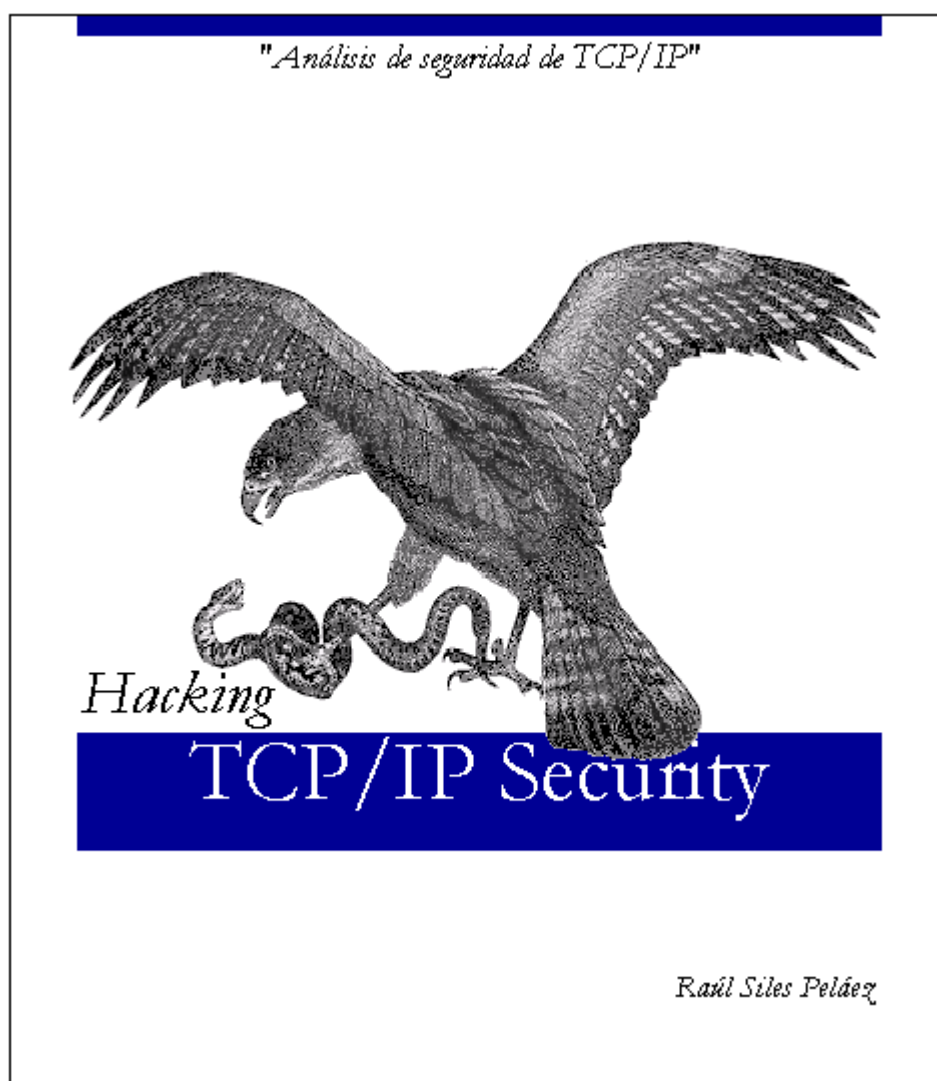


Análisis de seguridad de la familia de protocolos TCP/IP y sus servicios asociados

Edición I



Raúl Siles Peláez

Junio 2002

Análisis de seguridad de la familia de protocolos TCP/IP y sus servicios asociados

*Para Mónica, por estos nueve inmejorables años de
vida y por el nuevo, cercano y deseado futuro.*

*Para mi familia, por estar siempre ahí generando una
“tranquilidad” ideal.*

ii **Boo !!** ;-)

“Análisis de seguridad de la familia de protocolos TCP/IP y sus servicios asociados”

Primera Edición: Junio 2002

Autor: Raúl Siles Peláez

Correo electrónico: raul_siles@hp.com

NOTA: La portada ;-), similar a las empleadas por la conocida editorial O'Reilly & Associates, Inc., ha sido generada en la página Web: <http://www.ilbbs.com/oracovers/>

LICENCIA

Copyright (C) 2002 Raúl Siles Peláez.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, with the Front-Cover Texts being "Análisis de seguridad de la familia de protocolos TCP/IP y sus servicios asociados", and with no Back-Cover Texts.

A copy of the license is included in the section entitled ``GNU Free Documentation License''.

El presente documento se distribuye bajo la licencia conocida como "GNU Free Documentation License":
<http://www.gnu.org/copyleft/fdl.html>

*Análisis de seguridad de la familia de protocolos TCP/IP
y sus servicios asociados*

ÍNDICE

LICENCIA.....	4
NOTAS DEL AUTOR.....	8
1 INTRODUCCIÓN	9
2 TCP/IP	11
3 IP VERSIÓN 6.....	21
4 HISTORIA DE LAS VULNERABILIDADES.....	23
5 VULNERABILIDADES GENÉRICAS.....	25
5.1 FOOTPRINTING	26
5.2 FINGERPRINTING	30
5.3 ESCANEOS DE PUERTOS-VULNERABILIDADES.....	32
5.4 ESCANEOS BASADOS EN EL PROTOCOLO ICMP	35
5.5 SNIFFING	39
5.6 EAVESDROPPING	40
5.7 SNOOPING	40
5.8 IP SPOOFING.....	40
5.9 SMTP SPOOFING Y SPAMMING	41
5.10 DOS: DENIAL OF SERVICE	41
5.11 NET FLOOD.....	43
5.12 SMURF	43
5.13 TCP SYN FLOOD.....	43
5.14 CONNECTION FLOOD	44
5.15 SMTP FLOOD.....	45
5.16 DDoS.....	45
5.17 TRINOO.....	46
5.18 TRIBE FLOOD NETWORK Y TFN2K.....	46
5.19 STACHELDRAHT.....	46

5.20	<i>PING OF DEATH</i>	47
5.21	<i>LOKI</i>	47
5.22	<i>LAND</i>	48
5.23	<i>ROUTING PROTOCOLS</i>	48
5.24	<i>SESSION HIJACKING</i>	49
5.25	<i>SOURCE ROUTING</i>	50
5.26	<i>ICMP REDIRECTS</i>	51
5.27	<i>DIRECTED BROADCAST</i>	51
5.28	<i>SNMP</i>	51
5.29	<i>TCP INITIAL SEQUENCE NUMBERS</i>	52
5.30	<i>TINY FRAGMENT ATTACK</i>	53
5.31	<i>WINNUKE</i>	53
5.32	<i>TEARDROP</i>	54
5.33	<i>DNS</i>	54
5.34	<i>NTP</i>	55
5.35	<i>CABALLOS DE TROYA O TROJANOS</i>	55
5.36	<i>IPSEC</i>	56
5.37	<i>FINGER BOMB</i>	56
5.38	<i>RPC</i>	56
5.39	<i>RELACIONES DE CONFIANZA ENTRE SISTEMAS</i>	56
5.40	<i>BUFFER-OVERFLOWS</i>	57
5.41	<i>FORMAT STRINGS</i>	60
5.42	<i>COMUNICACIONES INALÁMBRICAS: WIRELESS</i>	61
6	PROTECCIONES Y HERRAMIENTAS	63
6.1	<i>FOOTPRINTING</i>	64
6.2	<i>FINGERPRINTING</i>	64
6.3	<i>ESCANEO DE PUERTOS-VULNERABILIDADES</i>	65
6.4	<i>ESCANEO BASADO EN EL PROTOCOLO ICMP</i>	66
6.5	<i>SNIFFING, EAVESDROPPING Y SNOOPING</i>	66
6.6	<i>ENCRIPCIÓN: SSL, PGP, S/MIME</i>	67
6.7	<i>IDS: INTRUSION DETECTION SYSTEMS</i>	69
6.8	<i>IP SPOOFING</i>	71
6.9	<i>SMTP SPAMMING</i>	73
6.10	<i>DoS Y DDoS</i>	74
6.11	<i>NET FLOOD</i>	76
6.12	<i>SMURF</i>	76
6.13	<i>TCP SYN FLOOD</i>	78
6.14	<i>CONNECTION FLOOD</i>	80
6.15	<i>BASTION HOSTS</i>	81
6.16	<i>BASTION ROUTERS</i>	83
6.17	<i>TRINOO, TRIBE FLOOD NETWORK, TFN2K, STACHELDRAHT</i>	84
6.18	<i>NAT: NETWORK ADDRESS TRASLATION</i>	84
6.19	<i>SCREENING ROUTERS</i>	84

6.20	<i>PING OF DEATH</i>	85
6.21	<i>FIREWALLS</i>	85
6.22	<i>LAND</i>	88
6.23	<i>ROUTING PROTOCOLS</i>	88
6.24	<i>SESSION HIJAKING</i>	89
6.25	<i>SOURCE ROUTING</i>	89
6.26	<i>ICMP REDIRECTS</i>	90
6.27	<i>DIRECTED BROADCAST</i>	91
6.28	<i>SNMP</i>	91
6.29	<i>TCP INITIAL SEQUENCE NUMBERS</i>	94
6.30	<i>TINY FRAGMENT ATTACK</i>	95
6.31	<i>WINNUKE</i>	95
6.32	<i>TEARDROP</i>	95
6.33	<i>DNS</i>	95
6.34	<i>NTP</i>	97
6.35	<i>CABALLOS DE TROYA O TROJANOS</i>	97
6.36	<i>IPSEC</i>	97
6.37	<i>FINGER BOMB</i>	99
6.38	<i>TCP WRAPPERS</i>	99
6.39	<i>MPLS</i>	99
6.40	<i>SSH: SECURE SHELL</i>	100
6.41	<i>PROGRAMACIÓN SEGURA: BUFFER OVERFLOWS, FORMAT STRINGS</i>	100
6.42	<i>PKI: PUBLIC KEY INFRASTRUCTURE</i>	101
6.43	<i>TABLA ARP</i>	101
6.44	<i>PARCHES SOFTWARE DE SEGURIDAD</i>	101
6.45	<i>COMPUTER EMERGENCY RESPONSE TEAM: CERT</i>	102
7	EJEMPLOS DE VULNERABILIDADES Y PROTECCIONES PARTICULARES.....	104
7.1	<i>DISPOSITIVOS CISCO: PUERTO DE IDENTIFICACIÓN 1999</i>	104
7.2	<i>DOŚ EN LOS PUERTOS DE ACCESO DE LOS ROUTERS</i>	105
7.3	<i>NETBIOS</i>	105
7.4	<i>ATAQUE DNS CACHE</i>	106
7.5	<i>VULNERABILIDADES DOŚ EN LAS PILAS TCP/IP</i>	107
8	FUTURO	108
9	CONCLUSIONES	110
10	BIBLIOGRAFÍA Y URLS.....	112
11	ANEXO I: WEBS DE SEGURIDAD	126
11.1	<i>PORTALES Y REPOSITORIOS DE SEGURIDAD (EN ORDEN ALFABÉTICO)</i>	126
11.2	<i>PORTALES DE SEGURIDAD DE LOS PRINCIPALES FABRICANTES</i>	129
11.3	<i>REVISTAS DE SEGURIDAD O E-ZINES</i>	130
11.4	<i>GRUPOS DE NOTICIAS</i>	130

11.5	LISTAS DE CORREO	130
11.6	VIRUS.....	131
12	ANEXO II: HERRAMIENTAS DE SEGURIDAD	132
12.1	TABLAS DE HERRAMIENTAS DE SEGURIDAD	132
12.2	REPOSITORIOS CON HERRAMIENTAS Y UTILIDADES DE SEGURIDAD	134
12.3	REPOSITORIOS DE <i>EXPLOITS</i> (<i>EXPLOITS ARCHIVES</i>).....	135
12.4	OTROS ENLACES.....	135
13	ANEXO III: LECTURAS ADICIONALES: LIBROS Y ARTÍCULOS	138
14	GNU FREE DOCUMENTATION LICENSE.....	138

NOTA: Para facilitar la búsqueda de información, se ha hecho corresponder el número de apartado de una vulnerabilidad determinada (apartado 4.X) con la protección o herramienta asociada (apartado 5.X).

NOTAS DEL AUTOR

El presente trabajo pretende únicamente ser una introducción al mundo de la seguridad de los sistemas de información desde la perspectiva y el enfoque que proporciona una de las familias de protocolos de comunicaciones más extendidas actualmente: TCP/IP.

La cantidad de información existente hoy en día respecto al interesante mundo de la **seguridad de la información** es infinita, actualmente podría medirse en yottas ;-). Para comprobarlo sólo es necesario realizar una búsqueda en Internet de alguno de los términos o conceptos relacionados con la seguridad y comprobar el número de referencias existentes. Debido a ésto, se recomienda al lector ampliar la información de su interés, sobre cualquiera de las técnicas y tecnologías mencionadas a través de los buscadores. Esta situación se ratifica igualmente mediante los anexos incluidos en este trabajo, dónde podrás encontrar un gran número de referencias en Internet relacionadas con la seguridad.

El autor del presente documento declina cualquier responsabilidad asociada al uso incorrecto y/o malicioso que pudiera realizarse con la información expuesta en el mismo. Por tanto, no se hace responsable de las consecuencias que puedan derivarse de la información contenida en él, incluyendo los posibles errores e información incorrecta existentes, así como de las consecuencias que se puedan derivar sobre su aplicación en sistemas de información reales.

Por último, sí como lector, tienes alguna duda, comentario, sugerencia, crítica o cualquier tipo de aportación constructiva, por favor, no dudes en contactar conmigo a través del correo electrónico.

1 INTRODUCCIÓN

Este trabajo pretende analizar los aspectos asociados a la seguridad de la familia de protocolos TCP/IP, así como de los servicios que se establecen típicamente sobre esta pila de protocolos.

Hoy en día, el uso de TCP/IP se ha extendido prácticamente a la totalidad de las redes de comunicaciones de datos, potenciado fundamentalmente por la expansión de Internet, así como de las redes corporativas y de cooperación asociadas a esta tecnología: *Intranets* y *Extranets*. En julio de 1999 se contabilizaban 56 millones de ordenadores conectados a Internet [I-1], [I-2], [I-3].

Habitualmente el diseño de las redes se basaba en características como la funcionalidad o la eficiencia, pero no en la seguridad; condiciones que son rentables desde un punto de vista de negocio a corto plazo, pero que pueden resultar caras a largo. Para la realización del análisis y diseño de una red segura es necesario conocer los detalles y características de los protocolos de comunicaciones subyacentes, que serán los encargados de transportar la información y datos que desean distribuirse. A su vez, deberán analizarse los servicios que se proporcionan en dicha red y sus detalles de funcionamiento.

En el presente trabajo se ha optado por un enfoque eminentemente técnico, comenzando por mostrar las características básicas del protocolo TCP/IP, para posteriormente profundizar en las vulnerabilidades existentes en los protocolos desde un punto de vista teórico. Este estudio se complementa con una visión práctica proporcionada por las herramientas existentes para permitir o denegar la ejecución de las mencionadas vulnerabilidades.

A lo largo del texto se presentan los medios existentes en ciertos sistemas operativos, tanto de sistemas como de dispositivos de red, para afrontar el análisis y protección de la seguridad de la implementación de la pila TCP/IP y de los servicios implementados sobre los protocolos superiores. Concretamente se hace mención a HP-UX (versión 11.00), Solaris (versión 8), Linux (versiones de *kernel* 2.2.x y 2.4.x), Windows NT/2000, Cisco IOS.

La información existente hoy en día, tanto en la bibliografía tradicional (libros, revistas, artículos técnicos...) como en la propia Internet (servidores Web, FTP, listas de correo, grupos de *news*...) respecto al tema genérico abordado en este documento, “La Seguridad de las Redes”, así como de las vulnerabilidades y protecciones asociadas, es realmente extensa, por lo que el presente trabajo pretende actuar como punto de partida para todo aquél que esté interesado en indagar en este apasionante mundo, pudiendo con el mismo adquirir unas nociones generales de la seguridad en las redes TCP/IP y sus servicios, y disponiendo de una amplia fuente de referencias por las que continuar sus andanzas en busca de más información.

Debe tenerse en cuenta que algunos de los temas tratados, como por ejemplo los *firewalls*, la encriptación, los sistemas IDS, las VPNs – IPsec, *Trojanos* avanzados o las PKIs no pueden ser analizados en profundidad, ya que la mera introducción detallada de los conceptos que estas

tecnologías incluyen requeriría una documentación abundante, y por si solos podrían constituir un trabajo de extensión similar al presente.

Un método realmente útil a la hora de obtener más información sobre seguridad es emplear un buscador de Internet, preferiblemente www.google.com. Mediante la referencia de los diferentes nombres de las vulnerabilidades o sistemas de protección mencionados a lo largo del trabajo, se podrá disponer de un gran número de enlaces Web con descripciones de su significado, así como de los *exploits* y programas que permiten su obtención y ejecución.

Analizando la seguridad de las redes, no está de más reflexionar sobre el comentario expuesto por Bruce Schneier en el *foreword* de la segunda edición de *Hacking Exposed* [L-1]: “sí una red de ordenadores tiene una vulnerabilidad de seguridad, pero nadie la conoce, ¿es insegura?”; concluyendo qué, “Una red de ordenadores con una vulnerabilidad de seguridad es insegura para aquellos que la conocen. Sí nadie la conoce – porque sea una vulnerabilidad que no ha sido descubierta – entonces la red es segura. Si una persona la conoce, entonces la red es insegura para él pero segura para cualquier otro. Sí el fabricante del equipamiento de red la conoce, sí los grupos de investigación de seguridad la conocen, sí la comunidad *hacker* la conoce... – la inseguridad de la red aumenta a medida que las noticias sobre la vulnerabilidad salen a la luz”.

Extrapolando el análisis previo, se pueden obtener las 3 leyes imperantes en la seguridad informática actualmente [L-15]:

1. Todo software tiene *bugs*.
2. Todo software de seguridad tiene *bugs* de seguridad.
3. Si el software no es utilizado, no se sabrá que *bugs* tiene realmente. La filosofía de seguridad de los *firewalls* se basa en ésta regla: “*security through obscurity*”.

2 TCP/IP

La familia de protocolos TCP/IP (*Transport Control Protocol / Internet Protocol*) caracteriza un estándar *ad-hoc* de protocolos de comunicaciones entre sistemas informáticos. El protocolo TCP/IP surgió alrededor de 1960 como base de un sistema de comunicación basado en redes de conmutación de paquetes desarrollado por el gobierno estadounidense y la agencia de defensa, ARPA. Actualmente constituye la infraestructura tecnológica más extendida y desarrollada sobre la que circulan las comunicaciones electrónicas (datos, voz, multimedia...). Su expansión se ha debido principalmente al desarrollo exponencial de la red mundial, Internet.

