcto funcionamiento de ambas herramientas y se creó una configuración apropiada para el tipo de servicios que brinda la red.

Finalmente, en el capítulo 6 se implementa una práctica sobre el reforzamiento de seguridad o *hardening* en el sistema operativo con el uso de la herramienta Bastille Linux, está práctica tiene como objetivo que el estudiante comprenda la importancia de la herramienta para el reforzamiento de la seguridad del sistema y para ello se realiza la instalación y configuración de dicha herramienta y se realizan algunas pruebas para verificar su correcto funcionamiento.

Mediante el correcto uso de las herramientas mencionadas se logró conocer los puntos vulnerables de la red y se implementaron las correcciones pertinentes para tener un control y administración segura de los recursos y servicios de la red, así como obtener el conocimiento necesario para la configuración y prueba de las herramientas y lograr su mejor desempeño.

En general, se creó un sistema de defensa completo con base en herramientas de *software libre* que permiten proteger de una manera muy efectiva los activos de una red y que contribuyen a un mejor desempeño de la misma. Estas herramientas constituyen por lo tanto un paquete importante de trabajo para un conocedor de la red y deben ser vistas como herramientas adaptables, que sin duda seguirán desarrollándose tanto como se modifiquen las necesidades de seguridad en cada área de las redes. Asimismo, se puede afirmar que quien decida implementar el sistema de defensa para una red propuesto aquí con herramientas de software libre puede tener la certeza de que como administrador de seguridad con conocimientos suficientes puede lograr una protección aceptable.

En lo personal, el presente trabajo me permitió adquirir conocimientos sobre seguridad de las redes en forma general, y en particular, sobre el tipo de sistemas de seguridad que se pueden implementar en cualquier red, las necesidades de seguridad y la forma de satisfacerlas con herramientas de *software libre*, la instalación y configuración de dichas herramientas para obtener su mejor desempeño y el manejo de la información obtenida para asegurar áreas vulnerables.

La administración de seguridad es un campo de constante actualización que requiere una correcta supervisión e investigación de los nuevos mecanismos de ataque y de defensa de un sistema y de las nuevas tecnologías y sus vulnerabilidades, por lo que se convierte en un campo de estudio muy complejo y muchas veces complicado, pero también lo convierte en un campo muy interesante para aquellas personas que ven la seguridad de los sistemas como un desafío y la solución de problemas como una forma de vida.

ANEXOS

ANEXO 1

PRÁCTICA - ALUMNOS

REFORZAMIENTO DE LA SEGURIDAD CON BASTILLE (HARDENING)

1.- OBJETIVOS DE APRENDIZAJE

Al terminar esta práctica, el alumno:

- Comprenderá la importancia de la herramienta de seguridad Bastille Linux, para el reforzamiento del sistema operativo.
- Será capaz de realizar la instalación de Bastille Linux.
- Podrá configurar Bastille Linux para reforzar el sistema operativo de acuerdo con las políticas de seguridad que requiera.

2.- Conceptos teóricos

Bastille Linux es una herramienta de seguridad para el reforzamiento del sistema operativo. Se trata de un conjunto de secuencias de comandos que permiten realizar la configuración del sistema de manera simplificada, de manera que podemos obtener que nuestro sistema operativo sea lo menos vulnerable posible. Mediante una buena configuración de Bastille Linux se pueden eliminar un gran número de vulnerabilidades comunes en plataformas Linux/Unix.

De manera interactiva Bastille Linux crea una configuración segura para el sistema basado en las respuestas del usuario.

Con Bastille se pueden realizar cuatro pasos para asegurar el sistema:

- Aplicar un firewall para prevenir el acceso a posibles servicios vulnerables.
- Aplicar actualizaciones para los agujeros de seguridad conocidos.
- Realizar un SUID, root audit (establecer permisos de acceso a archivos y directorios).
- Desactivar o restringir servicios innecesarios.

Los anteriores pasos se subdividen en módulos que cubren cada una de las áreas anteriormente descritas.

- Módulo ipchains: permite la configuración de un firewall para filtrar el tráfico y así proteger la red.

- Módulo patch download: ayuda a mantener actualizados los servicios que se usan en el servidor. Lo más importante es aplicar actualizaciones a los agujeros de seguridad conocidos.

- Módulo file permissions: este módulo permite realizar una auditoría sobre los permisos de archivos en el sistema. Restringe o habilita permisos para el uso de servicios y archivos binarios.

- Módulo account security: con este módulo se obliga al administrador a tener "buenas prácticas" en el manejo de cuentas, además de que provee algunos trucos para realizar de una manera más efectivo el manejo de cuentas.

- Módulo boot security: en este módulo podemos habilitar ciertas opciones de arranque para hacerlo más seguro y evitar que sean aprovechadas las vulnerabilidades inherentes a los sistemas operativos tipo Linux/Unix.

