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.