La mejor fuente bibliográfica asociada a TCP/IP se encuentra disponible en una trilogía publicada por *Richard Stevens*: [L-4] (Web: [TC-3]), [L-5] y [L-6], pese a que las fuentes de información acerca de TCP/IP existentes hoy en día son innumerables [TC-1] [TC-2].

Dentro de los equipos que poseen una implementación de la pila de protocolos TCP/IP, se distinguen de forma más detallada dos grupos, todos ellos objetivo de los potenciales ataques:

- Sistemas: son los equipos que engloban tanto a los clientes de un servicio o comunicación, ya sean PCs de sobremesa o estaciones de trabajo (que ejecutarán un sistema operativo cliente: Windows, Unix, MacOS...) así como dispositivos móviles (PDAs, teléfonos móviles...), como a los servidores que proporcionan el servicio, típicamente ejecutando un sistema operativo servidor: Unix (incluyendo todas sus variantes: HP-UX, Linux, Solaris, AIX...), AS/400, Windows NT/2000, Novell Netware.
Principalmente serán estos últimos el objetivo de los *hackers*, al contener información relevante.
- Dispositivos de red: son los encargados de que el tráfico de red fluya dentro o entre redes. Por tanto engloban a los repetidores, puentes o *bridges*, concentradores o *hubs*, conmutadores o *switches*, encaminadores o *routers*, cortafuegos o *firewalls*, servidores de terminales y acceso (RAS) (que contienen un conjunto de módems o accesos RDSI), dispositivos de almacenamiento (*storage appliance*)... Los principales fabricantes son Cisco, 3Com, Lucent, Nortel, HP...

Desde el punto de vista de la seguridad, la familia de protocolos TCP/IP puede ser vulnerada en base a dos conceptos inherentes a su diseño:

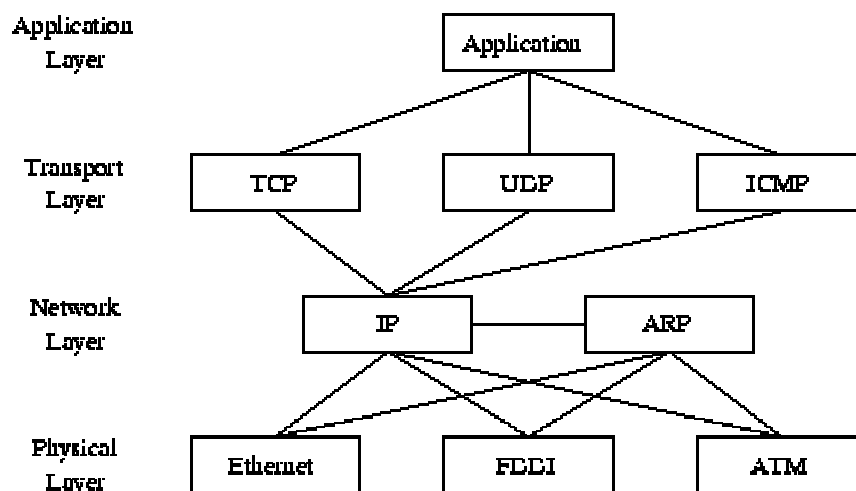
1.- El formato de los paquetes de los diferentes protocolos:

Aparte de la propia información transportada, la información contenida en cada uno de los campos de las cabeceras de los protocolos proporciona una fuente muy valiosa de conocimiento.

2.- El modo de funcionamiento de los protocolos:

Las etapas asociadas a cada proceso en los protocolos, así como el método de actuación en las diferentes situaciones posibles, ofrecen la información necesaria para analizar la existencia de vulnerabilidades.

TCP/IP está diseñado en una estructura en capas, fundamentada en el estándar de los protocolos de comunicaciones que diseñó la organización ISO, denominado OSI, *Open Systems Interconnection*. Cada una de las capas es responsable de llevar a cabo una tarea específica de la comunicación. Concretamente, TCP/IP dispone de cuatro capas:



La capa física (*Physical Layer*) asociada al medio de comunicación físico y de enlace, capas 1 y 2 de OSI, incluye los *drivers* que controlan las tarjetas de red y toda la gestión de la conexión entre el hardware y los cables y dispositivos de red. Los protocolos asociados a este nivel no pertenecen propiamente a TCP/IP pero son la base sobre el que éste se desarrolla.

La capa de red (*Network Layer*, o capa 3 de OSI) se encarga del envío y recepción de los paquetes a través de la red, así como de encaminarlos por las diferentes rutas que deben recorrer para llegar a su destino. Principalmente el protocolo IP, junto a ICMP (aunque como éste se encapsula en IP no siempre es considerado en este nivel – ver figura), se encargan de estas tareas.

La capa de transporte (*Transport Layer*, o capa 4 de OSI) se encarga de manejar los flujos de datos entre equipos. Existen dos protocolos principalmente a este nivel: TCP, un protocolo fiable y orientado a conexión, y UDP, un protocolo más simple pero que no garantiza la recepción de los datos, además, no orientado a conexión.

Por último la capa de aplicación (*Application Layer*, que en OSI correspondería a las capas 5, 6 y 7) gestiona las características de las comunicaciones propias de la aplicación. En TCP/IP en este nivel se encuentran numerosos protocolos, como Telnet, FTP, HTTP, SMTP, SNMP, NFS, NNTP...

Una vez introducidas las bases en las que se fundamenta el diseño de la familia de protocolos TCP/IP, se pasa a la descripción, por un lado, de los paquetes que conforman los protocolos básicos mencionados (principalmente la cabecera de cada uno de ellos), y por otro, se mencionarán los aspectos más relevantes del funcionamiento del protocolo más complejo de la familia, TCP.

Para cada protocolo se muestra el RFC original que dio lugar a sus especificaciones. Actualmente los protocolos han ido evolucionando por lo que el número de RFC actual es mayor.

- Protocolo ARP: RFC 826

El protocolo ARP, *Address Resolution Protocol*, permite realizar ciertas tareas cuyo objetivo es asociar un dispositivo IP, que a nivel lógico está identificado por una dirección IP, a un dispositivo de red, que a nivel físico posee una dirección física de red. Este protocolo se utiliza típicamente en entornos de redes de área local, *ethernet*, que es el entorno más extendido en la actualidad. Existe un protocolo, RARP, cuya función es la inversa. Pese a que este protocolo puede asociarse a otros protocolos de nivel 3 distintos de IP, se describen los campos en base a éste.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Hardware Protocol																Network Protocol															
Hardware Address Length								Network Address Length								Operation															
Sender Hardware Address																Sender Network Address															
Sender Network Address																															
Target Hardware Address																															
Target Network Address																															

Descripción de los campos ARP:

- Hardware Protocol (16 bits): tecnología de red empleada por debajo de TCP/IP, p.ej. *ethernet*.
- Network Protocol (16 bits): tipo de protocolo empleado a nivel 3, p.ej. IP.
- Hardware Address Length (8 bits): longitud de la dirección de red hardware.
- Network Address Length (8 bits): longitud de la dirección de red IP.
- Operation (16 bits): tipo de operación, para saber si se trata de una petición ARP o una respuesta. También es válido para RARP.

- Sender Hardware Address (48 bits): dirección *ethernet* (*hardware*) del sistema emisor.
- Sender Network Address (32 bits): dirección IP (de red) del sistema emisor.
- Target Hardware Address (48 bits): dirección *ethernet* del sistema receptor o destino.
- Target Network Address (32 bits): dirección IP del sistema receptor o destino.

- Protocolo IP: RFC 791

IP, *Internet Protocol*, es el protocolo principal de TCP/IP, encargado de la transmisión y enrutamiento de los paquetes de datos al equipo destino. Es un protocolo no fiable, es decir, que no asegura la recepción final en el equipo destinatario de la información. Para el control de los posibles errores dispone de un protocolo de aviso, ICMP. La fiabilidad de la comunicación debe proporcionarla los protocolos superiores, como TCP.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Version				IHL				Type of Service								Total Length															
Identification																Flags				Fragment Offset											
Time to Live								Protocol								Header Checksum															
Source Address																															
Destination Address																															
Options																								Padding							
Data																															

Descripción de los campos IP:

- Version (4 bits): el campo de versión especifica el formato de la cabecera IP. Actualmente solo se manejan dos, el IP estándar o versión 4 y la siguiente generación o versión 6 (IPv6 o IPng).
- Internet Header Length (IHL) (4 bits): longitud de la cabecera Internet o IP medida en palabras de 32 bits. La longitud mínima es de 5 palabras (cabecera sin opciones IP).
- Type of Service (8 bits): el TOS se emplea para especificar parámetros como la fiabilidad, la precedencia, el retardo y la capacidad de salida que deberían asociarse a este paquete.
- Total Length (16 bits): longitud total del datagrama IP en *bytes*, incluyendo la cabecera IP y los datos encapsulados por ésta.
- Identification (16 bits): este campo identifica de forma unívoca cada paquete enviado por un sistema emisor. Es empleado para la reconstrucción de paquetes grandes que debieron fragmentarse en algún punto de la red.
- Flags (3 bits):

- El primer *bit*, *flag More Flag (MF)*, se emplea en la fragmentación, para determinar si este paquete es el último fragmento de un paquete inicial, o si aún siguen más fragmentos.
 - El siguiente *bit*, *Don't Fragment (DF)*, permite especificar si el emisor desea que se fragmente el datagrama.
 - El último *bit* o *flag* actualmente no es utilizado.
 - Fragment Offset (13 bits): mediante el *offset* o desplazamiento es posible determinar la posición que ocupaba en el paquete original este fragmento, de forma que el destino pueda reconstruirlo consecuentemente.
 - Time to Live (8 bits): el TTL indica el tiempo máximo que el paquete puede permanecer en la red. Si su valor es cero, el paquete es destruido. Típicamente, en lugar de indicar un valor temporal se refiere al número de *routers*, o saltos, por los que puede pasar.
 - Protocol (8 bits): éste especifica el protocolo de siguiente nivel empleado, que recibirá los datos en el otro extremo. Es decir, el contenido del campo de datos comenzará por la cabecera del siguiente protocolo, por ejemplo TCP o UDP.
 - Header Checksum (16 bits): es un campo de comprobación de la integridad (*checksum*) sólo de la cabecera IP. Debido a que ciertos campos de la cabecera cambian en el transcurso del paquete por la red, por ejemplo el TTL en cada salto, éste campo debe recalcularse y verificarse en cada punto intermedio dónde la cabecera es procesada.
 - Source address (32 bits): dirección IP del dispositivo fuente o emisor.
 - Destination address (32 bits): dirección IP del dispositivo destino o receptor.
 - Options (variable): la longitud de este campo es variable, pudiendo tener cero o más opciones. Concretamente contempla las opciones solicitadas por el emisor, que pueden ser de seguridad, *source routing*, *timestamps*...
 - Padding (variable): asegura que la cabecera IP acaba en un múltiplo de 32 *bits*.
 - Data (variable): este campo contiene los datos a enviar, siendo su longitud múltiplo de 8 *bits*. El valor máximo de la longitud es 65.535 *bytes* (64 *Kbytes*). El campo comenzará con el contenido de la cabecera del protocolo de siguiente nivel: TCP o UDP.
-
- Protocolo TCP: RCF 793

El protocolo TCP, *Transport Control Protocol*, es el empleado en la mayoría de los servicios que componen Internet actualmente. Es un protocolo fiable, es decir, se asegura de que los paquetes de datos llegan al otro extremo mediante el uso de números de secuencia y de confirmaciones de recepción (*ACKs*), y es orientado a conexión, es preciso que las dos partes que van a comunicarse conozcan a la otra y establezcan una conexión formal. Asimismo, está debería terminarse de forma adecuada. Por último se trata de un servicio de *stream*, ya que las partes intercambian flujos de datos de 8 *bits* (1 *byte*), por lo que no existen marcadores en los datos, sólo información.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Source Port																Destination Port															
Sequence Number																															
Acknowledgement Number																															
Data Offset				Reserved						Flags						Window															
Checksum																Urgent Pointer															
Options																								Padding							
Data																															

Descripción de los campos TCP:

- Source Port (16 bits): número de puerto o servicio (aplicación) en el sistema origen.
- Destination Port (16 bits): número de puerto o servicio (aplicación) en el sistema destino.
- Sequence number (32 bits): especifica el número de secuencia del primer *byte* de datos de este segmento TCP. Cuando el *flag* SYN está activo (establecimiento de conexión), éste se considera parte de los datos, y se emplea un número inicial de secuencia, ISN (*Initial Sequence Number*), como valor inicial: por tanto el valor del campo suele ser ISN+1.
- Acknowledgement number (32 bits): Si el *flag* ACK está activo, este campo contiene el valor del siguiente número de secuencia que el sistema espera recibir. Este campo siempre contendrá un valor una vez que la conexión se haya establecido.
- Data offset (4 bits): número de palabras de 32 bits en la cabecera TCP. Indica dónde comienza exactamente el campo de datos y, por tanto, finaliza la cabecera.
- Reserved (6 bits): 6 *bits* reservados para uso futuro.
- Flags (6 bits): los siguientes *flags* determinan el funcionamiento de la conexión TCP:
 - URG: flag de urgencia. Permite que el emisor especifique al receptor la existencia de información urgente en el flujo de datos.
 - ACK: se trata de un paquete en el que se reconoce la recepción de datos. Especifica que el campo *acknowledgement number* es válido.
 - PSH: función *push*. Indica que el receptor debe pasar los datos contenidos en el paquete a la aplicación tan pronto como sea posible.
 - RST: produce un *reset* de la conexión.
 - SYN: paquete de establecimiento de conexión. Sincroniza los números de secuencia.
 - FIN: indica que el sistema no tiene más datos que enviar.
- Window (16 bits): la ventana TCP es empleada para el control de flujo de los datos. Contiene el número de *bytes* de datos que pueden ser recibidos empezando por el indicado en el campo *acknowledgement*.
- Checksum (16 bits): es un campo de comprobación de la integridad (*checksum*) de la cabecera TCP, los datos y la *pseudo* cabecera. Un valor de cero es ilegal, al ser un campo obligatorio.
- Urgent Pointer (16 bits): el objetivo de este puntero es indicar el desplazamiento en el campo de datos dónde se encuentran los datos urgentes, para que el receptor pueda adquirirlos directamente. Tiene validez sólo cuando el *flag* URG está activo.

- Options (variable): la longitud de este campo es variable, pudiendo tener cero o más opciones. La opción más común en TCP es MSS, *Maximum Segment Size*, y permite al sistema especificar el tamaño máximo de segmento que espera recibir. Suele indicarse en el paquete de establecimiento de conexión (SYN).
- Padding (variable): la cabecera TCP debe tener una longitud múltiplo de 32 *bits*. Este campo permite asegurarse de eso: se rellena el segmento TCP con ceros.

La *pseudo* cabecera, también aplicada a UDP, empleada en el cálculo del *checksum*, se basa en la inclusión en el algoritmo de *checksum* de 12 *bytes* correspondientes no a TCP o UDP, sino a la cabecera IP del paquete que los contiene, concretamente las direcciones IP fuente y destino, el tipo de protocolo y la longitud del datagrama.

- Protocolo UDP: RFC 768

UDP, *User Datagram Protocol*, es un protocolo muy sencillo, no orientado a conexión y no fiable, por lo que son los protocolos de nivel superior los que deben asegurarse de la recepción de los datos. Cada operación de envío genera un único datagrama UDP. Es empleado principalmente en aplicaciones multimedia, para el envío de flujos de información sin un coste de conexión asociado.

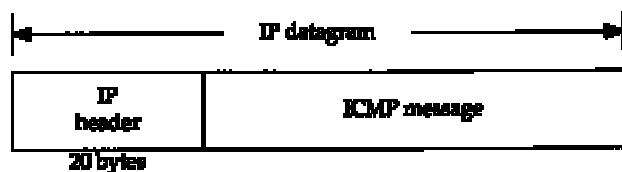
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Source Port																Destination Port															
Length																Checksum															
Data																															

Descripción de los campos UDP:

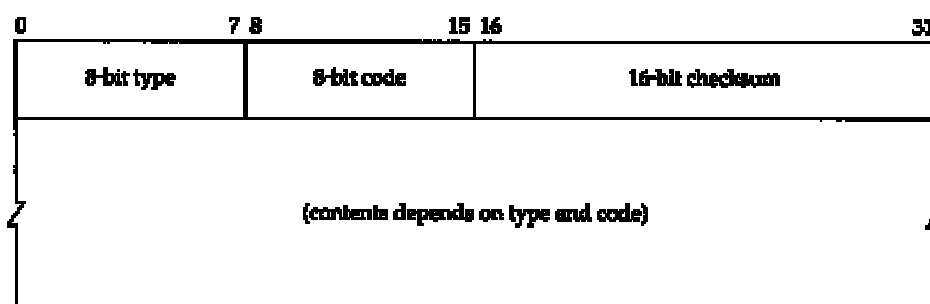
- Source Port (16 bits): número de puerto o servicio (aplicación) en el sistema origen.
- Destination port (16 bits): número de puerto o servicio (aplicación) en el sistema destino.
- Length (16 bits): longitud en *bytes* del datagrama, incluyendo la cabecera UDP y los datos.
- Checksum (16 bits): es un campo de comprobación de la integridad (*checksum*) de la cabecera UDP, la *pseudo* cabecera y los datos. Si el valor es cero indica que no debe realizarse ningún cálculo relativo al *checksum* (se trata de un valor opcional).

- Protocolo ICMP: RFC 792

ICMP, *Internet Control Messaging Protocol*, es considerado a nivel de IP, ya que es empleado por éste para notificar mensajes de error o situaciones que requieren cierta atención. Debido a que los paquetes ICMP viajan en paquetes IP es a veces considerado un nivel por encima.



Existen numerosos tipos de mensajes que permiten tanto notificar situaciones de error, como realizar peticiones de información. Asimismo, un mensaje ICMP siempre contiene los 8 primeros *bytes* del paquete IP que dio lugar a su generación, así el sistema receptor al extraerlo de la red sabrá a qué módulo asociárselo: TCP, UDP.