- Módulo secure inetd: con este módulo podemos deshabilitar servicios que podrían significar una vulnerabilidad grave al poder ser iniciados con privilegios de root. En caso de que no se pueda deshabilitar dichos servicios por ser necesarios para los usuarios o para el sistema, entonces se restringen los privilegios. En este caso se recomienda deshabilitar telnet, ftp, rsh, rlogin y talk. Pop e imap son protocolos de correo que deben ser deshabilitados únicamente si no se requieren en el sistema.

- Módulo disable user tools: en este módulo podemos restringir el uso del compilador para que únicamente sea accesible para el usuario root.

- Módulo configure misc pam: partiendo de la idea de que un servidor debe tener únicamente los servicios y herramientas necesarias, en este módulo se pueden hacer ciertas modificaciones que harán difícil a los usuarios, incluidos los usuarios "nobody", "web" y "ftp", que abusen de los recursos para producir un ataque de *denegación de servicios*.

- Módulo logging: aquí manejamos y configuramos las bitácoras adicionales que contienen información del sistema, mensajes del kernel, mensajes de errores graves, etc.

- Módulo miscellaneous daemons: en este módulo se desactivan todos los demonios del sistema que no sean necesarios, los cuáles se pueden volver a activar con el comando chkconfig.

- Módulo sendmail: permite deshabilitar comandos sendmail que son usados para obtener información acerca del sistema para realizar cracking o spamming.
- Módulo remote access: el acceso remoto se puede configurar de manera que se realice mediante *secure shell* client, un programa seguro de cifrado de información.
- Los módulos dns, *apache*, printing y ftp permiten configurar cada uno de los servicios que previamente se han asegurado en caso de que no se haya deshabilitado y establecer cierta seguridad en la forma de levantar dichos servicios.

Todos los módulos mencionados anteriormente se establecen al momento de responder a las preguntas que el script de Bastille realiza al ejecutarse, por lo que se simplifica la tarea de configuración de esta herramienta, sin embargo, es muy importante contestar dicho cuestionario de acuerdo con las políticas de seguridad establecidas por cada institución. En este caso se consideran los aspectos que generalmente se aseguran en un sistema tipo Linux, multiusuario y con características de servidor web.

3.- MATERIAL NECESARIO

Computadora con sistema operativo Linux con alguno de los siguientes sistemas operativos soportados por bastille:

LINUX:
'DB2.2' 'DB3.0' 'RH6.0' 'RH6.1' 'RH6.2'
'RH7.0' 'RH7.1' 'RH7.2' 'RH7.3' 'RH8.0'
'RH9' 'RHEL5' 'RHEL4AS' 'RHEL4ES' 'RHEL4WS'
'RHEL3AS' 'RHEL3ES' 'RHEL3WS' 'RHEL2AS' 'RHEL2ES'
'RHEL2WS' 'RHFC1' 'RHFC2' 'RHFC3' 'RHFC4'
'RHFC5' 'RHFC6' 'RHFC7' 'RHFC8' 'MN6.0'
'MN6.1 ' 'MN7.0' 'MN7.1' 'MN7.2' 'MN8.0'
'MN8.1' 'MN8.2' 'MN10.1' 'SE7.2' 'SE7.3'
'SE8.0' 'SE8.1' 'SE9.0' 'SE9.1' 'SE9.2'
'SE9.3' 'SE10.0' 'SE10.1' 'SE10.2' 'SE10.3'
'SESLES8' 'SESLES9' 'SESLES10' 'TB7.0'

Donde DB significa Debian, RH Red Hat, RHEL Red Hat Enterprise Linux, RHFC Red Hat Fedora Core, MN Mandriva, SE SUSE, SESLES SUSE Linux Enterprise Server.

Privilegios para realizar la instalación en dicho equipo, archivo Bastille-x.x.x.tar.bz2.

4.- DESARROLLO

4.1 Modo de trabajo

Al principio se revisará el estado de algunos elementos importantes del sistema operativo antes de ejecutarse Bastille, posteriormente se ejecutará Bastille y se realizará la configuración recomendada más adelante en esta práctica. Finalmente se comprobará que se hayan realizado los cambios de acuerdo con la configuración realizada.

4.3 Revisión del estado del sistema.

Abra una terminal o Shell (Main Menu, System Tools=> Terminal), y ejecute los siguientes comandos:

ls -l /bin/mount
-rwsr-xr-x 1 root root 52012 abr 6 2007 /bin/mount

ls -l /bin/ping
-rwsr-xr-x 1 root root 36140 abr 6 2007 /bin/ping

ls -l /usr/bin/at
-rwsr-xr-x 1 root root 45380 mar 27 2007 /usr/bin/at

Lo que se puede ver tras la ejecución del comando “ls” en los tres casos es los comandos mount, ping y at tienen el bit SUID activado. Este bit permite a un programa ser ejecutado por un usuario con los permisos de superusuario. Si bien se tiene una mejora por la facilidad de uso, esto reduce la seguridad, ya que algún usuario podría adquirir los derechos de superusuario sin estar autorizado. Se sugiere eliminar este ya que representa un problema de seguridad.

Otros programas que se consideran inseguros son rcp, rlogin, rsh, debido a que no cifran los datos que envían a través de la red y utilizan como método de autenticación únicamente la dirección IP, algo que se considera inadecuado. Al listar los comandos de dichos programas se puede observar que también tienen activado el bit SUID.

ls –l /usr/bin/rlogin
-rwsr-xr-x 1 root root 14388 abr 11 2007 /usr/bin/rlogin
ls –l /usr/bin/rsh
-rwsr-xr-x 1 root root 8940 abr 11 2007 /usr/bin/rsh
ls –l /usr/bin/rcp
-rwsr-xr-x 1 root root 18640 abr 11 2007 /usr/bin/rcp
ls –l /usr/bin/rexec
-rwsr-xr-x 1 root root 14300 abr 11 2007 /usr/bin/rexec

Otro factor importante es la máscara de usuario (umask), una función que establece los permisos por defecto para los nuevos archivos y directorios creados. Se puede ver la máscara establecida al ejecutar el comando umask, en este caso la máscara es 0022, lo que nos dice que los nuevos archivos y directorios creados tendrán permisos 755.

```
umask
0022
```

Ahora ejecute el comando ulimit para verificar el número de procesos que puede ejecutar un usuario.

```
ulimit
unlimited
```

Esto también representa una vulnerabilidad ya que se puede ejecutar un ataque de *denegación de servicios* para colapsar el sistema.

Los aspectos analizados anteriormente son una pequeña parte del hardening (reforzamiento de la seguridad) que se debe realizar en el sistema. En Linux, esto se realiza mediante el uso de comandos y la edición de los archivos de configuración correspondiente; con Bastille Linux ya no es necesario saber exactamente que comando hay que ejecutar y que archivos modificar para asegurar el sistema, esto representa una gran ventaja si se sabe que realiza Bastille al aplicar la configuración del sistema.

4.3 Instalación de Bastille Linux

Encienda la computadora y elija arrancar con el sistema operativo Linux.

- Abra una terminal o Shell (Main Menu, System Tools=> Terminal)

  Se requieren permisos de administrador para hacer la instalación de Bastille por lo que ejecutaremos el siguiente comando:

  su – root

  ahora ingresaremos la contraseña proporcionada por el profesor.

- A continuación usamos el comando yum (Yellow Dog Updater Modified), una herramienta para el manejo de paquetes en Linux, para instalar perl-Curses y perl-TK, dos librerías que nos permiten crear interfaces basadas en texto e interfaces gráficas. Ejecuté los siguiente comandos:

  *yum install perl-Curses**

  *yum install perl-Tk**

- En seguida copiamos el archivo de Bastille al directorio /usr/local, lo descomprimimos y realizamos su instalación con los siguientes comandos:

  bzip2 -d Bastille-x.x.x.tar.bz2

  tar xvf Bastille-x.x.x.tar

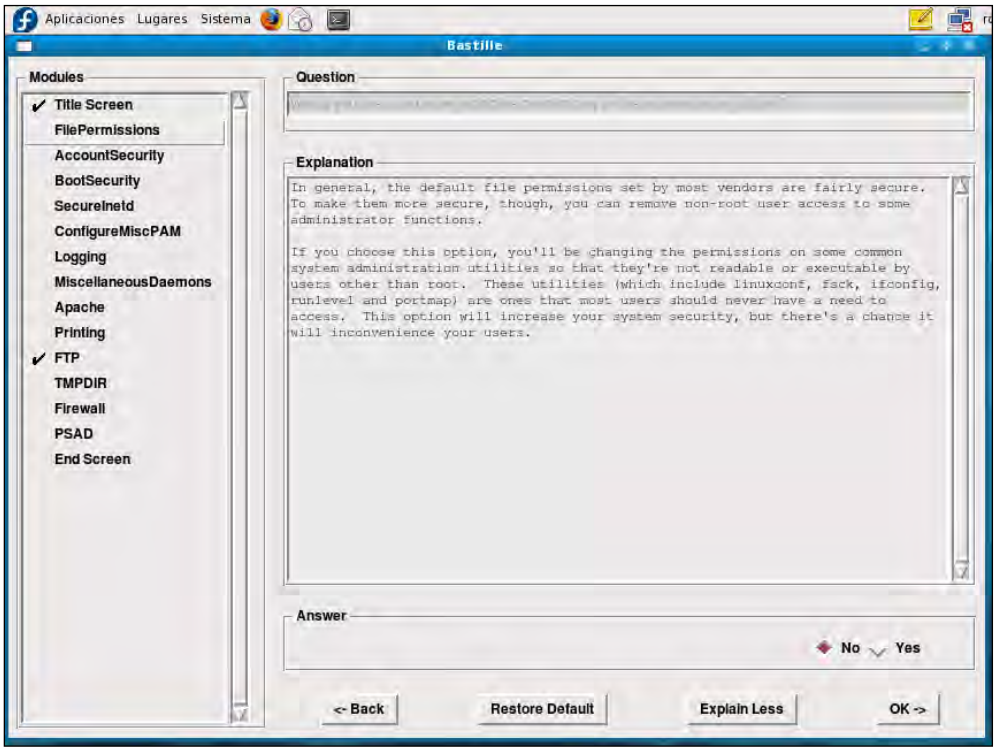
  cd Bastille

  sh Install.sh

  bastille –x

  accept

- Al ejecutar el comando "bastille –x" se abre la interfaz de configuración de Bastille Linux y podemos comenzar a configurar Bastille Linux, figura 6.1.