Descripción de los campos ICMP:

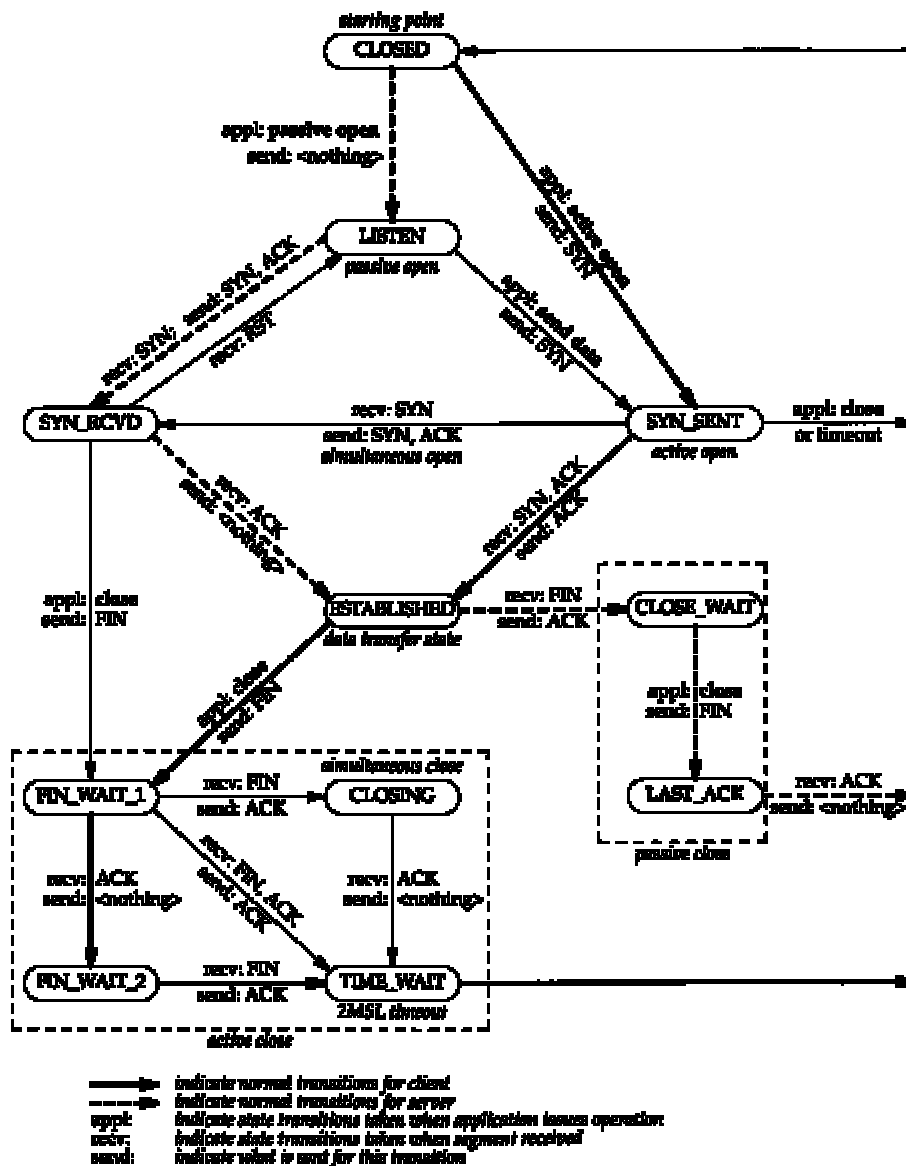
- Type (8 bits): identifica el tipo particular de mensaje ICMP. Existen 15 valores diferentes.
- Code (8 bits): dentro de los mensajes del mismo tipo, el código especifica diferentes condiciones.
- Checksum (16 bits): es un campo de comprobación de la integridad (*checksum*) de la totalidad del mensaje ICMP. Se trata de un campo obligatorio.
- Contents (variable): el contenido de este campo depende del tipo de mensaje ICMP.

Analizando los mensajes ICMP de error, los más habituales son el tipo 3 que permite especificar que no se puede llegar a una red destino, *destination unreachable*, que a su vez dispone de múltiples códigos, aplicados a la red, al *host*, al servicio...

Asimismo, ICMP proporciona la funcionalidad de adquirir información mediante pares de paquetes petición / respuesta, por ejemplo, para adquirir la máscara de red de un sistema o preguntar por la hora, mediante el manejo de *timestamps*. Por último, en numerosas ocasiones se emplea para comprobar la existencia de conectividad, como en la utilidad *ping*, empleando paquetes ICMP *echo* y *echo reply*.

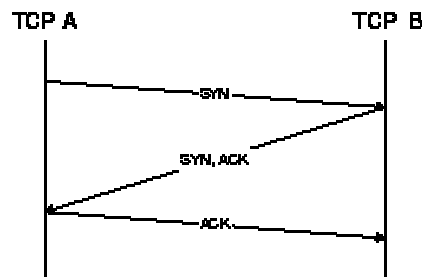
Dentro de los protocolos descritos existen diferentes funcionalidades que no se describen en el presente texto, como la fragmentación IP, el control de flujo de TCP mediante el algoritmo de ventana deslizante, el método de actuación frente a la recepción de los diferentes paquetes ICMP..., por lo que se recomienda al lector el recurrir a las referencias mencionadas para obtener su descripción.

Debido a que, de los mencionados, el protocolo más complejo es TCP, ya que se encarga de controlar el estado en todo momento de la comunicación mediante el concepto de conexión, se procede a introducir algún detalle particular de su funcionamiento. El proceso determinista que debe llevarse a cabo dentro de una conexión viene condicionado por el **diagrama de estados de TCP**. Éste especifica las reglas que debe seguir toda implementación de TCP durante la transmisión de información, y denota los estados posibles de una conexión en cada momento, así como los paquetes que deben recibirse o enviarse para transitar de un estado a otro.



Estudiando los procedimientos posibles en el transcurso de una comunicación TCP, los principales son el establecimiento y terminación de la misma.

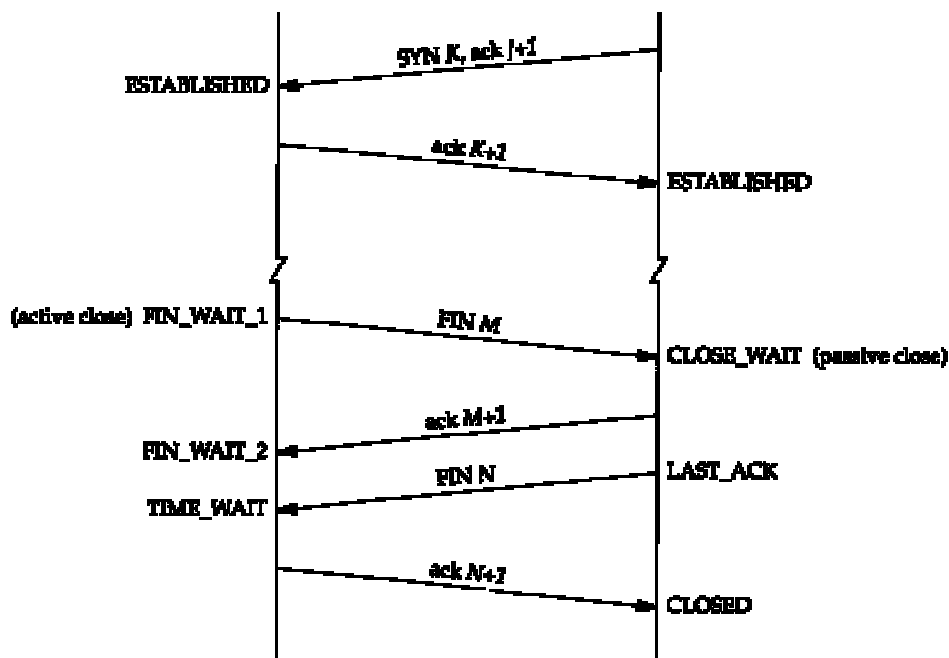
- Establecimiento de una conexión TCP:



Una conexión TCP se establece en tres pasos, en lo que se denomina el *three-way handshake*, en el que el sistema que inicia la conexión o cliente (*TCP A*), envía un paquete de SYN (con su número de secuencia inicial asociado a esta conexión) al destinatario o servidor (*TCP B*); éste le responde con un paquete SYN-ACK, confirmando la recepción del SYN inicial y enviándole su propio número de secuencia. Finalmente, el cliente reconoce la recepción del SYN del servidor mediante un ACK. En este momento la conexión se ha establecido y puede tener lugar toda la transferencia de datos.

- Finalización de una conexión TCP:

De igual modo, la finalización de la conexión se lleva a cabo mediante el intercambio de paquetes TCP, en este caso cuatro. El sistema que desea terminar la conexión envía un paquete de FIN notificándolo. El sistema remoto reconoce su recepción (mediante un ACK), e indica su interés de terminar con la conexión también mediante un paquete de FIN, que debe ser reconocido por el otro extremo (de nuevo con un ACK) para que la misma se de por cerrada.



En la figura se muestra también parte del establecimiento inicial (2 primeros paquetes).

3 IP VERSIÓN 6

El rápido crecimiento de Internet, así como la existencia de numerosos dispositivos que requieren comunicarse mediante TCP/IP: servidores, PCs, dispositivos de red, PDAs, teléfonos móviles y otros dispositivos inalámbricos, teléfonos IP, equipos de electrónica: TV, vídeos, DVDs..., electrodomésticos..., está empujando al cambio de IP versión 4 a IP versión 6 o *IP Next Generation* [NG-4] [NG-5]. La posibilidad de poder escalar el número de direcciones IP, que escasea actualmente, es su propiedad fundamental.

En lugar de los 32 bits de las direcciones en IPv4, se pasará a 128 bits en IPv6, pudiendo por tanto disponer teóricamente de $3,4 \times 10^{38}$ nodos únicos, unos 340 billones de billones de billones.

El inmenso dominio de direcciones existentes no es su única característica; su propiedad de auto-configuración y auto-descubrimiento sustituirá al actual DHCP. Las direcciones se asignarán en base a un identificador de interfaz, como por ejemplo la dirección MAC, y un prefijo de red. Asimismo, IPv6 introduce una arquitectura P2P, que permite esquemas de direccionamiento globales extremo a extremo. Esta tecnología sustituirá al NAT. A su vez, incluye características para asegurar la calidad de servicio (QoS).

La definición e implementación del nuevo protocolo se está llevando a cabo por el IETF junto a los fabricantes de equipos de red. Asimismo, el IETF posee grupos dedicados a la transición de la Internet actual, basada en la versión 4, a la futura, basada en la versión 6, como el llamado “NGtrans WG”.

Actualmente existe una red basada en IPv6 sobre la Internet actual, empleada para la realización de pruebas y mejoras del protocolo: **6Bone**. Esta red se basa principalmente en la creación de túneles que encapsulan el tráfico IPv6 sobre las redes IPv4. Por otro lado, y con el objetivo de independizar ambos protocolos, los grandes operadores de telecomunicación están creando redes troncales de transporte de IPv6, sobre enlaces serie (Frame-Relay, ATM o DWDM): www.v6start.net.

El otro protocolo que permitirá una correcta transición basada en la independencia de las dos versiones es MPLS, empleado ampliamente hoy en día en las redes principales de los proveedores de comunicaciones.

Desde el punto de vista de los sistemas, una alternativa es la conocida como *dual-stack*, en el que un equipo posea dos pilas TCP/IP simultáneamente, una para cada versión, pudiendo comunicarse con ambos tipos de redes y equipos.

4 HISTORIA DE LAS VULNERABILIDADES

En los primeros años, los ataques involucraban poca sofisticación técnica. Los ataques internos se basaban en utilizar los permisos para alterar la información. Los externos se basaban en acceder a la red simplemente averiguando una clave válida. A través de los años se han desarrollado formas cada vez más sofisticadas de ataque para explotar vulnerabilidades en el diseño, configuración y operación de los sistemas. Esto permitió a los nuevos atacantes tomar control de sistemas completos, produciendo verdaderos desastres que en muchos casos llevaron a la desaparición de aquellas organizaciones o empresas con altísimo grado de dependencia tecnológica (bancos, servicios automáticos, etc).

Estos nuevos métodos de ataque han sido automatizados, por lo que en muchos casos sólo se necesita un conocimiento técnico básico para realizarlos. El aprendiz de intruso, *script-kiddie* o *ankle biter*, o aprendiz de *hacker*, *lamer* o *wannabee*, tiene acceso hoy en día a numerosos programas y *scripts* (*exploits*) que se aprovechan de las vulnerabilidades, disponibles desde numerosas fuentes *underground*, como *hacker newsgroups*, *mailing-lists* y *web sites*, donde además encuentra todas las instrucciones para ejecutar ataques con las herramientas disponibles.

Bruce Schneier [HV-1], en numerosos artículos, ha definido y clasificado las generaciones de ataques en la red existentes a lo largo del tiempo [HV-2]:

- La primera generación: ataque físico
Ataques que se centraban en los componentes electrónicos: ordenadores y cables. El objetivo de los protocolos distribuidos y de la redundancia es la tolerancia frente a un punto único de fallo. Son mayormente problemas para los que actualmente se conoce la solución.
- La segunda generación: ataque sintáctico (*objeto del presente trabajo*)
Las pasadas décadas se han caracterizado por ataques contra la lógica operativa de los ordenadores y las redes, es decir, pretenden explotar las vulnerabilidades de los programas, de los algoritmos de cifrado y de los protocolos, así como permitir la denegación del servicio prestado. En este caso se conoce el problema, y se está trabajando en encontrar soluciones cada vez más eficaces.
- La tercera generación: ataque semántico [HV-3]
Se basan en la manera en que los humanos asocian significado a un contenido. El hecho es que en la sociedad actual la gente tiende a creerse todo lo que lee (medios informativos, libros, la Web...). El inicio de este tipo de ataques surgió con la colocación de información falsa en boletines informativos o *e-mails*, por ejemplo, para beneficiarse de las inversiones dentro de la bolsa financiera. También pueden llevarse a cabo modificando información caduca.
Esta generación de ataques se lleva a su extremo si se modifica el contenido de los datos de los programas de ordenador, que son incapaces de cotejar o sospechar de su veracidad, como por ejemplo la manipulación del sistema de control de tráfico aéreo, el control de un coche inteligente, la base de datos de los libros más vendidos o de índices bursátiles como el NASDAQ. Lo más curioso es que estos ataques han existido fuera del entorno informático

desde hace muchos años como estadísticas manipuladas, falsos rumores..., pero es la tecnología la que potencia su difusión.

Su solución pasará no sólo por el análisis matemático y técnico, sino también por el humano.

La conclusión tras el análisis de las vulnerabilidades desde un punto de vista operacional es que para evitarlas pueden definirse las tareas a realizar dentro de un sistema de seguridad en tres etapas:

1. Prevención: implementada por dispositivos como los *firewalls*.
2. Detección: a través de sistemas como los IDS.
3. Respuesta: las acciones a tomar deben ser dirigidas por la parte humana, típicamente los administradores de la red.

5 VULNERABILIDADES GENÉRICAS

Las vulnerabilidades pretenden describir las debilidades y los métodos más comunes que se utilizan para perpetrar ataques a la seguridad de la familia de protocolos TCP/IP (confidencialidad, integridad y disponibilidad de la información).

Los ataques pueden estar motivados por diversos objetivos, incluyendo fraude, extorsión, robo de información confidencial, venganza, acceso no autorizado a un sistema, anulación de un servicio o simplemente el desafío de penetrar un sistema.

Éstos pueden provenir principalmente de dos fuentes:

- Usuarios autenticados, al menos a parte de la red, como por ejemplo empleados internos o colaboradores externos con acceso a sistemas dentro de la red de la empresa. También denominados *insiders*.
- Atacantes externos a la ubicación física de la organización, accediendo remotamente. También denominados *outsiders*.

Los métodos de ataque descritos se han dividido en categorías que pueden estar relacionadas entre sí, ya que el uso de un método permite o facilita el uso de otros, en ocasiones, complementarios. Un ejemplo de ataque podría ser la realización del análisis de un sistema, mediante *fingerprinting*, tras el cual es posible explotar una vulnerabilidad como un *buffer-overflow* de un servicio TCP/IP, enviando paquetes que parecen válidos mediante *IP spoofing*. Dentro de los métodos no se han incluido ataques de alto nivel, como por ejemplo la distribución y ejecución de virus a través del correo electrónico (protocolo SMTP), ya que afectan a vulnerabilidades particulares de las aplicaciones y los lenguajes de programación soportados por éstas.

En numerosas ocasiones se ha empleado inicialmente el término inglés para nombrar la vulnerabilidad, ya que es como se conoce comúnmente, para posteriormente asociarle su posible traducción al español.

Las vulnerabilidades pueden clasificarse según dos criterios:

- Número de paquetes a emplear en el ataque:
 - *Atomic*: se requiere un único paquete para llevarla a cabo.
 - *Composite*: son necesarios múltiples paquetes.
- Información necesaria para llevar a cabo el ataque:
 - *Context*: se requiere únicamente información de la cabecera del protocolo.
 - *Content*: es necesario también el campo de datos o *payload*.

Context	<i>Ping of death</i> <i>Land attack</i> <i>WinNuke</i>	<i>Port scan</i> <i>SYN Flood</i> <i>TCP hijacking</i>
	<i>DNS attack</i> <i>Proxied RPC</i> <i>IIS attack</i>	<i>SMTP attacks</i> <i>String matches</i> <i>Sniffing</i>
Content	Atomic	Composite

5.1 Footprinting

La regla número uno antes de planificar o analizar un posible ataque a un sistema, o red, es conocer el objetivo, es decir, obtener su huella identificativa o *footprinting* – el arte de extraer toda la información posible de la red objetivo del ataque. Por tanto la primera tarea a realizar pasa por dedicar un esfuerzo considerable a obtener y recolectar ésta información.

Existen numerosas utilidades para obtener la información de un sistema: ping, whois, finger, rusers, nslookup, rcpinfo, telnet, dig, nmap...

El atacante podría comenzar por ejecutar un ping contra el sistema a atacar:

```
$ ping www.sistema.es
```

para comprobar su existencia (también el uso de `tracert` o de `nslookup` permiten obtener información si los paquetes ICMP están deshabilitados). Posteriormente podría intentar extraer información del sistema y sus usuarios mediante la utilidad `finger`:

```
$ finger root@www.sistema.es
```

pudiendo realizar diversas pruebas tratando de descubrir algún usuario válido, para posteriormente intentar adivinar su clave y disponer de acceso remoto al sistema mediante `telnet`.

La primera etapa sería buscar información general de la empresa en Internet. Los pasos que podrían realizarse a continuación irían desde realizar búsquedas en ICANN para obtener el rango de redes asociado a la organización, como descubrir que sistemas activos existen (mediante *pings*), para posteriormente conocer los posibles servicios vulnerables, empleando para ello la técnica de escaneo de puertos. Finalmente, a través de una conexión, por ejemplo mediante *telnet*, al puerto seleccionado se podrá obtener información en la mayoría de los casos, como cadenas de texto (*banner grabbing*), que identifiquen el servicio, y que permitirán conocer el tipo de servidor y su versión. Esta técnica puede aplicarse mediante el uso de utilidades como *telnet* o *netcat* [FN-22].