6.1, Configuración de Bastille Linux

En esta práctica se realizará la configuración de Bastille para asegurar el sistema operativo sin incluir la configuración del firewall, ya que se considera que el firewall es una parte del sistema de defensa tan importante que requiere una instalación y configuración independiente.

Ahora, por cada pregunta que nos presenta Bastille Linux, tenemos que ingresar una respuesta del tipo “Y” (si) o “N” (no), lo cual haremos a continuación, además, se deben completar las razones por las cuáles se responde Y o N, en su caso, donde aparezca la línea contínua:

FILE PERMISSIONS

- ***Would you like to set more restrictive permissions on the administration utilities? [N]***

  Útil en máquinas con múltiples cuentas de usuario pero en este caso se trata de una máquina dedicada para establecer el sistema de seguridad.

- ***Would you like to disable SUID status for mount/umount? [Y]***
- ***Would you like to disable SUID status for ping? [Y]***
- ***Would you like to disable SUID status for at? [Y]***

  Anote la justificación a las respuestas anteriores:

  _____
  _____
  _____
  _____

- ***Would you like to disable the r-tools? [Y]***

  Anote la justificación a la respuesta anterior:

  _____
  _____
  _____

- ***Would you like to disable SUID status for usernetctl? [Y]***

ACCOUNT SECURITY

- ***Should Bastille disable clear-text r-protocols that use IP-based authentication? [Y]***

  Anote la justificación a la respuesta anterior:

_____
_____
_____
_____

- ***Would you like to enforce password aging? [Y]***

  Anote la justificación a la respuesta anterior:

  _____
  _____
  _____

- ***Do you want to set the default umask? [Y]***

  El umask son los permisos por defecto que el sistema asigna a los archivos que se van creando.

- ***What umask would you like to set for users on the system? _____***

  Continuación de la pregunta anterior, lo mejor es usar la opción ____ para que únicamente el dueño de los archivos y nadie más pueda escribir o leer sobre ellos.

- ***Should we disallow root login on ttys 1-6 ? [N]***

  Esta opción es extremadamente útil con equipos a los que se pueda acceder por SSH sin limitación en cuanto a la IP de origen. En este caso no se limitará el acceso ya que se considera que la configuración del SSH lo contempla.

BOOT SECURITY

- ***Would you like to password-protect the GRUB prompt? [N]***

  Si se tiene acceso físico al equipo cualquier persona podría reiniciar el equipo y obtener una consola de administrador pasándole ciertos parámetros al GRUB, por lo que se debe proteger con una contraseña. Para efectos de la práctica responderemos con [N].

- ***Would you like to password protect single-user mode? [N]***

  Este modo (mono-usuario) permite arrancar el sistema de manera que solamente acceder el administrador del equipo. Generalmente no solicita autenticación por lo que se recomienda proteger este modo mediante una contraseña.

En esta práctica responderemos con [N].

SECURE INETD

- ***Would you like to set a default-deny on TCP Wrappers and xinetd? [N]***

  Se permitirá que se ejecuten estos servicios de conectividad a internet.

  TCP Wrappers es un sistema que se usa para filtrar el acceso de red a servicios de protocolos de Internet. Xinetd es un servicio de Linux que sirve para administrar la conectividad basada en Internet.

- ***Should Bastille ensure the telnet service does not run on this system? [Y]***

  Anote la justificación a la respuesta anterior:

  _____
  _____

- ***Should Bastille ensure inetd's FTP service does not run on this system? [Y]***

  Anote la justificación a la respuesta anterior:

  _____

- ***Would you like to display "Authorized Use" messages at log-in time? [N]***

  Esta opción se puede activar para mostrar un mensaje con las restricciones para uso del sistema.

  Si la respuesta fuera [Y] Bastille preguntará el nombre del responsable del uso del equipo, con lo que creará un mensaje que se alojará en /etc/issue, mensaje que se puede modificar de acuerdo con las especificaciones de la organización a la que pertenece el equipo que se está configurando.

CONFIGURE MISCPAM

- ***Would you like to put limits on system resource usage?____***

  Con esta medida establecemos un límite de 150 procesos por usuario, suficiente para trabajar y evita un ataque del tipo _____.

  AL DAR CLICK EN OK => System rosource limits have been set in the file /etc/security/limits.conf,which you can edit later as necessary.

- ***Should we restrict console access to a small group of user accounts? [N]***

  Permite denegar el acceso a la consola excepto a un grupo determinado de cuentas.

- ***Would you like to add aditional logging? [N]***

  Aunque no representa un peligro tener dos registros adicionales, no se considera necesario en este equipo.

LOGGING

- ***Do you have a remote logging host? [N]***

  Solo se activa en caso de tener un host remoto.

- ***Would you like to set up process accounting? [N]***

  Esta opción permite establecer la contabilidad de procesos y saber quién ejecutó cierto proceso y cuando, aunque parece útil esto puede consumir muchos recursos del sistema y crear bitácoras muy grandes en poco tiempo.

MISCELLANEOUS DAEMONS

- ***Would you like to deactivate NFS and Samba? [Y]***

  El Network File System *(*Sistema de archivos de red*)*, o NFS, es un protocolo utilizado para sistemas de archivos distribuidos. Posibilita que distintos sistemas conectados a una misma red accedan archivos remotos como si se tratara de locales. Samba es una implementación libre del protocolo de archivos compartidos de Windows, antiguamente llamado SMB, fue renombrado recientemente para sistemas de tipo Unix.

- ***Would you like to deactivate the HP OfficeJet (hpoj) script on this machine? [Y]***

  Este script inicia el sistema de soporte en Linux para los dispositivos HP all in one.

- ***Would you like to deactivate ISDN script on this machine? [Y]***

  ISDN es un método para conectar equipos a Internet, con una velocidad de 128 kbps ha sido sustituido por IDSL otro tipo de conexión más rápida.

APACHE

- ***Would you like to deactivate the Apache web server? [N]***

  Anote la justificación a la respuesta anterior:

  _____
  _____

  Se puede habilitar nuevamente con el comando /sbin/chkconfig httpd on

- ***Would you like to bind the Web server to listen only to the localhost? [N]***

  Esto es muy útil cuando se hace desarrollo web, ya que permite editar un sitio web localmente antes de que se cargue en otro servidor.

- ***Would you like to bind the Web server to a particular interface? [N]***

  En caso de que se permita el acceso al servidor web únicamente a la red interna se tiene que contestar [Y] e ingresar la ip del equipo que se quiere asociar y el puerto que debe escuchar.

- ***Would you like to deactivate the following of symbolic links? [Y]***

  En general trataremos de limitar la información del servidor web que puede ser vista por los usuarios. Al desactivar la opción de seguimiento de ligas simbólicas se previene el que un visitante web pueda leer o modificar archivos que no se encuentran en el directorio de publicación web.

TMPDIR

- ***Would you like to install TMPDIR/TMP scripts? [N]***

  Esto nos permitiría crear directorios temporales alternativos al directorio /tmp con la ventaja de ejecutar scripts cuando los usuarios accedieran a dichos directorios.

  Activar esta opción hará que Bastille instale unos scripts en las cuentas de los usuarios que configuren las variables TMPDIR y TMP de tal manera que utilicen directorios de ficheros temporales completamente individuales, en vez de que todos usen el */tmp* lo que puede ser extremadamente peligroso en entornos multiusuarios.

PRINTING

- ***Would you like to disable printing? [Y]***

  Anote la justificación a la respuesta anterior:

Esto se puede revertir con los siguientes comandos:

/bin/chmod 06555 /usr/bin/lpr /usr/bin/lprm

/sbin/chkconfig lpd on

- ***Would you like to disable CUPS'legacy LPD support? [N]***

  Para efectos de la práctica contestaremos [N], pero si el equipo realizará funciones de impresión, se puede deshabilitar el soporte para el protocolo LPD del sistema de impresión común de Unix.

FTP

FIREWALL

- ***Would you like to run the packet filtering script? [N]***

  Esta opción activaría el firewall nativo de Linux, en este caso no se activará porque se recomienda que el firewall se configure por separado y con otras herramientas específicas para la configuración del mismo.

- ***Are you finished answering the questions, i.e. may we make the changes? [Y]***

  Respondemos afirmativamente para que se apliquen los cambios en el sistema.

  Guardar la configuración y después aplicarla al sistema.

Se ha terminado de configurar Bastille Linux, además de aplicarse los cambios en el sistema operativo.

Ahora con una cuenta de usuario sin privilegios probaremos que la configuración anterior funciona correctamente, en este caso probaremos ciertos puntos en el sistema operativo, que deben estar protegidos por la configuración hecha con Bastille Linux:

Los comandos mount, ping y at no deben tener activo el bit SUID

ls -l /bin/mount
-rwr-xr-x 1 root root 52012 abr 6 2007 /bin/mount

ls -l /bin/ping
-rwr-xr-x 1 root root 36140 abr 6 2007 /bin/ping

ls -l /usr/bin/at
-rwr-xr-x 1 root root 45380 mar 27 2007 /usr/bin/at

Otros programas que inseguros como rcp, rlogin, rsh también deben tener desactivado el bit SUID.

ls –l /usr/bin/rlogin
-rwr-xr-x 1 root root 14388 abr 11 2007 /usr/bin/rlogin
ls –l /usr/bin/rsh
-rwr-xr-x 1 root root 8940 abr 11 2007 /usr/bin/rsh
ls –l /usr/bin/rcp
-rwr-xr-x 1 root root 18640 abr 11 2007 /usr/bin/rcp
ls –l /usr/bin/rexec
-rwsr-xr-x 1 root root 14300 abr 11 2007 /usr/bin/rexec

La máscara de usuario (umask), función que establece los permisos por defecto para los nuevos archivos y directorios creados debe tener el nuevo valor (077)
umask
0077

El número de procesos por usuario debe estar restringido a 150, y eso se comprueba con el comando ulimit.

ulimit -a
max user processes (-u) 150

Por lo que el sistema se encuentra protegido si un ataque de saturación de buffer o *denegación de servicios* intenta hacer una replicación infinita de procesos.