Ejemplos de cadenas identificativas:

```
$ ftp 10.10.10.10
Connected to 10.10.10.10.
220 hostname FTP server (Version 1.1.214.2 Mon May 11 12:21:14 GMT 1998) ready.
User (10.10.10.10:(none)) :

$ telnet 10.10.10.10
```

```
HP-UX hostname B.11.00 A 9000/712 (t0)
login:
```

La información a obtener de un objetivo se puede clasificar en 4 grupos principales [L-1]: Internet, Intranet, Extranet y Acceso Remoto. A continuación se analizan diversos métodos empleados:

A través de una búsqueda en los grupos de noticias (USENET), un atacante puede obtener información emitida por los usuarios de una organización, y así conocer detalles de los sistemas existentes en la misma, de las tecnologías empleadas y de la relevancia de la seguridad por parte de los administradores, con el objetivo de obtener el perfil de la organización. Para ello basta con buscar por la cadena “@dominio.com” en www.dejanews.com. Asimismo, es recomendable buscar información relacionada con la organización en meta buscadores o en los grandes buscadores de Internet [FN-2].

El análisis tanto de las páginas Web como de los fuentes correspondientes a éstas páginas HTML pueden proporcionar información interesante, principalmente las etiquetas de comentarios:<, ! o --. Para realizarlo se podrán emplear herramientas que permite realizar una copia completa de un servidor, como *wget*, *Teleport* o *Web Snake*.

La información relativa a los dominios asociados a una organización así como sus subredes correspondientes puede obtenerse en los servicios WHOIS [WI-1] (ver protecciones DoS y *NetFlood*).

Otro de los servicios que proporciona información muy útil es el servicio de nombres o DNS. Si el servicio no se ha configurado adecuadamente, será posible realizar una consulta de transferencia de zona completa, lo que permitirá obtener toda la información de traducción de direcciones IP a nombres de máquinas. Este tipo de consulta puede realizarse con la utilidad *nslookup*:

```
$ nslookup
Default Server:  dns.dominio.com
Address:         30.1.1.1
> server 100.100.100.1
Default Server:  [100.100.100.1]
Address:         100.100.100.1
> set type=any
> ls -d dominio_objetivo.com. > /tmp/fichero_zona
```

La información de la zona puede permitir obtener relaciones entre sistemas, el propósito para el que se emplean los mismos, el sistema operativo que ejecutan o el tipo de sistema que ocultan las la nomenclatura: todo ello en base al nombre o a ciertos registros propios del servicio DNS.

Por otro lado, el DNS permite conocer los servidores de correo asociados a una organización a través de los registros MX. Para obtenerlos basta con recurrir de nuevo a la utilidad *nslookup*:

```
$ nslookup
Default Server:  dns.dominio.com
Address:         30.1.1.1
> server 100.100.100.1
Default Server:  [100.100.100.1]
Address:         100.100.100.1
> set type=mx
> dominio_objetivo.com.
```

Por ejemplo, los servidores encargados de la recepción del correo del dominio *yahoo.com* son:

```
> yahoo.com
Name Server:  dnserver.dominio.com
Address:  100.100.100.1

Trying DNS
Non-authoritative answer:
yahoo.com      preference = 1, mail exchanger = mx2.mail.yahoo.com
yahoo.com      preference = 1, mail exchanger = mx3.mail.yahoo.com
yahoo.com      preference = 9, mail exchanger = mta-v18.mail.yahoo.com
yahoo.com      preference = 1, mail exchanger = mx1.mail.yahoo.com

Authoritative answers can be found from:
yahoo.com      nameserver = NS3.EUROPE.yahoo.com
yahoo.com      nameserver = NS1.yahoo.com
yahoo.com      nameserver = NS5.DCX.yahoo.com
mx3.mail.yahoo.com  internet address = 216.115.107.17
mx3.mail.yahoo.com  internet address = 216.136.129.16
mx3.mail.yahoo.com  internet address = 216.136.129.17
NS3.EUROPE.yahoo.com  internet address = 217.12.4.71
NS1.yahoo.com        internet address = 204.71.200.33
NS5.DCX.yahoo.com    internet address = 216.32.74.10
>
```

Simplemente añadir que existen numerosas utilidades para automatizar la extracción de información del DNS, como por ejemplo, *host*, *Sam Spade*, *axfr*, *dig*...

Traceroute:

Toda red está caracterizada por una topología o distribución, tanto física como lógica, concreta. Existe una herramienta que ayuda a la obtención de ésta: *traceroute* [FN-4], creada originalmente para solucionar problemas en una red. Esta técnica permite saber todos los sistemas existentes en un camino entre dos equipos.

Su funcionamiento se basa en el manejo del campo TTL de la cabecera IP de un paquete, de forma que es capaz de determinar uno a uno los saltos por los que un determinado paquete avanza en la red. El campo TTL actúa como un contador de saltos, viéndose decrementado en uno al ser reenviado por cada *router*. Por tanto, mediante esta utilidad de diagnóstico se podrá obtener una lista de los elementos de red recorridos desde una ubicación origen hasta un sistema destino. Los paquetes de comprobación son enviados de tres en tres.

El primer datagrama enviado tiene un TTL de valor 1, por lo que generará en el primer salto un paquete *ICMP Time Exceeded*. El siguiente datagrama verá el valor del TTL incrementado en uno, por lo que será capaz de llegar un salto más en la red que el datagrama anterior.

Debe tenerse en cuenta que el comportamiento de la herramienta puede variar en función de la plataforma: el *traceroute* de Unix, por ejemplo Linux o HP-UX, utiliza el protocolo UDP (pudiendo usar ICMP mediante la opción “-I”). En el caso de Windows NT (*tracert*), se emplea el protocolo

ICMP. Por tanto, en función de los filtros existentes en los dispositivos de red que deben atravesarse, será necesario usar uno u otro protocolo. En el caso de UDP, el mensaje generado por el sistema final es *ICMP Port Unreachable*, mientras que en los sistemas intermedios, al igual que en todos los sistemas en el caso de ICMP, es un *ICMP Echo Reply*.

En el caso de la implementación basada en UDP, el número de puerto UDP se incrementa en cada iteración, por lo que es necesario conocer el puerto UDP con el que se debe enviar en base a una fórmula: $(\text{Puerto objetivo} - (\text{n}^\circ. \text{ de saltos} * \text{n}^\circ. \text{ de pruebas})) - 1$

El principal problema de esta versión es que si existe un *firewall*, éste filtrará los paquetes UDP al ir con diferentes puertos. Existe una implementación de Michael Shiffman que evita estos incrementos en los puertos; presenta el problema de que sí en ese puerto UDP hay un servicio disponible, no se generará el paquete ICMP de vuelta.

Ejemplo de *traceroute*:

```
# traceroute 15.13.120.190
traceroute to 15.13.120.190 (15.13.120.190), 30 hops max, 20 byte packets
 1 pop (15.128.104.12)                    5 ms
   15.128.104.1 (15.128.104.11)          6 ms      3 ms
 2 15.191.224.13 (15.191.224.113)        126 ms    141 ms    133 ms
 3 15.191.40.1 (15.191.40.11)            134 ms    152 ms    129 ms
 4 172.16.8.37 (172.16.8.137)            248 ms    273 ms    253 ms
 5 172.17.192.49 (172.17.192.149)        314 ms    324 ms    302 ms
 6 15.73.152.2 (15.73.152.12)            313 ms    304 ms    308 ms
 7 15.68.136.3 (15.68.136.13)            304 ms    325 ms    326 ms
 8 15.61.211.83 (15.61.211.183)          347 ms    369 ms    312 ms
 9 15.75.208.38 (15.75.208.138)          296 ms    273 ms    309 ms
10 15.13.120.187 (15.13.120.190)        305 ms    285 ms    282 ms
#
```

Existen herramientas gráficas con una funcionalidad similar a *traceroute* [FN-5] [FN-6], que permiten visualizar el mapa mundial con las correspondientes asociaciones de cada elemento IP y su ubicación física.

Asimismo, mediante la utilización de paquetes ICMP (ECHO y REPLY, *ping sweep*) se puede obtener la lista de dispositivos IP activos. Existen herramientas que facilitan la obtención de este tipo de información: *fping*, *gping* y *Pinger* [FN-9] [FN-10] [FN-11]. Asimismo, mediante la utilidad “*nmap*” con la opción “-sP” se obtienen resultados similares.

Este protocolo permite obtener también información de otro tipo [FN-14], como la franja horaria del sistema destino o la máscara de subred empleada en los diferentes subinterfaces (paquetes ICMP de tipo 13 y 17 respectivamente).

Por último, en el caso de disponer del demonio *fingerd* (puerto TCP 79), el sistema operativo del *host* puede ser identificado en muchos casos a través de una petición *finger* del tipo “*root@host*, *bin@host* o *daemon@host*”. Este servicio se creó en los comienzos de Internet para obtener información de contacto de los usuarios de un sistema, cuando la seguridad no era un hecho a tener en cuenta. Otros

comandos que permiten información de este servicio son: “finger -l @objetivo.com y finger 0@objetivo.com”. Por ejemplo:

```
# finger root@host.dominio.com
[host.dominio.com]
Login name: root          (messages off)
Directory: /              Shell: /sbin/sh
On since May 23 13:49:41 on pts/ta from sistema.dominio.com
New mail received Thu May 10 16:18:39 2001;
  unread since Wed May 23 13:49:42 2001
No Plan.
#

# finger -l @ host.dominio.com
[host.dominio.com]
Login name: root          (messages off)
Directory: /              Shell: /sbin/sh
On since May 23 13:49:41 on pts/ta from sistema.dominio.com
1 minute 38 seconds Idle Time
New mail received Thu May 10 16:18:39 2001;
  unread since Wed May 23 13:49:42 2001
No Plan.
#
```

Otra técnica empleada sobre los servidores de correo electrónico para obtener información del sistema y la red destino es mediante la ejecución del comando SMTP “expn <user>”.

Existen herramientas integradas cuyo objetivo es aunar las diferentes técnicas presentadas en este apartado inicial, de forma que se obtenga toda la información posible de un entorno de red [FN-21].

5.2 Fingerprinting

Una técnica más específica que permite extraer información de un sistema concreto es el *fingerprinting* [FN-17], es decir, la obtención de su huella identificativa respecto a la pila TCP/IP. El objetivo primordial suele ser obtener el sistema operativo que se ejecuta en la máquina destino de la inspección. Esta información junto con la versión del servicio o servidor facilitará la búsqueda de vulnerabilidades asociadas al mismo.

Gran parte de la información de la pila TCP/IP puede obtenerse en base al intercambio en tres pasos propio del protocolo TCP/IP (*TCP three-way handshake*) [FN-15].

La probabilidad de acierto del sistema operativo remoto es muy elevada, y se basa en la identificación de las características propias de una implementación de la pila TCP/IP frente a otra, ya que la interpretación de los RFCs no concuerda siempre. Para poder aplicar esta técnica con precisión es necesario disponer de un puerto abierto (TCP y/o UDP).

Las diferentes pruebas a realizar para diferenciar los sistemas operativos son:

- *FIN probe*: Al enviarse un paquete de FIN el sistema remoto no debería responder, aunque implementaciones como la de Windows NT devuelven un FIN-ACK.

- *Bogus flag probe*: Se activa un *flag* TCP aleatorio en un paquete SYN. La respuesta de implementaciones como Linux devuelven un SYN-ACK con el mismo *flag* activo.
- *ISN sampling*: Pretende encontrarse un patrón empleado por la implementación para seleccionar los números iniciales de secuencia (ISN) de una conexión TCP.
- Monitorización del “*Don’t fragment bit*”: Se analiza si el sistema operativo establece por defecto el *bit* de no fragmentación (DF) como activo o no.
- Tamaño de ventana TCP inicial: El tamaño de ventan empleado por defecto en cada implementación es muy particular y ayuda a descubrir de cual puede tratarse.
- Valor de ACK: El valor del número de secuencia asignado en el campo ACK diferencia también la implementación, ya que algunas devuelven el valor recibido como número de secuencia mientras que otras lo incrementan en uno.
- Mensaje de error de *ICMP quenching*: El RFC 1812 determina que el control de flujo de mensajes de error debe limitarse. Al enviar un paquete UDP a un número elevado de puerto, aleatoriamente, se puede medir el número de mensajes de tipo *unreachable* por unidad de tiempo.
- *ICMP message quoting*: Los comentarios añadidos a los mensajes de error ICMP varían en función del sistema operativo.
- Mensajes de error ICMP-integridad: Las cabeceras IP pueden ser alteradas por las diferentes implementaciones al devolver mensajes de error ICMP. Un análisis exhaustivo de los cambios en las cabeceras puede permitir determinar el S.O.
- TOS (Tipo de Servicio): Ante los mensajes “*ICMP port unreachable*” puede examinarse el campo TOS, que suele ser cero pero puede variar.
- Gestión de la fragmentación [FN-18]: El manejo de los paquetes fragmentados que se superponen es gestionado de forma particular por cada pila: al reensamblar los fragmentos, algunas sobrescriben los datos más antiguos con los nuevos y viceversa.
- Opciones TCP: Los RFCs 793 y 1323 definen las opciones TCP posibles. Mediante el envío de paquetes con muchas opciones avanzadas activas (*no operation*, *MSS*, *Window scale factor*, *timestamps*...) puede descubrirse el comportamiento de cada S.O.

Dos de las herramientas que facilitan esta tarea son NMAP [PT-1] y QUESO [FN-16]. Mientras que la funcionalidad de la primera es muy amplia, la segunda sólo se aplica a la aplicación de esta técnica (identificación de sistemas a través del comportamiento de la pila TCP/IP). Es posible añadir nuevas identificaciones de sistemas a una base de datos ya existente en <http://www.insecure.org/cgi-bin/nmap-submit.cgi>.

Dentro de las técnicas de identificación de un sistema existen otras, denominadas pasivas, que no se basan en enviar paquetes al sistema a atacar. Para ello monitorizan el tráfico asociado al sistema [FN-19] [FN-20], y en función de los atributos y características de los paquetes, principalmente de las cabeceras TCP, determinan su origen:

- TTL: ¿cuál es el valor del campo *Time To Live* (TTL) en los paquetes salientes?
- Tamaño de ventana: ¿cuál es el valor fijado por el S.O.?
- TOS: ¿se fija algún valor para el campo Tipo de Servicio, TOS?
- DF: ¿se activa o no el *bit* de no fragmentación?

5.3 Escaneo de puertos-vulnerabilidades

Una vez que se dispone de los dispositivos a nivel IP activos en una red (por ejemplo, mediante ICMP), puede aplicarse a cada uno de ellos una técnica, centrada en la posterior búsqueda de vulnerabilidades, basada en una exploración de escaneo de puertos abiertos, tanto UDP como TCP.

El escaneo es la determinación de las características de una red o sistema remotos, con el objetivo de identificar los equipos disponibles y alcanzables desde Internet, así como los servicios que ofrece cada uno. Permite saber los sistemas existentes, los servicios ofrecidos por ellos, cómo están organizados los equipos, que sistemas operativos ejecutan, cual es el propósito de cada uno.

De forma general, entre los métodos de escaneo se incluyen técnicas como:

- *Ping sweep*
- Escaneo de puertos
- *Firewalking*
- *Trace routing*
- Identificación de Sistema Operativo

Al escanear los puertos de los sistemas se descubren puntos de entrada a los mismos, que abren las puertas a nuevas vulnerabilidades potenciales en base a la implementación del servidor que escucha tras cada puerto. Además, esta técnica también permite identificar el tipo de sistema existente, así como su sistema operativo, y las aplicaciones que ofrecen un servicio en la red, así como su versión asociada.

La herramienta por excelencia para realizar un escaneo de puertos es NMAP [PT-1]. Las técnicas existentes en el proceso de escaneo emplean diferentes procedimientos para descubrir la información del servicio:

- *TCP connect scan*: Mediante el establecimiento de una conexión TCP completa (3 pasos).
- *TCP SYN scan*: Se abren conexiones a medias, ya que simplemente se envía el paquete SYN inicial, determinando la existencia de un servicio si se recibe del sistema objetivo un SYN-ACK. Si, por el contrario, se recibe un RST-ACK, es que no existe un servicio. En el caso de la existencia de un servicio, se envía un RST-ACK para no establecer conexión alguna, y no ser registrados por el sistema objetivo, a diferencia del caso anterior.

Estos dos tipos funcionarán en todos los sistemas con implementaciones TCP/IP, mientras que los siguientes variarán según la implementación particular:

- *TCP FIN scan*: Al enviar un FIN a un puerto, RFC 793 [PT-2], debería recibirse como resultado un paquete de *reset* si está cerrado (se aplica principalmente a las pilas TCP/IP de Unix).
- *TCP Xmas Tree scan*: Esta técnica es similar a la anterior, obteniéndose como resultado también un RST si el puerto está cerrado. En este caso se envían paquetes FIN, URG y PUSH.
- *TCP Null Scan*: En el caso de poner a cero todos los *flags* de la cabecera TCP, debería recibirse de nuevo como resultado un paquete RST en los puertos no activos.

- *TCP ACK scan*: Mediante este procedimiento puede determinarse si un *firewall* es simplemente de filtro de paquetes (manteniendo el tráfico de sesiones abiertas, caracterizadas por el *flag ACK*) o si mantiene el estado, con un sistema de filtro de paquetes avanzado.
- *TCP window scan*: Mediante una anomalía en ciertas implementaciones en como se muestra el tamaño de ventana TCP, puede saberse si un puerto está abierto o si es o no filtrado.
- *TCP RPC scan*: Es una técnica propia de sistemas Unix que permite conocer puertos de llamadas a procedimientos remotos (RPCs), junto al programa asociado a los mismos y su versión.
- *UDP scan*: Al enviarse un paquete UDP a un puerto destino, puede obtenerse como resultado un paquete ICMP de puerto inalcanzable (*port unreachable*), con lo que se determina que el puerto no está activo. En caso contrario, no se recibirá ese mensaje.

Debido a que UDP es no orientado a conexión, la fiabilidad de éste método depende de numerosos factores (más aún en Internet), como son la utilización de la red y sus recursos, la carga existente, la existencia de filtros complejos. Asimismo y a diferencia de los escaneos TCP, se trata de un proceso lento, ya que la recepción del mencionado paquete se rige por el vencimiento de temporizadores.

Mediante pruebas UDP es posible determinar si un sistema está o no disponible, así como sus servicios UDP. Para ello se envían datagramas UDP con 0 *bytes* en el campo de datos. En el caso de que el puerto esté cerrado, se recibirá un mensaje *ICMP Port Unreachable*. Si está abierto el puerto, no se recibirá ninguna respuesta. En el caso en el que se detecten un elevado número de puertos UDP abiertos, podrá indicar que existe un dispositivo de filtrado entre el atacante y el objetivo.

Para confirmar esta última posibilidad, se puede enviar un paquete UDP al puerto cero, lo que debería generar una respuesta ICMP como la ya comentada. Si no se recibe ninguna respuesta quiere decir que hay un dispositivo filtrando el tráfico. Este tipo de prueba suele ser detectada por los IDS.

Existen otras comprobaciones más centradas en el nivel de aplicación, como pueden ser la extracción de los servicios RPC existentes a través del *portmapper*, la obtención de listados de sistemas de ficheros compartidos a través de *nfsd*, *Samba* o *NetBios*, el escaneo de vulnerabilidades de CGIs en los servidores Web, así como de versiones conocidas de servicios típicos: *Sendmail*, *IMAP*, *POP3*, *RPC status* y *RPC mountd*.