El reforzamiento de la seguridad con Bastille Linux se debe verificar en otros aspectos que en este caso no fueron analizados, para ello es necesario saber qué archivos se van a modificar o qué comandos se ejecutarán mediante la configuración hecha con este programa, pero al revisar estos aspectos básicos podemos darnos una idea de las ventajas de utilizar un software de este tipo.

5.- CUESTIONARIO

1.- ¿Cuáles son los cuatro pasos que se pueden hacer con Bastille para asegurar el sistema operativo?

_____
_____
_____

2.- ¿Qué son las r-tools y qué método de autenticación utilizan?

_____
_____
_____

3.- ¿Qué problema de seguridad presenta el servicio Telnet?

_____
_____
_____

4.- ¿Por qué es importante establecer un límite de procesos que puede ejecutar un usuario?

_____
_____
_____

5.- ¿Cuál es la ventaja de utilizar Bastille para asegurar el sistema operativo?

_____
_____
_____

6.- CUESTIONARIO PREVIO

1.- Investigue ¿Qué medidas para reforzar un sistema operativo tipo Linux existen?

_____
_____
_____

2.- ¿Qué función tiene el comando “umask”?

¿qué permisos se deben establecer en un sistema para que se considere seguro?

_____
_____
_____

3.- ¿Qué servicios remotos se consideran inseguros en un sistema tipo Linux y por qué?

_____
_____
_____

4.- ¿Qué es la denegación de servicios?

_____
_____
_____

5.- Investigue ¿Qué herramientas para el reforzamiento del sistema operativo conoce y cuáles son sus características?

_____
_____
_____

GLOSARIO

Apache: Servidor web HTTP de distribución libre y de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, implementa la noción de sitio virtual y tiene una penetración de más del 50% del total de servidores en el mundo.

ARP: El protocolo de resolución de direcciones es responsable de convertir las direcciones de protocolos de alto nivel (direcciones IP) a direcciones de red físicas.

ATM: ATM es una tecnología se usa para crear redes de alta capacidad y respuesta para permitir el trafico de grandes cantidades de información, este estándar de redes permite la transmisión de cualquier tipo de información como video, voz ,datos y cualquier tipo de información que pueda viajar dentro de una red.

Denegación de servicios (DoS, denial of service): Un ataque por denegación de servicios es aquel que causa que algún recurso o servicio esté demasiado ocupado y por lo tanto sea inaccesible para usuarios legítimos.

DES (Data Encryption Standard, estándar de cifrado de datos): es un algoritmo de cifrado en bloque simétrico, de longitud fija, desarrollado originalmente por IBM y posteriormente modificado y adoptado por el gobierno de EE.UU. en 1977 como estándar de cifrado de todas las informaciones sensibles no clasificadas.

DHCP: es el protocolo de configuración de host dinámico, se trata de un protocolo que permite que un equipo conectado a una red pueda obtener su configuración de red en forma dinámica, es decir, sin intervención particular. Sólo tiene que especificarle al equipo, mediante DHCP, que encuentre una dirección IP de manera independiente. El objetivo principal es simplificar la administración de la red.

DMZ (DMZ, DeMilitarized Zone): Una DMZ se define como una red local que se ubica entre la red interna de una organización y una red externa como Internet. Su objetivo es el de permitir conexiones desde la red interna hacia la red externa, pero limitar las conexiones desde el exterior hacia el interior.

Eavesdropping: La interceptación o eavesdropping, también conocida por "escucha secreta" (passive wiretapping) es un proceso mediante el cual un agente capta información, en claro o cifrada, que no le iba dirigida.

Exploit: (del inglés to exploit, explotar o aprovechar) es una pieza de software, un fragmento de datos, o una secuencia de comandos con el fin de automatizar el

aprovechamiento de un error, fallo o vulnerabilidad, a fin de causar un comportamiento no deseado o imprevisto en los programas informáticos, hardware, o componente electrónico (por lo general computarizado).

FDDI: (Fiber Distributed Data Interface) es un conjunto de estándares ISO y ANSI para la transmisión de datos en redes de computadoras de área extendida o local (LAN) mediante cable de fibra óptica.

Firewall (cortafuegos): Un firewall es un dispositivo que controla las comunicaciones, permitiendo o denegando las transmisiones de una red a la otra. En una arquitectura de red, generalmente se sitúa entre una red local e Internet, como dispositivo de seguridad para evitar que los intrusos puedan acceder a información confidencial.

Frame Relay: es una técnica de comunicación mediante retransmisión de tramas para redes de circuito virtual. Consiste en una forma simplificada de tecnología de conmutación de paquetes que transmite una variedad de tamaños de tramas o marcos ("frames") para datos, perfecto para la transmisión de grandes cantidades de datos.