Algunas herramientas existentes (las más populares y reconocidas a lo largo del tiempo), aparte de NMAP que permiten aplicar algunas de estas técnicas se muestran a continuación. El análisis de todas ellas supondría la elaboración de un extenso estudio, por lo que más información sobre las características y capacidades de cada una puede ser obtenida de las referencias asociadas.

- Nmap (Unix): analiza tanto UDP como TCP, además de poseer numerosas funcionalidades.
<http://www.insecure.org/nmap/>
- nmapNT (Windows) versión de NMAP para Windows.
<http://www.eeye.com/html/Research/Tools/nmapnt.html>

- SuperScan (Windows): escáner de puertos TCP.
<http://members.home.com/rkeir/software.html>
- NetScan Tools Pro 2000 (Windows): incluye utilidades no solo para el escaneo de puertos.
<http://www.nwpsw.com>
- Strobe (Unix): solo permite el análisis de servicios TCP.
<ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/distfiles/strobe-1.06.tgz>
- udp_scan (Unix): añade a la anterior el soporte de UDP (originalmente contenida en SATAN).
- Netcat o nc (Unix y NT): además de otras funcionalidades añadidas es aplicable a UDP y TCP.
<http://www.l0pht.com/~weld/netcat/index.html>
- WinScan (Windows): escáner de puertos TCP, en modo texto y gráfico.
<http://www.prosolve.com>
- WUPS (Windows): escáner de puertos UDP.
<http://ntsecurity.nu>
- ADMHack:
<http://adm.freelsd.net/ADM/>
- Security Analyzer (Windows): NetIQ
<http://www.netiq.com/products/sa/default.asp>

Por ejemplo, mediante nmap pueden realizarse las siguientes acciones de identificación de sistemas:

- Descubrimiento de direcciones IP activas mediante un escaneo de la red:

```
# nmap -sP <<rango_direcciones_IP>>
```

- Escaneo de puertos TCP activos:

```
# nmap -sT <<rango_direcciones_IP>>
```

- Escaneo de puertos UDP activos:

```
# nmap -sU <<rango_direcciones_IP>>
```

- Intento de obtención del sistema operativo de un equipo en red:

```
# nmap -O <<rango_direcciones_IP>>
```

Utilidades de escaneo de vulnerabilidades:

- NESSUS (Unix y Win): Permite el análisis de vulnerabilidades conocidas sobre un sistema.
<http://www.nessus.org>
- SATAN (Security Administrator Tool for Analyzing Networks, Unix):
<http://www.fish.com/satan/>
<http://www.porcupine.org/satan/>
<http://www.trouble.org/~zen/satan/satan.html>
- SAINT (nuevas versiones de SATAN)

<http://www.wwdsi.com/saint/>

- SARA: Security Auditor's Research Assistant
<http://www-arc.com/sara/>
- TITAN: Escaner de vulnerabilidades para Solaris
<http://www.fish.com/titan/>

Este tipo de utilidades permite comprobar si un sistema es vulnerable a un conjunto muy amplio de problemas de seguridad encontrados en el pasado e incluidos en la base de datos de las diferentes aplicaciones, alertándonos sobre su existencia y su posible solución. Las comprobaciones afectan a un gran número de servicios asociados a la pila TCP/IP.

La lista detallada del estándar que define la reserva y uso concreto de cada puerto TCP/IP puede obtenerse de [PT-8]. Algunas utilidades, como SATAN, fueron las precursoras inicialmente de los sistemas de ataque y protección actuales, pero hoy en día no tienen utilidad al ser más lentas e intentar explotar vulnerabilidades reparadas actualmente.

Mediante la ayuda de la utilidad NMAP pueden analizarse todas las técnicas empleadas por la herramienta, que por su innovación y complejidad, representan las principales utilizadas hoy en día.

Una vez que el atacante dispone de la lista de sistemas externos e internos a su alcance, empleará las herramientas mencionadas para analizar su comportamiento y uso.

5.4 Escaneo basado en el protocolo ICMP

Una vez conocido el propósito original del protocolo ICMP, notificar errores y condiciones inusuales que requieren atención respecto del protocolo IP, y el formato de sus paquetes, es necesario analizar los usos indebidos que se le pueden dar, todos asociados al escaneo de un sistema remoto [IC-1] [IC-2].

De manera excepcional, se incluirán en cada una de las técnicas basadas en ICMP los métodos para su detección, en lugar de incluirse en el apartado de protecciones.

- ICMP Echo (*Ping sweep*):

Mediante esta técnica se pretenden identificar los equipos existentes en las redes objetivo de un ataque, típicamente accesibles desde Internet. Constituye uno de los pasos principales en la obtención de información.

Empleando para ello los paquetes ICMP de tipo *echo* (8) y *echo reply* (0), se sabrá si una determinada dirección IP está o no activa. Se envía un paquete de tipo *echo*, y si se recibe el paquete de *echo reply* es que dicha dirección está siendo utilizada.

La técnica envía numerosos paquetes de este estilo para conocer todos los equipos disponibles en una subred. Existen numerosas herramientas que implementan este proceso, como por ejemplo, *fping*, *gping*, el propio *nmap*, o la utilidad *pinger* de Rhino9.

Para detectar este tipo de paquetes enviados de forma masiva puede analizarse el log del servidor DNS asociado al dominio escaneado, ya que aparecerán múltiples intentos de resolución de nombres de direcciones IPs consecutivas. Asimismo, se podrá obtener la dirección IP del atacante.

Los sistemas IDS también permiten su detección, tanto cuando se usa de forma secuencial, como cuando se lanzan los *pings* en paralelo.

- ICMP *broadcast*:

Cuando se envía un paquetes ICMP *echo* a la dirección de *broadcast* o a la dirección de red, con un único paquete enviado se consigue que todos los equipos respondan con su *echo reply* asociado. Las implementaciones de los diferentes sistemas operativos se comportan de manera diferente. Esta técnica puede emplearse en las variantes de Unix, pero los SO de Microsoft, Windows, no responden a este tipo de paquetes.

El RFC1122 especifica que este comportamiento mencionado en último lugar debería ser el correcto: “*if we send an ICMP echo request to an IP Broadcast or IP multicast addresses it may be silently discarded by a host*”.

Existen técnicas de escaneo más avanzadas basadas en ICMP, pero NO en los paquetes de tipo *echo*. Podrían considerarse técnicas tanto de *ICMP sweep* como de *ICMP broadcast*, pero con otros tipos de paquetes ICMP, no *echo*. Estos paquetes se van a analizar a continuación:

- ICMP *Timestamp*:

Mediante el envío de un paquete ICMP de tipo *timestamp* (13), si un sistema está activo, se recibirá un paquete de *timestamp reply* (14) indicando que implementa este tipo de transferencia de información que permite conocer la referencia de tiempo en el sistema destino. Tal y como denota el RFC 1122, la decisión de responder a estos paquetes depende de la implementación. Algunos sistemas Windows sí responden mientras que otros no, sin embargo la mayoría de los Unix sí que lo implementan.

- ICMP *Information*:

El propósito de los paquetes ICMP de información (15) y su respuesta asociada (16), *information reply*, es permitir que ciertos equipos que no poseían disco del que extraer su propia configuración, pudieran autoconfigurarse en el momento de su arranque, principalmente para obtener su dirección IP. En el paquete, tanto la dirección origen como destino tienen el valor cero.

Tanto el RFC 1122 como el 1812 indican que los sistemas no deberían generar ni responder a este tipo de paquetes, pero la realidad de las implementaciones existentes es otra. Algunos sistemas operativos

responderán cuando la dirección IP destino del paquete tiene el valor de una dirección IP específica. En la respuesta, en lugar de tener la dirección IP de la red en el campo de dirección origen, se tiene la dirección IP del *host*.

Algunos UNIX comerciales y equipos Cisco implementan la respuesta ante este tipo de paquetes.

- *ICMP Address Mask:*

El propósito de los paquetes de tipo *address mask* (17) y *address mask reply* (18), era que los equipos o estaciones de trabajo sin disco pudiesen obtener la máscara de red asociada a la subred en la que estaban conectados en el momento de arrancar. Se supone que un sistema no debería responder con un paquete de este tipo salvo que fuera un agente autorizado para notificar la máscara, típicamente el *router* de la subred.

Mediante esta información, un atacante puede conocer la estructura interna de la red y el esquema de enrutamiento empleado. Asimismo, permite identificar los *routers* existentes en el camino que une al atacante con la red objetivo. Un ejemplo de SO que “ayuda” mucho en este aspecto es Sun Solaris y versiones personales de Windows.

Es posible emplear técnicas de detección de equipos más avanzadas, no ya en función del tipo de paquete ICMP, sino en base al comportamiento de las implementaciones del protocolo ICMP. Para ello se analizarán los mensajes de error de ICMP, generados desde las máquinas que sirven como prueba, lo que nos permitirá saber si existe algún dispositivo de filtrado presente, así como descubrir la configuración de las listas de acceso empleadas.

De manera general, los métodos a emplear incluyen:

- Modificación maliciosa de la cabecera IP de un paquete, por ejemplo cambiando el campo de la longitud de la cabecera, o los campos de opciones del protocolo IP.
 - Uso de valores inválidos en los campos de la cabecera IP.
 - Posibilidad de abusar de la fragmentación.
 - Emplear el método de escaneo basado en el protocolo UDP: es el protocolo ICMP el que se encarga de notificar las anomalías de éste.
-
- *IP bad headers fields:*

Fijando un valor incorrecto de los campos de la cabecera IP, se pretende obtener de la máquina objetivo un mensaje ICMP de error: *ICMP Parameter Problem*. Este mensaje se obtiene cuando un *router* o sistema procesa un paquete y encuentra un problema con los parámetros de la cabecera IP que no está contemplado en ningún otro mensaje de error ICMP. Es enviado sólo si el paquete es descartado debido al error.

Los *routers* deberían generar este tipo de error, pero no todos ellos comprueban que ciertos campos de la cabecera IP son correctos. Las comprobaciones varían en función del *router*, por lo que es posible según su comportamiento identificar el fabricante del mismo. Por ejemplo, típicamente comprueban el *checksum*, y si no es correcto, descartan el paquete.

Los sistemas implementan generalmente la verificación de versión IP, y si no es 4, descartan el paquete. Igualmente, comprueban el valor del *checksum*, para protegerse frente a errores producidos en el transporte de los datos por la red.

Un atacante empleará esta funcionalidad para escanear el rango de IPs completo asociado a una subred. En el caso de que exista un *firewall* protegiéndola, mediante el envío de paquetes falsos a los puertos que supuestamente deberían estar abiertos, como HTTP(80), FTP(21), DNS(53), sendmail(25)..., se podrá saber si hay algún equipo. Los sistemas IDS deberían avisar de este tipo de tráfico anormal.

En el caso de querer descubrir la configuración de ACLs existente, deberá escanearse todo el rango de IPs con todos los protocolos y puertos posibles, de forma que se obtenga la visión más detallada posible de la topología y servicios de la red. Esto permitirá saber las ACLs empleadas, al poder visualizar que tráfico pasa y cual no.

Una red se puede proteger frente a este ataque si los *firewalls* o *screening routers* se encargan de verificar y descartar este tipo de errores, no permitiendo este tipo de tráfico. Asimismo, si el dispositivo de filtrado no implementa esta característica, es posible filtrar los paquetes *ICMP Parameter Problem* en su camino de vuelta.

Existe una herramienta, ISIC: *IP Stack Integrity Check*, de Mike Frantzen [IC-3], disponible para este tipo de pruebas, que permite poner a prueba la pila TCP/IP, encontrar debilidades en un *firewall*, y comprobar la implementación de *firewalls* e IDS. Permite especificar si los paquetes se fragmentan, sus opciones IP, las opciones TCP y el bit URG.

- IP *non-valid field values*:

Es posible modificar un paquete IP para que contenga valores no válidos en algunos de sus campos. Cuando un equipo recibe un paquete de este estilo modificado, generará un mensaje *ICMP Destination Unreachable*.

Por ejemplo, es posible fijar un valor en el campo que especifica el protocolo, que no represente un protocolo válido. Cuando el sistema objetivo reciba este paquete generará el error ICMP. Si no se recibe esta respuesta, podemos asumir que existe un dispositivo de filtrado que no permite que el paquete llegue a su destino, salvo en algunos Unix, como AIX o HP-UX.

Es posible, por tanto, realizar un escaneo probando todos los valores de protocolo posibles, 256 (8 bits). Esta funcionalidad está implementada en la utilidad NMAP. Sí se detectan muchos protocolos

abiertos, indicará que existe un dispositivo de filtrado. Si el dispositivo filtra los mensajes de *ICMP Protocol Unreachable*, entonces parecerá que los 256 protocolos existen y están disponibles.

Existen dos opciones para protegerse frente a este ataque. Por un lado, comprobar que el *firewall* bloquea los protocolos que no están soportados, denegando todo por defecto salvo lo permitido. Por otro, el *firewall* puede bloquear la salida de los paquetes *ICMP Protocol Unreachable* como respuesta al paquete falso.

- *IP fragmentation*:

Cuando un sistema recibe un fragmento de un paquete IP y algunos de los fragmentos del datagrama total se han perdido, y no son recibidos en un periodo de tiempo determinado, el sistema descartará ese paquete. Asimismo, generará un mensaje *ICMP Fragment Reassembly Time Exceeded* hacia el origen de esa comunicación.

Si se realiza un escaneo hacia todos los puertos TCP y UDP del rango de direcciones IPs de la subred a atacar, es posible determinar la configuración de las ACLs existentes que filtran el tráfico. Si se recibe el paquete ICMP de tiempo excedido al reconstruir los fragmentos, quiere decir que el puerto está disponible y sin filtrar; en caso contrario, o está filtrado o cerrado.

Los fragmentos empleados para este escaneo no pueden tener un tamaño menos al de la cabecera IP más la cabecera TCP o UDP.

De nuevo, la contramedida frente a este ataque es no permitir la salida hacia el exterior de los paquetes ICMP de este tipo.

5.5 Sniffing

Un ataque realmente efectivo, ya que permite la obtención de gran cantidad de información sensible enviada sin encriptar, como por ejemplo usuarios, direcciones de e-mail, claves, números de tarjetas de crédito..., es emplear *sniffers* u olfateadores en entornos de red basados en difusión, como por ejemplo *ethernet* (mediante el uso de concentradores o *hubs*). El análisis de la información transmitida permite a su vez extraer relaciones y topologías de las redes y organizaciones.

Los *sniffers* operan activando una de las interfaces de red del sistema en modo promiscuo. En este modo de configuración, el *sniffer* almacenará en un *log* todo el tráfico que circule por la tarjeta de red, ya sea destinado o generado por el propio sistema o desde/hacia cualquiera de los sistemas existentes en el entorno de red compartido (segmento *ethernet*). Asimismo, pueden ser instalados tanto en sistemas como en dispositivos de red [SI-1] [SI-2].

La efectividad de esta técnica se basa en tener acceso (habitualmente es necesario además disponer de dicho acceso como administrador o *root*) a un sistema interno de la red; por tanto, no puede ser llevado a cabo desde el exterior. Antes de la instalación de un *sniffer*, normalmente se instalarán versiones

modificadas (*trojanos*) de comandos como “ps” o “netstat” (en entornos Unix), para evitar que las tareas ejecutadas con el *sniffer* sean descubiertas.

Cuando los *sniffers* se emplean para la obtención de *passwords*, en ocasiones éstos no son necesarios, ya que los administradores de sistemas descuidan los equipos dejándolos configurados con las *passwords* que por defecto proporcionan los fabricantes [PA-1].

Aparte de los programas independientes existentes para ésta tarea, los sistemas operativos poseen *sniffers* en las distribuciones comerciales, típicamente utilizados por el administrador de red para resolver problemas en las comunicaciones:

Software general:

- Network Associates Sniffer
- NetXray
- HP Internet Advisor (dispositivo hardware de escaneo)

Software incluido en múltiples sistemas operativos:

- Unix: `ethereal`
- HP-UX: `nettl`
- Solaris: `snoop`
- Linux: `tcpdump`
- Windows: *Microsoft Network Monitor*
- Cisco IOS: `comandos debug`

5.6 *Eavesdropping*

El *eavesdropping* es una variante del *sniffing* caracterizada porque únicamente contempla la adquisición o intercepción del tráfico que circula por la red de forma pasiva, es decir, sin modificar el contenido de la misma.

5.7 *Snooping*

De nuevo, ésta es otra variante dentro del *sniffing* basada en el almacenamiento de la información obtenida en el ordenador del atacante (*downloading*). También se asocia a la obtención de la información existente en un sistema y no sólo a la extraída del tráfico de red. En este caso, tampoco se modifica la información incluida en la transmisión.

5.8 *IP spoofing*

El *spoofing* como tal, se basa en actuar en nombre de otro usuario tal y como si se fuese él mismo (*impersonation*). En el caso que se está analizando, TCP/IP, se basa en la generación de paquetes IP con una dirección origen falsa. El motivo para realizar el envío de paquetes con esa IP puede ser, por ejemplo, que desde la misma se disponga de acceso hacia un sistema destino objetivo, porque existe un

dispositivo de filtrado (*screening router* o *firewall*) que permite el tráfico de paquetes con esa dirección IP origen, o porque existe una relación de confianza entre esos dos sistemas.

En los equipos Cisco es muy sencillo implementar este ataque, ya que puede configurarse un *interface* de *loopback* (*interface* lógico interno al *router*), al que se le puede asociar la dirección IP deseada. Por ejemplo:

```
router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
router(config)#int loopback 0
router(config-if)#ip address 31.31.31.33 255.255.255.255
router(config-if)#

!
interface Loopback0
 ip address 31.31.31.33 255.255.255.255
!
```

En los S.O. actuales también es posible la configuración de interfaces virtuales que permiten asociarle al sistema más de una dirección IP.

Las direcciones IP seleccionadas para emplear esta técnica deben ser direcciones libres en Internet, ya que si el sistema con la dirección IP falseada existe, el funcionamiento normal del mismo será enviar un paquete de *reset* al recibir un SYN-ACK para el cual no envió un SYN (el SYN fue enviado por la herramienta de *spoofing*). Por tanto, la conexión falseada (*spoofeada*) finalizará. Otra solución es aplicar alguna técnica de DoS sobre ese sistema, para inhabilitarlo e imposibilitar la respuesta de RST.

Existe una modalidad denominada *Blind Spoofing*, que permite llevar a cabo este ataque sin realizar *sniffing* de los datos que se intercambian por la red. Ver “TCP Initial Sequence Numbers”.

5.9 SMTP Spoofing y Spamming

En un nivel superior, concretamente a nivel de aplicación, en el protocolo SMTP (puerto TCP 25) es posible falsear la dirección fuente de un correo o e-mail, enviando por tanto mensajes en nombre de otra persona. Es así porque el protocolo no lleva a cabo ningún mecanismo de autenticación cuando se realiza la conexión TCP al puerto asociado.

El *spamming* consiste en el envío masivo de un mensaje de correo a muchos usuarios destino, pudiendo llegar a saturarse los servidores de correo. Suele emplearse para el envío no deseado de publicidad o información.

5.10 DoS: Denial of Service

En los protocolos TCP/IP, se analizan los ataques basados en la denegación de servicio (DoS) desde el exterior de un sistema, a través de la red, y no una vez se disponga de acceso de administrador en el mismo. En este segundo caso, en los sistemas Unix sería tan sencillo efectuar un ataque de este tipo

como eliminar todos los ficheros del sistema mediante el comando “`rm -rf / &`”, dependiendo el restablecimiento del servicio de la política de backup del sistema.

Si se tiene acceso a los dispositivos de red, éstos pueden rearrancarse o apagarse, con la implicación que tendría en las comunicaciones de la red de la organización afectada.

Un ataque de denegación de servicio se centra en sobrepasar los límites de recursos establecidos para un servicio determinado, obteniendo como resultado la eliminación temporal del servicio. Por ejemplo, si un servidor es capaz de procesar 10 peticiones por segundo, y se le envían 30, parte del tráfico legítimo no recibirá servicio, o incluso, puede que la saturación del tráfico provoque que el servidor deje de responder a ninguna petición. Los destinos de estos ataques suelen ser objetivos visibles, como servidores Web, o DNS, o elementos básicos de la red, *routers* o enlaces de red.

Este tipo de ataques no supone ningún peligro para la seguridad de las máquinas, ya que no modifica los contenidos de la información, por ejemplo páginas Web, ni permite obtener información sensible. Simplemente persiguen entorpecer el acceso de los usuarios a los servicios de un sistema. Normalmente, una vez que el ataque finaliza, se vuelve a la situación normal.

En algunas ocasiones se han empleado para encubrir otros ataques simultáneos cuyo objetivo sí era comprometer el sistema. Asimismo, la probabilidad de que el administrador, intentando defenderse del DoS cometa un error de configuración es mayor en el momento del ataque, pudiendo dejar al descubierto una vulnerabilidad protegida previamente.

Si se desea obtener un análisis más detallado de los diferentes tipos de ataques basados en DoS de los últimos años, 1999-2001, así como sus características, naturaleza y variantes, existe un análisis del CERT al respecto [DD-12]. Los tres protocolos en los que se basan las técnicas de saturación de paquetes, o *flooding*, son TCP, UDP e ICMP. Los ataques y herramientas más empleados en estos años para llevar a cabo ataques DoS y DDoS, algunos de los cuales se analizan en este documento, son: *Smurf*, *Fraggle*, *Trinoo*, *TFN*, *Stacheldraht*, *TFN2K*, *mstream*, *t0rnkit*, *Trinity DDoS*, *erkms*, *li0n*, *carko*, *w0rmkit*, así como algunos virus y/o gusanos, *VBS/LoveLetter*, *Ramen worm*, *VBS/OnTheFly*, *cheese worm*, *sadmin/IIS worm*, *W32/Sircam*, *Leaves*, *CodeRed*, *CodeRed II*, *Knight/Kaiten*, *Nimda*... Asimismo, las vulnerabilidades que se han explotado correspondían a los servidores de nombres (BIND), IIS, e-mail, telnetd, SMB...

A grandes rasgos en este periodo se ha observado que se ha pasado de disponer de los agentes de un DoS solo en plataformas Unix, a disponer de ellos también en el entorno Windows. Esto, unido al incremento de conexiones de banda ancha en los hogares (tecnologías xDSL y cable) ha incrementado la potencia de los ataques que pueden realizarse. Asimismo, debido a la potencia del ancho de banda del que disponen los *routers*, éstos están siendo empleados como agentes para originar ataques DoS contundentes.

En los apartados posteriores se analizarán de forma particular vulnerabilidades que se englobarían dentro de los DoS [DD-11].

5.11 *Net Flood*

El objetivo de este ataque es degradar la capacidad de conexión a la red de un sistema, saturando sus enlaces de comunicaciones. Por ejemplo, si el enlace de una organización dispone de un ancho de banda de 34 Mb. y un atacante dispone de un enlace de 155 Mb., prácticamente la totalidad del tráfico cursado por la organización pertenecerá al atacante, por lo que no podrá enviarse tráfico útil.

Para disponer de altos anchos de banda puede recurrirse a la obtención de múltiples sistemas desde los que efectuar el ataque (ver las vulnerabilidades DDoS) o apoderarse de sistemas mal administrados y protegidos que posean redes de gran capacidad, como por ejemplo, los existentes en las universidades.

Las dos técnicas aplicadas en este tipo de ataques se basan en los protocolos ICMP y UDP, al tratarse de protocolos no orientados a conexión y que permiten el envío de paquetes sin requisitos previos: *ICMP Flood* y *UDP Flood*.

5.12 *Smurf*

Dentro del concepto de *Net Flood*, existe una técnica que se aprovecha de las características de *broadcast* de las redes: el *Smurf*. Previamente denominado *nuking* (ver apartado de *Winnuke*).

La dirección lógica de *broadcast*, es decir, aquella que representa a todas las máquinas de una red, se utiliza en algunos protocolos para localizar el sistema que proporciona un servicio concreto de forma sencilla, es decir, preguntando a la red, y no consultando uno por uno a todos los sistemas existentes.

Si esta dirección se encuentra disponible también para usuarios externos a la red, es posible que un atacante pueda enviar un paquete de datos a la misma, provocando que todos los sistemas pertenecientes a dicha red respondan simultáneamente, aumentando la potencia de la respuesta en un factor de N, siendo N el número de máquinas disponibles en la red. Es decir, se realiza un ataque a una red desde otra red intermedia que permite multiplicar los recursos existentes (elementos válidos para desarrollar ataques DDoS). Este método no implica tener que controlar las redes empleadas como multiplicadoras del efecto de ataque.

Si se aúna esta técnica junto a la de *IP spoofing*, al enviar un paquete ICMP con la dirección IP origen de la máquina a atacar y dirección IP destino la dirección de *broadcast* de una red con un elevado número de máquinas, digamos cientos, todas las respuestas de la red de *broadcast* se dirigirán realmente a la dirección IP del sistema “*spoofeado*”. Este ataque es realmente denominado *Smurf* [SM-6].

5.13 *TCP Syn Flood*

Dentro de los ataques DoS, existe uno asociado directamente al protocolo TCP. Consiste en el envío masivo de paquetes de establecimiento de conexión (SYN) contra un sistema. La recepción de estas solicitudes provoca que el sistema destino, objetivo del ataque, reserve cierta cantidad de memoria (*buffers*) para almacenar las estructuras de datos asociadas a cada una de las nuevas conexiones en curso [SY-1].

El protocolo TCP requiere del establecimiento de una conexión, que se realiza en tres pasos. Tras la recepción del paquete SYN, responderá con su paquete SYN-ACK, permaneciendo a la espera del paquete final (ACK) que confirma el establecimiento de la conexión TCP (*three-way handshake*). La conexión permanece en el estado semiabierto, concretamente SYN_RCVD. El atacante no enviará nunca ese ACK esperado, por lo que la memoria del destino es copada en su totalidad por conexiones falsas, no siendo posible el establecimiento de conexiones de clientes reales, y por tanto anulándose el servicio [SY-2].

Asimismo, ciertos sistemas imponen un número máximo de conexiones en este estado, por lo que una vez alcanzado éste, no será posible establecer más conexiones. Tras un periodo de tiempo controlado por un temporizador (que suele ser de 2 minutos), las conexiones que continúan en este estado expiran, permitiendo la creación de nuevas conexiones. Esto solo será posible si el ataque *TCP SynFlood* ha cesado, ya que mientras se mantenga, serán sus nuevos inicios de conexión los que ocuparán el espacio de memoria liberado por las sesiones expiradas.

Suponiéndose un número máximo de conexiones igual a 30, y el temporizador igual a 2 minutos, se podría desarrollar un ataque de este tipo enviando un paquete SYN cada 4 segundos: tiempo necesario por cada conexión para poder enviar el máximo de 30 conexiones en los 120 segundos de expiración.

Normalmente, para que su detección sea más compleja este ataque se realiza variando la dirección IP del emisor, mediante direcciones falsas (*IP spoofing*), de forma que se simule de forma más fehaciente una situación real de conexiones realizadas por multitud de clientes.

Algunas herramientas dedicadas a este tipo de ataques son: “neptune” y “synk4”.

5.14 Connection Flood

Los servicios TCP orientados a conexión, que son la mayoría (telnet, ftp, http, smtp, nntp...) tienen un límite máximo de conexiones simultáneas soportadas; cuando este límite se alcanza, cualquier conexión nueva es rechazada.

De forma similar al *Syn Flood*, si un atacante es capaz de monopolizar el límite definido con conexiones de su propiedad, que simplemente son establecidas pero por las que no se realiza ninguna comunicación posterior, el sistema no proporcionará servicio.

Al igual que antes, las conexiones expiran progresivamente con el paso del tiempo, pero un ataque constante de apertura de conexiones mantendrá continuamente el límite en su valor máximo. La diferencia está en que en este caso la conexión se ha establecido y por tanto se conoce la identidad del atacante (dirección IP), y a su vez, la capacidad del sistema o sistemas atacante/s debe ser lo suficientemente elevada como para mantener abiertas todas las sesiones que colapsan el servidor atacado.

Existe una variante de estos ataques basada en el uso de un cliente que establezca conexiones contra un sistema, pero que no las finalice de forma correcta, de modo que en el servidor los *sockets* correspondientes a estas comunicaciones seguirán estando activos y consumiendo recursos, concretamente en el estado TCP denominado TIME_WAIT.

5.15 SMTP Flood

Mediante el envío masivo de mensajes de correo electrónico a grandes listas de usuarios de forma continua, se provoca la saturación de los servidores de correo destino o intermedios.

5.16 DDoS

Una variante más potente a la de los ataques de Denegación de Servicio, son los DDoS, o ataques de denegación de servicio **distribuidos** [DD-2], que se basan en realizar ataques DoS de forma masiva a un mismo objetivo desde diferentes localizaciones en la red, de forma que la potencia de ataque sea mucho mayor. Si un ataque desde una fuente es potente, desde 1000 lo será mucho más, es decir, es la aplicación del “divide y vencerás” a la técnica DoS.

Su origen se remonta a los comienzos de la seguridad en Internet, cuando el famoso *Gusano* de Robert Morris, Jr [BO-3], desencadenó una denegación de servicio por un error de programación. El gusano fue capaz de colapsar por aquel entonces gran parte de los sistemas existentes en Internet. Sin embargo su expansión se ha producido principalmente en el año 2000, haciéndose eco los medios de comunicación [DD-1], al surgir numerosas herramientas que permiten su ejecución, de forma coordinada y a gran escala. Un único atacante puede desencadenar una agresión desde centenares de máquinas repartidas por todo el mundo, como ha ocurrido en las Webs de Yahoo, Amazon, CNN, eBay, Buy, ZDNet... Dado el elevado número de sistemas existentes en Internet, la capacidad de “reclutar” recursos es inmensa.

Debido a las vulnerabilidades de los sistemas operativos y de las aplicaciones, como los *buffer-overflows* y los *format-strings*, un atacante es capaz de apoderarse de un conjunto de sistemas (de cientos a miles) e instalar en ellos un servicio que acepte órdenes del atacante para ejecutar un DDoS contra una máquina objetivo [DD-9]. La sofisticación de las herramientas actuales [DD-10] hace que no se requieran conocimientos técnicos avanzados para llevar a cabo este tipo de ataques: ellas se encargan de analizar y vulnerar los sistemas, para copiarse e instalarse automáticamente (en unos segundos).

El proceso esta compuesto de 4 pasos principales:

- 1) Fase de escaneo con un conjunto objetivo de sistemas muy elevado, 100.000 o más. Se prueban éstos frente a una vulnerabilidad conocida.
- 2) Se obtiene acceso a parte de esos sistemas a través de la vulnerabilidad.
- 3) Se instala la herramienta de DDoS en cada sistema comprometido.
- 4) Se utilizan estos sistemas para escanear y comprometer nuevos sistemas.

El modo de operación genérico de las herramientas de DDoS tiene la siguiente topología: el intruso se comunica mediante comandos con un elemento denominado *handler*. Éste se encarga de gestionar el registro, realizado previamente, de un conjunto de agentes, normalmente elevado en número, que son realmente el origen de los paquetes del DDoS. Por tanto, los agentes y el *handler* conforman una red

de ataque, que actúa en el momento en que el *handler* retransmite a todos y cada uno de los agentes las órdenes invocadas por el intruso remotamente. Las comunicaciones entre estos elementos se realizaban originalmente por puertos fijos y, a la larga, conocidos, por lo que este modo de funcionamiento podía ser detectado por sistemas IDS con facilidad. La difusión en el uso del IRC o *chat*, a dado lugar a la utilización de este medio (y sus puertos TCP asociados, del 6660 al 6669) para constituir los canales de control de los elementos de un DDoS.

5.17 Trinoo

Esta herramienta de DDoS, aunque antigua [TR-2], permite el acceso a través de autenticación basada en claves (mediante `crypt()`), y a su vez, permite determinar si un binario concreto de una máquina actúa como maestro o como esclavo. Para ello emplea la técnica comentada en [TR-1]. Inicialmente surgió por la explotación de un *buffer-overflow* en los sistemas que actuaban de víctimas. Existe una versión asociada a Windows, llamada *WinTrinoo*.

Los sistemas maestros disponen de una lista con los *hosts* que pueden ser controlados, a través de los que se puede realizar el ataque distribuido. La comunicación entre los maestros y los esclavos (o demonios) no está encriptada, y se realiza típicamente (por defecto) a través de los siguientes puertos, por lo que es sencillo de detectar si no se han modificado:

TCP: 1524 27665 (*client-master*)
UDP: 27444 31335 (*master-server*)

5.18 Tribe Flood Network y TFN2K

La comunicación entre clientes y servidores se realiza a través de paquetes de ping: *ICMP echo request* e *ICMP echo reply* [TF-1] [TF-2], aunque posibilita ataques DoS basados en *ICMP flood*, *SYN flood*, *UDP flood*, y *Smurf*, así como obtener una *shell* de *root* asociada a un puerto TCP seleccionado.

La comunicación entre clientes y servidores no emplea puertos concretos. Éstos pueden determinarse en el momento de la ejecución o pueden elegirse aleatoriamente en el propio programa, pero consisten en una combinación de los protocolos ICMP, TCP y UDP [TF-4].

Asimismo añade capacidades de encriptación, eliminando así la detección por los sistemas IDS.

5.19 Stacheldraht

Esta herramienta es una combinación de las anteriores, creando una sesión *telnet* encriptada entre clientes y servidores [ST-1] [ST-2]. La comunicación se realiza típicamente (por defecto) a través de los siguientes puertos, por lo que es sencillo de detectar si no se han modificado:

TCP: 16660 65000
ICMP echo request e *ICMP echo reply*

5.20 *Ping of death*

Conocido como ping de la muerte [PI-1], este ataque se basa en enviar un paquete de ping (*ICMP echo request*) de un tamaño muy grande. Teniendo en cuenta que el tamaño máximo de paquete en TCP/IP es de 64 *Kbytes* (65535 *bytes*), la implementación de la pila TCP/IP asigna un *buffer* en memoria de este tamaño. En el caso de que la información sea mayor, el buffer puede desbordarse. El resultado obtenido en muchas ocasiones es que el sistema destino deja de proveer servicio al bloquearse, ya sea rearrancándose (*reboot*) o incluso apagándose (*shutdown*). Lo que sucede realmente es que el paquete emitido es fragmentado, a nivel de IP, en las redes intermedias, los fragmentos van siendo encolados en el sistema destino hasta que se reciben los último pedazos (un paquete de 65536 *bytes* es suficiente), que son los que desbordan el buffer, provocando un comportamiento anómalo.

En el momento de la aparición de este ataque (1996) muchas de las implementaciones TCP/IP, tanto de Unix, de Windows, como de los dispositivos de red fueron vulnerables. Simplemente mediante el envío de un paquete cuyo campo de datos sea mayor de (65507 *bytes* = 65535 – 20 – 8) se puede llevar a cabo.

Debe tenerse en cuenta que los SS.OO. actuales no permiten al cliente de *ping* enviarlo, obteniéndose respuestas como “*ping: illegal packet size*” (HP-UX) o “*Bad value for option -l, valid range is from 0 to 65500*” (Windows 2000):

```
Unix:      # ping victima.com 65510
Windows:   c:\>ping -l 65510 victima.com
```

Para realizar este ataque es necesario disponer de una herramienta que lo implemente o modificar el límite impuesto en el código fuente del cliente de *ping*, por ejemplo en Linux.

Asimismo, existen otros ataques basados en la fragmentación de paquetes ICMP [FA-1]. Todos los tipos de paquetes ICMP son de un tamaño reducido, por lo que la detección de un paquete de gran tamaño debería dar lugar a sospechas.

5.21 *Loki*

Este ataque fue inicialmente el nombre de un proyecto, pasando posteriormente a convertirse en una herramienta [LK-1] cuyo objetivo es demostrar la posibilidad de encubrir tráfico en túneles ICMP y UDP, bajo lo que se ha dado en denominar canales encubiertos. En el caso de que este tráfico este permitido a través de los *firewalls*, el ataque es posible. Debe tenerse en cuenta que en la mayoría de ocasiones es necesario habilitar al menos ciertos tipos de paquetes ICMP, como los de la familia *unreachable*, para que funcionalidades de la pila TCP/IP se desarrollen, por ejemplo, el algoritmo PMTUD, *Path MTU Discovery*.

El objetivo del ataque es introducir tráfico encubierto, típicamente IP, en paquetes ICMP (o UDP) que son permitidos. La herramienta consta de un cliente, *loki*, y un servidor, *lokid*, que se encargan de encapsular y desencapsular el tráfico en ambos extremos.

5.22 *Land*

Este ataque [LN-1] [TD-3] permite bloquear un sistema, mediante el envío de un paquete SYN cuya dirección IP fuente y destino es la misma. Existe una variación de este ataque, basada en que los puertos origen y destino también son iguales. Para ello es necesario enviar paquetes IP mediante la técnica de *spoofing*.

Debe tenerse en cuenta que algunos sistemas IDS detectan la primera situación y otros la segunda. Por tanto, podría darse algún caso en el que se establezca una conexión a la propia máquina, se envíe por tanto un paquete [127.0.0.1:puerto_cliente ==> 127.0.0.1:puerto_servidor], y el sistema IDS lo detecte como un ataque cuando en realidad no lo es. Este ejemplo, aplicable a un gran número de las vulnerabilidades mencionadas, refleja la estrecha línea existente entre un ataque real y una situación convencional, denotando que su detección y automatización no es trivial.