Hardening: conjunto de actividades que son llevadas a cabo por el administrador de un sistema operativo para reforzar al máximo posible la seguridad de su equipo.

ICMP: El protocolo de mensajes de control de Internet permite administrar información relacionada con errores de los equipos en red. ICMP notifica los errores a los protocolos de capas cercanas, por lo tanto, este protocolo es usado por todos los routers para indicar un error (llamado un problema de entrega).

IGMP: El protocolo de red IGMP (Internet Group Management Protocol) se utiliza para intercambiar información acerca del estado de pertenencia entre enrutadores IP que admiten la multidifusión y miembros de grupos de multidifusión. Los hosts miembros individuales informan acerca de la pertenencia de hosts al grupo de multidifusión y los enrutadores de multidifusión sondean periódicamente el estado de la pertenencia.

IP: El protocolo IP es parte de la capa de Internet del conjunto de protocolos TCP/IP. Es uno de los protocolos de Internet más importantes ya que permite el desarrollo y transporte de datagramas de IP (paquetes de datos), aunque sin garantizar su entrega.

IP spoofing: La suplantación de IP consiste básicamente en sustituir la dirección IP origen de un paquete TCP/IP por otra dirección IP a la cual se desea suplantar.

Iptables: es un sistema de firewall vinculado al kernel de linux que permite no solamente filtrar paquetes, sino también realizar traducción de direcciones de red (NAT) para IPv4 o mantener registros de log.

MAC address spoofing: El atacante modifica la dirección IP o la dirección MAC de origen de los paquetes de información que envía a la red, falsificando su identificación para hacerse pasar por otro usuario. De esta manera, el atacante puede asumir la identificación de un usuario válido de la red, obteniendo sus privilegios.

MD5: MD5 es uno de los algoritmos de reducción criptográficos diseñados por el profesor Ronald Rivest. Procesa mensajes de una longitud arbitraria en bloques de 512 bits generando un compendio de 128 bits. Debido a la capacidad de procesamiento actual esos 128 bits son insuficientes, además de que una serie de ataques criptoanalíticos han puesto de manifiesto algunas vulnerabilidades del algoritmo.

NASL: Las siglas NASL responden a Nessus Attack Scripting Language, es un lenguage script especialmente pensado para Nessus, y cuyo objetivo es poder lanzar funcionalidades del escáner a través de programas externos que se definan para tales efectos.

OpenSSH: OpenSSH (Open Secure Shell) es un conjunto de aplicaciones que permiten realizar comunicaciones cifradas a través de una red, usando el protocolo SSH.

OpenSSL: Consiste en un robusto paquete de herramientas de administración y librerías relacionadas con la criptografía, que suministran funciones criptográficas a otros paquetes como OpenSSH y navegadores web (para acceso seguro a sitios HTTPS).

OSI (Open Systems Interconnection): El modelo de referencia de Interconexión de Sistemas Abiertos es el modelo de red descriptivo creado por la Organización Internacional para la Estandarización lanzado en 1984. Es decir, es un marco de referencia para la definición de arquitecturas de interconexión de sistemas de comunicaciones.

Perl: perl es un lenguaje de programación diseñado por Larry Wall en 1987. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y, en un grado inferior, de muchos otros lenguajes de programación.

Posix: Portable Operating System Interface, se trata de una familia de estándares de llamadas al sistema operativo definidos por el IEEE y especificados formalmente en el IEEE 1003. Persiguen generalizar las interfaces de los sistemas operativos para que una misma aplicación pueda ejecutarse en distintas plataformas.

Protocolo: Conjunto de reglas usadas por computadoras para comunicarse unas con otras a través de una red. Se trata de una convención o estándar que controla o permite la conexión, comunicación, y transferencia de datos entre dos puntos finales.

Puerta trasera (backdoor): Es una secuencia especial dentro del código de programación mediante la cual se puede evitar los sistemas de seguridad del algoritmo (autentificación) para acceder al sistema.

Secure Shell: SSH es el intérprete de órdenes seguro, se trata del nombre de un protocolo y del programa que lo implementa, y sirve para acceder de manera más segura a máquinas remotas a través de una red.

SHA-256: Secure Hash Algorithm es un sistema de funciones hash criptográficas relacionadas de la Agencia de Seguridad Nacional de los Estados Unidos. SHA-256 toma como entrada un mensaje de longitud máxima 2^{64} bits (más de dos mil millones de Gigabytes) y produce como salida un resumen de 256 bits.

SHA-512: Secure Hash Algorithm que produce como salida un resumen de 512 bits.

Sistema de Detección de Intrusos (SDI): Un IDS (Intrusion Detection System) o Sistema de Detección de Intrusiones es una herramienta de seguridad que intenta detectar o monitorizar los eventos ocurridos en un determinado sistema informático o red informática en busca de intentos de comprometer la seguridad de dicho sistema.

Sniffer de red: Un sniffer es un programa de para monitorear y analizar el trafico en una red de computadoras, detectando los cuellos de botellas y problemas que existan en ella.

Software libre: es la denominación del software que respeta la libertad de los usuarios sobre su producto adquirido y, por tanto, una vez obtenido puede ser usado, copiado, estudiado, cambiado y redistribuido libremente.

TCP: (*Protocolo* de Control de Transmisión) es uno de los principales *protocolos* de la capa de transporte del modelo TCP/IP. En el nivel de aplicación, posibilita la administración de datos que vienen del nivel más bajo del modelo, o van hacia él, (es decir, el *protocolo* IP). *TCP* es un protocolo orientado a conexión, es decir, que permite que dos máquinas que están comunicadas controlen el estado de la transmisión.

Token ring: Token Ring es una arquitectura de red desarrollada por IBM en los años 1970 con topología lógica en anillo y técnica de acceso de paso de testigo. Token Ring se recoge en el estándar IEEE 802.5. En desuso por la popularización de Ethernet; actualmente no es empleada en diseños de redes.

UDP: El *protocolo* UDP (*Protocolo* de datagrama de usuario) es un *protocolo* no orientado a conexión de la capa de transporte del modelo TCP/IP. Este *protocolo* es muy simple ya que no proporciona detección de errores (no es un *protocolo* orientado a conexión).

REFERENCIAS BIBLIOGRÁFICAS

[2] RAYA, José Luis, RAYA Cristina; *TCP/IP para Windows 2000 Server*; RA-MA Editorial; 2001, Madrid, España.; p. 1, 12-13.

[3] MCLEAN, Ian; *La biblia TCP/IP*; Ediciones Anaya Multimedia (Grupo Anaya, S.A.), 2001, Madrid, España.; p. 79, 80, 233.

[4] PÉREZ AGUDÍN, Justo et al., *La biblia del hacker, edición 2006*; Ediciones Anaya Multimedia (Grupo Anaya, S.A.), 2006, Madrid, España.; p. 89-91, 680, 954.

[6] LÓPEZ B., M. JAQUELINA, et Al.; *Fundamentos de seguridad informática;* México, UNAM, Facultad de Ingeniería; 2006; p. 28,91

[8] BAUER, Michael D.; *Seguridad en servidores Linux, Técnicas avanzadas para blindar su servidor Linux*; Ediciones Anaya Multimedia (Grupo Anaya, S.A.); 2005; Madrid, España; p. 34, 58.

[14] BASTA, ALFRED, et Al; *Computer security and penetration testing*; Thompson course technology; 2008, USA; p.69.

[19] SANDERS CHRIS; *Practical packet analysis, using Wireshark to solve real-world network problems*; No starch press; 2007; San Francisco USA; p.15.

SHEMA, Mike et al., *Superutililidades hacker*; 1ª. Edición; Ediciones Anaya Multimedia (Grupo Anaya, S.A.); 2007, Madrid, España.

HOWLETT, Tony; *Software libre, herramientas de seguridad*; 1ª Edición; Ediciones Anaya Multimedia, (Grupo Anaya, S.A.); 2005; Madrid, España.

NEMETH, Evi et al.; *La biblia de administración de sistemas Linux, edición 2008*; Ediciones Anaya Multimedia (Grupo Anaya, S.A.); 2007; Madrid, España.

SITIOS CONSULTADOS

[1] http://www.gnu.org/philosophy/free-sw.es.html, sitio del proyecto gnu, 21/09/2010.

[5] http://www.commoncriteriaportal.org, portal del estándar de los criterios comunes para evaluación de seguridad de tecnología de la información, 3/10/2010.

[7] www.nsa.gov, portal de la agencia de seguridad nacional de Estados Unidos de América, 29/09/2010.

[9] http://www.openssh.com, sitio web de openssh, herramienta desarrollada por el proyecto OpenBSD para la conectividad por medio de secure shell, 07/09/2010.

[10] http://bastille-linux.sourceforge.net/, sitio del proyecto bastille linux, programa para el reforzamiento del sistema operativo, 07/09/2010.

[11] http://www.openwall.com/john/, sitio del proyecto openwall, proyecto que tiene una serie de software dedicado a la seguridad, como John the Ripper, 07/09/2010.

[12] http://www.nessus.org/nessus/, sitio web del escáner de vulnerabilidades Nessus para diversos sistemas operativos, 05/09/2010.

[13] http://cirt.net/nikto2, sitio del scaner de vulnerabilidades web Nikto, 04/09/2010

[15] http://www.wireshark.org, sitio del analizador de protocolos en red Wireshark, 06/10/2010.

[16] http://www.turtlefirewall.com/, sitio del proyecto Turtle Firewall, software que permite implementar un cortafuegos de manera sencilla, 06/09/2010.

[17] http://www.snort.org/, sitio web del sniffer de paquetes y detector de intrusos Snort, basado en red, 08/09/2010.

[18] http://base.secureideas.net/, sitio del proyecto BASE, Basic Analysis and Security Engine, aplicación web hecha en PHP que permite analizar los logs y las alertas generadas por Snort, 07/09/2010.