Esta técnica afecta a implementaciones tanto de sistemas como de dispositivos de red, como los equipos Cisco [LN-2]. Las versiones afectadas de Cisco pueden obtenerse de una página en la que publican los denominados “*Field Notices*”, es decir, noticias de implementación relacionadas con sus productos [LN-3].

El *exploit*, al igual que muchos de los comentados a lo largo de los diferentes apartados posee la extensión “.c”, por estar programado en lenguaje C, por tanto se denomina “*land.c*”. Existe un ataque similar que en lugar de enviar un único paquete TCP, envía grupos de éstos a la vez que realiza un escaneo de puertos en un rango concreto de direcciones. Éste se denomina “*latierra.c*”.

5.23 *Routing protocols*

Los protocolos de enrutamiento pueden ser vulnerados principalmente mediante la introducción de paquetes de actualización de rutas, de forma que es posible adecuar y condicionar los caminos que seguirá el tráfico según un criterio específico.

Uno de los protocolos que puede ser falseado (*spoofing*) es RIP [RP-1], en su versión 1, RFC 1058, y 2, RFC 1723. Se trata de un protocolo UDP (puerto 520), por tanto acepta paquetes de cualquier sistema sin necesitar ninguna conexión previa. La versión 1 no dispone de sistema de autenticación, mientras que la versión 2 presenta un método basado en el envío de claves en claro de 16 *bytes*.

Para vulnerar RIP, como se especifica a continuación, es necesario inicialmente identificar un *router* que hable este protocolo a través de la identificación del puerto UDP 520. En el caso de pertenecer al mismo segmento de red, deben escucharse las actualizaciones RIP que circulan por la red o solicitárselas directamente a alguno de los *routers*. De esta forma se obtendrá la tabla de rutas que se anuncia en ese momento. Si no se está en el mismo segmento, se dispone de herramientas como *rprobe* para realizar una petición RIP remota: el resultado se obtendrá mediante un *sniffer* en el sistema desde el que se ataca.

Una vez definida la información que se pretende inyectar en la tabla de rutas anunciada, por ejemplo, redireccionar todo el tráfico a un sistema desde el que se pueda analizar el mismo, mediante utilidades

como `strip`, se inyectará la ruta deseada. A partir de ese momento todo el flujo de tráfico pasará por el nuevo camino definido. Para que el funcionamiento habitual no se vea modificado, es necesario que el nuevo sistema al que van destinado los paquetes los redireccione consecuentemente: *ip forwarding*.

A lo largo del texto se muestra, en los diferentes análisis de las vulnerabilidades y sus defensas, como modificar los parámetros TCP/IP en el *kernel* de los distintos sistemas operativos mencionados.

El primer ejemplo de configuración de este tipo se aplica a la capacidad de los *kernels* para hacer *routing* entre varios interfaces. Ésta se configura como sigue:

- En HP-UX basta con introducir en el fichero `/etc/rc.config.d/nddconf`:

```
TRANSPORT_NAME[1]=ip
NDD_NAME[1]=ip_forwarding
NDD_VALUE[1]=1
```

- En Solaris basta con ejecutar mediante *ndd* en un *script* de arranque:

```
ndd -set /dev/ip ip_forwarding 0
```

- En Linux, se debe añadir también en un *script* de arranque:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

En el caso de Linux es posible especificar los parámetros en el fichero `/etc/sysctl.conf` (p.ej., Red Hat) que es cargado por la utilidad `/sbin/sysctl`. La documentación al respecto se encuentra en el directorio `/Documentation` del *kernel*, en el fichero `proc.txt`.

- En Windows las modificaciones se realizan a través del editor del registro (`regedt32.exe`); para ello debe localizarse la clave:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
```

Bajo la misma es necesario añadir el valor:

Value Name: `IPForwarding`

Data Type: `REG_DWORD`

Value: `1`

5.24 Session hijacking

Considerando la importancia de la información transmitida a través de las redes de datos, y las medidas de seguridad que deben desarrollarse, esta técnica pretende mostrar la posibilidad de apoderarse de una sesión ya establecida. Este avance podría suponer el obviar todo el proceso de autenticación previo.

El *TCP hijacking* puede realizarse en entornos de red de difusión, basado en introducir paquetes en medio de una transmisión como si provinieran del dispositivo original (*IP spoofing*). Este tipo de

ataques también se conoce como “Man in the middle attack”, ya que el atacante debe situarse entre el equipo que estableció la conexión original y la víctima [HI-4].

Para poder tomar el control de una conexión previamente es necesario obtener la información asociada a como transcurre ésta a lo largo del tiempo, concretamente en TCP, deben conocerse los números de secuencia actuales, ya sea directamente o a través de los ISNs y del número de *bytes* transmitidos. Una vez conseguido el control, el objetivo será ejecutar algún comando, típicamente se pretende apoderarse de sesiones de terminal, que permita acceder al sistema remoto de forma directa.

Habitualmente el control de la sesión se realiza empleando técnicas como *source-routing*, de forma que los paquetes de vuelta lleguen al atacante y no al destino real. En caso de no disponer de esta facilidad la técnica se conoce como *blind-hijacking* y se basa en adivinar o intuir las respuestas de los sistemas que intervienen en la comunicación. Para obtener la información de la conexión existente debe emplearse un *sniffer*, situándose entre los sistemas que se están comunicando, ataque conocido como “*man-in-the-middle attack*” [HI-3].

Existen dos herramientas principales para llevarlo a cabo, aunque numerosos *sniffers* también incluyen esta funcionalidad, por ejemplo *ethereal*: se denominan *Juggernaut* y *Hunt* [HI-1] [HI-2].

Asimismo, existen métodos para apoderarse de las conexiones encriptadas, por ejemplo de SSH v.1 o SSL. Existen herramientas como “*dsniff*” [HI-6] o “*ettercap*” [HI-7] que facilitan aplicar la técnica de *hijacking* en estos entornos [HI-5].

5.25 Source routing

Esta funcionalidad propia del protocolo IP permite enviar dentro del mismo paquete de datos la información necesaria para su enrutamiento, es decir, la dirección IP de cada uno de los dispositivos de red intermedios que deben cruzarse hasta llegar al destino final.

Esta característica puede emplearse para tareas de verificación y configuración de los enlaces, pero desde el punto de vista de la seguridad supone que un atacante es capaz de manejar por dónde deben viajar sus paquetes IP, saltándose todas las reglas de enrutamiento definidas en los dispositivos de red. Asimismo, puede permitir la realización de pruebas para conocer las redes internas y también permitir a un atacante el alcanzar redes con IPs internas (RFC 1918).

Unida a la técnica de *IP spoofing* permite que un atacante se haga pasar por otro sistema IP, siendo capaz de enviar y recibir todas las respuestas asociadas a una comunicación falseada.

Una de las principales dificultades asociadas al *spoofing* es que realmente no se puede simular de forma totalmente real la dirección falseada, ya que la red enviará las respuestas generadas por el sistema atacado al equipo que verdaderamente posee la dirección IP falsa. Por lo tanto, para tener éxito en el ataque es necesario eliminar momentáneamente al equipo al que se le ha robado la dirección IP, por ejemplo mediante un ataque DoS. En el caso de poder emplear el *source routing*, ésto no sería necesario.

5.26 ICMP redirects

Existe un tipo de paquete ICMP que es empleado por los dispositivos de enrutamiento para informar de las alternativas de rutas por las que debe dirigir el tráfico un sistema ejecutando el protocolo IP. En algunos casos estos mensajes tienen la utilidad de aconsejar a un sistema del camino a seguir, es decir, el siguiente equipo al que debe enviarle los paquetes IP. En el caso de cambios en la red la redirección funciona de manera dinámica.

La vulnerabilidad asociada a esta funcionalidad se basa en generar paquetes de redirección hacia un sistema objetivo, de forma que se oriente su flujo de tráfico hacia sistemas controlados por el mismo atacante, para por ejemplo, analizarlo mediante un *sniffer* o incluso realizar cualquier tipo de modificación en los datos.

Realmente, si un sistema acepta la recepción de este tipo de paquetes, lo que hace es actualizar su tabla de rutas con una entrada de tipo dinámica, que le indica la nueva ruta a seguir.

5.27 Directed broadcast

Este tipo de tráfico puede dar lugar a la existencia de redes amplificadoras de tráfico empleadas en ataques de tipo DoS como *Smurf* (ver apartado correspondiente) [DB-1].

Es necesario comentar que existen otros tipos de paquetes de *broadcast* que permiten a un atacante extraer información valiosa de la red y su composición. Mediante el uso de paquetes *broadcast* de máscara de red, un atacante puede obtener los bloques de redes empleados para posteriormente emplear rangos de IPs precisos en sus escaneos. Asimismo, mediante la utilización de *broadcast* de tipo *timestamp*, el atacante puede extraer información de identificación de los sistemas existentes.

5.28 SNMP

El protocolo SNMP, *Simple Network Management Protocol*, también conocido como “*Security Not My Problem*” ;-), al menos hasta la versión 3, permite la gestión y administración de dispositivos de red: RFC 1157 (versión 1) y RFC 1446 (versión 2). La seguridad de la primera versión se basa en el uso de claves conocidas como *community names*, mientras que la versión 2 gestiona la integridad mediante el uso del algoritmo MD5, y permite encriptación, mediante DES, pero ésta no limita el uso de claves simples.

La versión 3, RFC2570, profundiza más en la seguridad de los dispositivos [SP-1].

A grandes rasgos existen dos tipos de comunidades: lectura (RO) y lectura-escritura (RW). Cada clave asociada permite realizar la operación referida, por lo que la clave RW permitiría modificar la información del dispositivo de red, mediante el comando `snmpset`. Una vulnerabilidad habitual son los ataques de fuerza bruta sobre el “*public / private*” *community string*.

Los *routers* suelen disponer de agentes SNMP con grandes cantidades de información acerca de la red debido a su función y situación, por lo que constituirán uno de los objetivos principales de un escaneo SNMP.

Para explotar las vulnerabilidades asociadas a este protocolo, basta con disponer de una utilidad como `snmpwalk`, disponible para numerosos sistemas Unix, que permite obtener la información almacenada en la MIB de un sistema conectado a la red de forma remota. En el caso de que las comunidades de lectura y escritura del sistema no se hayan modificado, dispondrá de las establecidas en el estándar, que son respectivamente “public” y “private”.

Podrá obtenerse información de cualquier objeto de la MIB como sigue (en Unix):

```
# snmpwalk <<dirección_IP | nombre_host>> COMUNIDAD [id_objeto]
```

Por ejemplo, para obtener la información de los interfaces de red de un equipo:

```
# snmpwalk host99 public interfaces
```

5.29 TCP Initial Sequence Numbers

El protocolo TCP genera un ISN, o número de secuencia inicial, para poder realizar el control de flujo de la conexión. Este es uno de los ataques más antiguos, data de 1985, y se basa en la utilización de *pseudo-random number generators* (PRNGs) para la generación de los ISNs. Si los números de secuencia pueden ser predichos, puede llegar a ser posible el modificar la información de la conexión, apoderándose de ella mediante *hijacking* o realizar *blind spoofing* sobre futuras conexiones [IS-1] [IS-4] [IS-7].

La modificación de los datos en la conexión puede realizarse inyectando paquetes válidos, al conocerse el ISN inicial y el número de *bytes* intercambiado, y por tanto, el número de secuencia actual. Si no se conoce exactamente este valor, pero sí de forma aproximada, puede enviarse también un grupo de paquetes en un rango de secuencia concreto (que vendrá limitado por el tamaño de ventana TCP), con el objetivo de que alguno coincida con el número de secuencia actual [IS-3].

Inicialmente se modificaron las implementaciones para hacer lo más aleatoria posible la generación de estos números.

La generación de números aleatorios, tanto en los números de secuencia iniciales de TCP, *ISNs*, como en los algoritmos de encriptación y generación de claves, tiene un peso muy relevante respecto a la seguridad. Una dirección interesante y muy particular al respecto es <http://www.lavarnd.org/> (antiguamente <http://lavarand.sgi.com/>), dónde se muestra la generación de números aleatorios en función de la instantánea en un momento dado de una *lava lamp*.

En el sistema operativo Linux se han introducido mecanismos para la obtención de números aleatorios más reales, mediante dos ficheros de dispositivo, `/dev/random` y `/dev/urandom`, que gestionan la entropía existente en el sistema según su actividad.

Actualmente se ha encontrado una nueva vulnerabilidad [IS-2] que se presenta cuando se usan incrementos aleatorios, al aumentar constantemente el valor de los ISNs generados. Debido a las implicaciones del teorema del límite central, el sumatorio de una serie de números no proporciona la

suficiente varianza en el rango de valores de ISN deseados, por lo que un atacante puede apoderarse de las conexiones. Por tanto, los sistemas basados en la generación de números mediante incrementos aleatorios son vulnerables a ataques estadísticos.

Esta debilidad en las implementaciones del protocolo TCP permite emplear una técnica conocida como *Blind Spoofing*, en la que se realiza un ataque de *IP Spoofing* pero sin la posibilidad de interceptar el tráfico de red intercambiado entre los sistemas. Para poder mantener la comunicación, es necesario “adivinar” los números de secuencia empleados por el sistema a atacar.

5.30 *Tiny Fragment Attack*

Para comprender este ataque debe considerarse como tiene lugar la fragmentación de paquetes TCP sobre IP. Cuando un paquete IP supera el tamaño máximo de transmisión, MTU, debe dividirse en paquetes menores. El primero de ellos incluirá la cabecera TCP asociada al paquete original, mientras que el resto de fragmentos simplemente contendrán la cabecera IP y los datos, pero no información de TCP (cabecera TCP). A través del campo de *fragment offset* de la cabecera IP se determina si existen más fragmentos y la relación entre éstos.

Cuando se gestiona un sistema de filtrado de paquetes, lo habitual es permitir que los fragmentos de un paquete IP pasen el filtro, ya que no se dispone de información TCP para tomar una decisión de filtrado en función, por ejemplo, de los puertos origen y destino.

La técnica presentada [FA-1] pretende enviar un paquete TCP inicial con la siguiente información: SYN=0, ACK=1, FO=”*more packets follow*”. De esta forma, el paquete puede atravesar un filtro concreto (*stateless*), al no disponer del *flag* SYN activo.

Este paquete no sería peligroso de no ser porque el tamaño de *offset* (20 bytes) es lo suficientemente pequeño como para sobrescribir ciertos campos de la cabecera TCP mediante el paquete que representa el supuesto fragmento esperado a continuación. Este segundo paquete en el proceso de fragmentación cambiará los valores de TCP a SYN=1, ACK=0, por tanto, se tendrá un paquete de establecimiento de conexión reconstruido en la máquina destino, aunque los filtros explícitamente no permiten el establecimiento de conexión desde ese sistema IP origen.

Por ejemplo, al enviar un paquete de 8 bytes, suficiente para contener los puertos fuente y destino (además del número de secuencia), se obligará a recibir los *flags* TCP en el siguiente paquete. Este segundo paquete o fragmento no posee cabecera TCP, por lo que el filtro no se podrá aplicar sobre él, ni tampoco en el resto de fragmentos. Realmente, el campo de datos del segundo fragmento, contiene el resto de la cabecera TCP tras los 8 bytes, es decir, los *flags* TCP.

5.31 *Winnuke*

Este ataque [WN-4] [WN-5] [WN-10] afecta a los sistemas que utilizan el protocolo NetBIOS sobre TCP/IP, típicamente en el sistema operativo Windows. Este protocolo emplea los puertos UDP 137, 138 y 139. El envío de un paquete urgente (*bit* URG=1), conocido como paquete “*Out of Band*” (OOB) [WN-6] da lugar al envío de datagramas UDP a estos puertos, que al intentar ser enviados a las

capas superiores, pueden provocar que el sistema destino se “cuelgue” o que disminuya su rendimiento de forma notable [WN-2] [WN-8] (ésta referencia contiene menciones a otros ataques de los protocolos TCP/IP).

Existe una página Web que permite probar la eficacia de este ataque contra un sistema concreto [WN-1]. Asimismo, existen *exploits* similares, como *supernuke* [WN-9].

El termino *Nuking* (*nuke.c*) no debe ser confundido con *Winnuke*. Es una técnica antigua, por lo que no funciona en los sistemas modernos, y para ser ejecutado debe tenerse privilegio de *root*. El ataque se basa en enviar fragmentos o paquetes ICMP no válidos, con el objetivo de ralentizar al objetivo o incluso bloquearlo. Posteriormente surgió una variante de éste denominada *Smurfing* (ver apartado).

5.32 Teardrop

El ataque *teardrop* [TD-1] [TD-3] [TD-5] se basa en el envío de fragmentos de paquetes en lugar de paquetes completos. Se comprobó que algunas implementaciones de la pila TCP/IP no eran capaces de reconstruir paquetes con fragmentos cuyos *bytes* se superponen. El resultado es de nuevo que el sistema destino puede llegar a bloquearse [TD-4] [TD-9]; apareció en Linux inicialmente. Para llevarlo a cabo bastaría con 2 paquetes, A y B, dónde el *offset* del paquete B indica que comienza dentro del paquete A

Existen dos versiones de este ataque: *teardrop* y *teardrop2* [TD-7]. La variación de la segunda respecto a la primera se basa en la inclusión del *flag* de urgencia (URG) en la cabecera TCP de los fragmentos. Por ejemplo, Windows NT 4 SP3 se parcheó frente a la primera versión, pero era vulnerable a la segunda.

Existen variantes de *teardrop* en las que el paquete enviado tiene el *flag* SYN activo, como “*syndrop.c*” [TD-2], así como otras orientadas a sistemas Windows, “*bonk.c*”.

5.33 DNS

DIG es una utilidad para la obtención de información del servicio de nombres DNS [DG-1] [DG-2].

Como ya se mencionó en el apartado de *footprinting*, el DNS es una fuente de información de red muy valiosa. La utilidad mencionada permite copiar una base de datos entera de nombres (dominio) desde un servidor DNS, para su posterior análisis. Asimismo sus características avanzadas facilitan extraer toda la información asociada al protocolo DNS, no permitiendo únicamente la realización de peticiones, como *nslookup*.

Existen páginas Web con servicios basados en DIG [DG-3].

5.34 NTP

El protocolo NTP, *Network Time Protocol* [NT-1], permite sincronizar la hora de forma simultánea en todos los equipos de una red. Este protocolo presenta diferentes vulnerabilidades, ya que por ejemplo, sistema de alta disponibilidad configurados en cluster se basan en el momento horario de cada uno de sus nodos para gestionar el *cluster* que ofrece el servicio.

En el caso de poder modificar la hora en un nodo, podrían obtenerse resultados inesperados en el conjunto de ellos, como por ejemplo que se desconfiguraran ciertos nodos, no formando parte del *cluster*, pudiendo llegar a anularse la alta disponibilidad.

Mediante el comando `ntpdate` se puede modificar la hora de un sistema:

```
# ntpdate -d <<dirección_IP>>
```

Mediante el comando `ntpq` se pueden hacer consultas al servicio NTP de un sistema. Por ejemplo para ver las asociaciones de un equipo, es decir, de quién obtiene la hora:

```
# ntpq -p <<dirección_IP>>
```

5.35 Caballos de Troya o *Trojanos*

Pese a que esta vulnerabilidad está más asociada a los sistemas y no ha TCP/IP, se presenta una visión general, ya que en numerosas ocasiones, es empleada para introducir servicios TCP/IP no deseados en sistemas destino y poder así ejecutar ataques remotos posteriormente o incluso tomar su control por completo.

También conocidos como puertas traseras (*back doors*), son fragmentos de programas no autorizados que se introducen en otros para que el programa original ejecute ciertas acciones no deseadas. En el caso de los *trojanos* que afectan a los servicios TCP/IP más directamente, se trata de programas completos, que normalmente se justifican como herramientas de administración remota o RAT (típicamente de Windows) como por ejemplo:

- Back Orifice (<http://www.blackhat.com> y <http://www.cultdeadcow.com/tools/>).
- Back Orifice 2000 (<http://www.bo2k.com> - <http://www.bo2k.de>).
- NetBus (<http://www.netbus.org>).
- SubSeven (<http://www.sub7files.com> - <http://www.sub-seven.com/>).

Asimismo, las herramientas típicas de administración y acceso remoto podría incluirse en este grupo, ya que facilitan el acceso y el completo control del sistema destino. Entre estas se encuentran *PCAnywhere*, *VNC*, *Windows Terminal Services*...

Habitualmente este tipo de software se descarga en los sistemas objetivo al visitar alguna página Web o servicio electrónico público sin que el usuario se percate de ello. Las consecuencias y acciones de cada herramienta pueden variar en función de la idea con la que se diseñaron: desde conectarse a canales IRC, a distribuirse para actuar como fuente futura de ataques DDoS o incluso manipular y extraer información del sistema que les hospeda.

5.36 IPSec

La seguridad del estándar IPSec ha sido analizada en numerosos estudios, poniéndose en entre dicho como característica negativa la complejidad de sus especificaciones y del propio protocolo [IP-7]. Aunque el análisis también está enfocado desde un punto de vista político centrado en el control de la encriptación, denota que es la implementación de los diferentes fabricantes la que determinará su seguridad al 100%, en función de si los estándares son respetados y la interoperabilidad posible.

Los dispositivos que “hablan” IPSec pueden ser identificados por tener el puerto 500 escuchando, ya que es el asociado al estándar de intercambio de claves o IKE, *Inter Key Exchange protocol*.

Por otro lado, la debilidad desde el punto de vista de la seguridad, no es tanto la vulnerabilidad del propio IPSec, como la de los algoritmos de encriptación asociados al mismo, como el RSA [IP-8] y RC-5 [IP-9]. Debe tenerse en cuenta que en el intercambio de información mediante IP, gran parte de la información asociada a los protocolos es conocida, por lo que puede emplearse como texto en claro para romper la encriptación [IP-10] mediante, por ejemplo, ataques estadísticos.

Existe un estudio detallado sobre las vulnerabilidades de protocolos de túneles como PPTP [VP-2] [VP-3]. En éste se muestra la posibilidad de aplicar la técnica de *spoofing* para adquirir credenciales de autenticación en un entorno PPTP entre un cliente, ya conectado a Internet (no realizando una conexión conmutada o de *dial-up*) y un servidor; no así entre servidores.

5.37 Finger Bomb

Existe otro tipo de ataque DoS que permite forzar al sistema destino a un consumo elevado de CPU realizando una petición *finger* recursiva [FI-1] [FI-2]. Asimismo, se dispone de *scripts* como *kaput* que hacen uso de esta vulnerabilidad [FI-3].

5.38 RPC

Existe una tecnología de red, inventado originalmente por Sun Microsystems, que permite invocar procedimientos y acciones de forma remota desde otro equipo. Ésta se denomina RPC, *Remote Procedure Call*.

Es posible obtener la lista de servicios activos en un equipo mediante el siguiente comando en Unix:

```
# rpcinfo -p <<dirección_IP>>
```

5.39 Relaciones de confianza entre sistemas

Los comandos *r-Unix* (*rsh*, *rcp*, *rlogin*...) requieren de relaciones de confianza entre sistemas para accederse entre sí directamente, esquivando los controles de seguridad y autenticación habituales. Esto significa que si se es capaz de vulnerar uno de los sistemas incluidos en el círculo de confianza, se dispondrá de acceso al resto.

Si se desea filtrar del exterior la utilización de estos programas puede hacerse mediante los puertos 512, 513 y 514. Los ficheros implicados en la obtención de permisos son:

`/etc/hosts.equiv`, `$HOME/.rhosts`, `/etc/hosts.allow` o `/etc/hosts.deny`, y `.shosts`.

Por ejemplo, si un atacante es capaz de obtener a través de un *exploit* uno de estos ficheros, es capaz de ver las puertas de entrada desde las que el acceso está permitido, por lo que su siguiente paso será adquirir el control de alguno de los sistemas contenidos en el fichero.

Existen asimismo ataques que se basan en modificar el fichero de confianza, por ejemplo el `.rhost` en `/usr/bin`, para poder acceder libremente desde el sistema actual.

5.40 Buffer-overflows

Los desbordamientos de un *buffer*, normalmente la pila de ejecución, se mencionan, junto a los *format strings*, como el último tipo de vulnerabilidad dentro de las asociadas a las comunicaciones por TCP/IP, ya que podría considerarse más una vulnerabilidad del sistema [BO-4] que de la red. Se ha incluido ya que está asociada a los servicios proporcionados sobre TCP/IP y porque es ejecutada mediante el envío de paquetes de información desde la red, explotando una debilidad.

La primera vulnerabilidad encontrada en Internet de este tipo fue el famoso *Gusano* de Robert Morris, Jr [BO-3]. El 2 de noviembre de 1988, éste estudiante generó un *exploit* que aprovechaba dos vulnerabilidades: la primera asociada al modo de depuración del demonio *sendmail* (que permite el envío de *e-mails*), y la segunda relativa al demonio *fingerd* (que implementa la identificación mediante peticiones *finger*) de los sistemas Unix. El gusano fue capaz de colapsar por aquel entonces gran parte de los sistemas existentes en Internet, provocando gran conmoción respecto a la seguridad en La Red.

Los servidores que proporcionan un servicio TCP/IP, a través de protocolos de nivel superior, como HTTP, SMTP, FTP, DNS, NNTP..., pueden presentar errores de diseño o implementación que dan lugar a vulnerabilidades de seguridad. Como objetivo final de éstas se busca la posibilidad de ejecutar código arbitrario, que consiste en proporcionar un *shellcode*, o código ensamblador que permite obtener una *shell* de *root* en el sistema, es decir, con todos los permisos de administrador.

Los SS.OO. abiertos suelen ser estudiados en mayor profundidad, por lo que las vulnerabilidades existentes son más conocidas, y por tanto, están más controladas (se habrá distribuido un parche software para corregir el error). El mejor ejemplo de un sistema así es Linux [BO-1]. En el caso de SS.OO. propietarios, como por ejemplo la familia Windows, la información al respecto depende del fabricante.

El ataque que permite explotar este tipo de vulnerabilidad se presenta normalmente en forma de *exploit*, que no es ni más ni menos que un programa escrito en C y en ensamblador que fuerza las condiciones necesarias para aprovecharse del error de seguridad subyacente. Existen infinidad de *exploits* para servicios como *imap*, *pop3*, *sendmail*, *inn*, *automountd*, *ftp*, *bind*, *httpd*, *samba*...

Sin duda alguna, uno de los mejores artículos existentes en la red es “*Smashing the Stack for Fun and Profit*” [BO-2] aparecido en Phrack [U-1].

Técnicamente, este método se basa en la posibilidad de escribir información sobrepasando los límites de un *array*, almacenado en la pila asociada a la rutina de un programa dónde el *array* está definido, consiguiendo así corromper la pila de ejecución, sobrescribiendo el valor retorno de la función, y causando que el flujo de ejecución continúe en una dirección arbitraria (introducida en los datos del *buffer-overflow*).

Esta técnica es posible al emplearse en los programas funciones de manipulación de *buffers* que no comprueban los límites de las estructuras de datos, como por ejemplo “`strcpy()`”, en lugar de las que sí lo hacen, por ejemplo “`strncpy()`” [BO-5].

Por ejemplo, en el siguiente programa, la situación genérica de la pila quedaría como sigue:

```
fuentes.c
-----
void funcion(int a, int b) {
    char buffer[100];
}

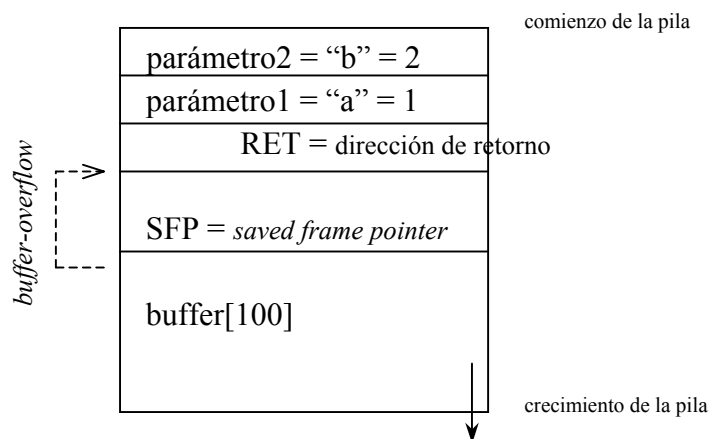
void main() {
    funcion(1,2);
}
-----
```

Situación de la pila:

Parámetros de la función:

Valores para el retorno de la llamada a la función: *call()*:

Variables locales de la función:



Para poder llevar a cabo esta técnica es necesario conocer información muy precisa de la arquitectura del sistema destino, tanto de la arquitectura de la CPU subyacente, como del sistema operativo sobre el que se provee el servicio a atacar. Por ejemplo, el sentido de crecimiento de la pila de ejecución, es decir, si es hacia direcciones menores o mayores de memoria, la definición del puntero de pila (SP), es decir, si éste referencia a la última posición ocupada en la pila o a la primera posición libre...

Asimismo, se requiere conocer en detalle el orden en el que se depositan los diferentes elementos en la pila: el puntero de *frame* anterior (SFP), la dirección de retorno (RET), las variables locales, los

parámetros que se le pasan a la función, el valor de retorno... Con toda esta información, se podrá introducir un valor en la posición de la dirección de retorno que envíe el flujo de ejecución justo al punto que se desee, es decir, una posición de memoria dónde se haya depositado previamente un *shellcode*; el objetivo suele ser disponer de un *shell* en el sistema con los permisos asociados al usuario con el que el servicio atacado (sobre el que se realiza el *buffer-overflow*) está ejecutando, que en numerosas ocasiones implica disponer de todos los permisos, al ejecutar como *root* (en Unix).

Una vez se disponga de toda la información detallada, se utilizará un puntero contra la dirección de memoria que contiene el valor de la dirección de retorno para modificarlo. El valor introducido apuntará a la dirección de la pila en la que se haya depositado el *shellcode*. Debido a que el código a ejecutar se deposita directamente en la pila, éste debe ser el propio código ensamblador extraído de su equivalente fuente. Para ello puede compilarse el fuente con la opción de generación del propio código ensamblador asociado o extraer éste mediante una utilidad como `gdb`. Por ejemplo, en lenguaje C un *shell* se obtendría mediante el siguiente código:

```
shellcode.c
-----
#include <stdio.h>

void main() {
    char *name[2];

    name[0] = "/bin/sh";
    name[1] = NULL;
    execve(name[0], name, NULL);
    exit(0);
}
-----
```

La ejecución real de *exploits* basados en este método requiere la utilización de técnicas más avanzadas, como la utilización de referencias relativas al *shellcode*, ya que se desconoce la posición absoluta en memoria, o lo que es lo mismo, el estado de la pila, en el momento de su ejecución.

Asimismo es necesario realizar un tratamiento al contenido del buffer a volcar en memoria, para evitar la existencia de elementos NULL que podrían concluir la lectura de un *string* de caracteres y por tanto no ejecutar el programa en su totalidad.

A su vez es necesario simular una condición de finalización correcta, tanto si realmente es así, como si ocurre algún problema en las llamadas al sistema, de forma que en ningún caso aborte la ejecución del servidor al intentar ejecutar el *exploit*.

Otro aspecto a tener en cuenta es la posible variación de la posición en la pila del código a introducir, por lo que es posible apuntar a direcciones de memoria no permitidas. Para evitarlo sería necesario averiguar el tamaño del *buffer* sobre el que aplicar el *overflow*, así como el desplazamiento necesario en la pila: para facilitar su situación en memoria suele emplearse el código NOP (*No Operation*), que le indica al procesador que no haga nada, de forma que puede utilizarse parte de memoria con este tipo de instrucción como margen en el desplazamiento a aplicar.

Resumiendo, los *buffers-overflows* se producen al escribir en memoria por encima de los límites de un *array* o *buffer* existente, y son consecuencia de que ciertas funciones, de lenguajes como C, implícitamente no comprueban los límites de los *arrays*. La librería estándar de C posee numerosas funciones para ejecutar operaciones de copia y concatenación de *strings*, sin comprobación de límites: `strcat()`, `strcpy()`, `sprintf()`, and `vsprintf()`. Todas operan considerando el final del *string* al observar el primer carácter NULL. Otras funciones afectadas podrían ser: `gets()`, `scanf()`, así como el uso de las funciones `getc()`, `fgetc()`, or `getchar()` en bucles de lectura de caracteres. Por tanto, el encontrar software vulnerable es tan sencillo como buscar en los fuentes por funciones como las comentadas y analizar su utilización.

Asimismo, existen generadores de *shellcodes* automáticos:

<http://www.deepzone.org/olservices/xploitit/index.htm>

5.41 *Format strings*

Los ataques de *format string* [FS-1] se producen al imprimir o copiar a otro *buffer* un *string*. El programador pretende imprimir su contenido con una sentencia como:

```
printf("%s", str);
```

pero en su lugar, por “vaguería”, escribe:

```
printf(str);
```

El resultado funcional es el mismo, pero el resultado técnico es muy diferente, ya que esta sentencia genera un agujero de seguridad en el código que permite controlar su flujo de ejecución. Aunque el programador le indica a la función el *string* a imprimir, ésta lo interpreta como un *string* de formato, es decir, pretende encontrar en su contenido caracteres de formato especiales, como por ejemplo “%d”, “%s”, “%x”. Por cada uno de los caracteres de formato, un número variable de argumentos será extraído de la pila. La vulnerabilidad obvia es que podrán leerse los valores de la pila que se encuentren por encima del *string* de formato, y la no tan obvia es que ofrece el control suficiente como para escribir en la memoria del programa.

Analizando la función *printf* sobre la que se desarrollará el ataque: aparte de las utilidades propias de la función para imprimir enteros, *strings* y delimitar la longitud de los campos a imprimir, ésta permite:

- [1] Obtener en cualquier momento el número de caracteres en la salida: al encontrarse un “%n”, el número de caracteres en la salida antes de encontrar éste campo se almacenará en la dirección pasada en el siguiente argumento:

```
int valor, x = 100, y = 20;  
printf("%d %n%d", x, &valor, y);
```

- [2] El carácter de formato “%n” devuelve el número de caracteres que deberían haberse emitido a la salida, y no el número de los que realmente se emitieron. Por ejemplo, al formatear un *string* en un *buffer* de tamaño fijo, el *string* podía ser truncado. A pesar de esto, el valor devuelto por “%n” será el desplazamiento si el *string* no se hubiera truncado:

```
char buf[20];  
int valor, x = 0;  
sprintf(buf, sizeof buf; "%.100d%n", x, &valor);  
printf("Valor: %d", valor);
```

Este ejemplo imprimirá “Valor: 100” y no “Valor: 20”.

Por tanto, mediante la manipulación correcta de funciones como `sprintf()` y `printf()` se pueden escribir caracteres en la pila, concretamente el número de *bytes* indicados por “%n” [1], en la dirección que se le indique a la función a través del *string* de formato (ya que ese valor se deposita en la pila, para que actúe como el siguiente argumento en [1]). El valor a escribir puede ser manejado como se desee por [2], ya que con “%.numerod”, se añade el valor “número” a los caracteres existentes realmente antes de “%n” en el *string*.

Al igual que en los *buffers-overflows*, es necesario conocer la arquitectura del sistema, por ejemplo, si la representación numérica es *little-endian* o *big-endian*, así como ciertas características del sistema operativo y del entorno.

La conclusión de esta técnica es que es posible escribir el valor deseado en casi cualquier dirección de memoria, luego puede usarse para sobrescribir el comando a ser ejecutado, o el UID asociado a un programa, o la dirección de retorno de forma que apunte a una posición de memoria donde se haya ubicado previamente un *shellcode* (tal y como ocurre en los *buffer-overflows* ;-).

5.42 Comunicaciones inalámbricas: *wireless*

Las redes inalámbricas sobre las que seguirá hablándose TCP/IP serán el objetivo de los *hackers* en un futuro no muy lejano. La facilidad de acceso a estas redes, en el caso de no encontrarse protegidas de forma adecuada, es mucho mayor que en el caso de las redes de datos comunes, ya que no es necesario el obtener un punto físico mediante el que conectarse a la red. Basta con disponer de una tarjeta de red que hable el protocolo 802.11b y de un portátil para desplazarnos al área de transmisión de una red concreta y comenzar a indagar [WI-1].

Existen herramientas que permite escanear el “ambiente” en busca de redes de este tipo, como por ejemplo, Net Sumbler [WI-2].

6 PROTECCIONES Y HERRAMIENTAS

Las protecciones especifican las técnicas y herramientas que se pueden implementar para la defensa frente a los ataques o vulnerabilidades presentados previamente. Saber cómo pueden atacar (y desde dónde), es tan importante como saber con qué soluciones se cuenta para prevenir, detectar y reparar un ataque de red.

No se debe olvidar que éstas siempre son una combinación de herramientas que tienen que ver con la tecnología, en la que se centra este estudio, y con los recursos humanos (políticas, formación, concienciación, capacitación, tanto de los usuarios como de los administradores de la red).

Los administradores de los sistemas disponen de herramientas para descubrir las vulnerabilidades existentes y para controlar que "todo vaya bien", si los procesos son los normales o si hay movimientos sospechosos en la red o en los sistemas. Por ejemplo, que un usuario esté recurriendo a vías de acceso para las cuales no está autorizado o que alguien intente ingresar en un sistema repetidas veces con claves erróneas que esté probando.

Debe tenerse en cuenta que muchas de las utilidades o técnicas presentadas son a su vez empleadas por los atacantes o *hackers*, pero es este motivo el que las hace realmente útiles desde el punto de vista de la protección, ya que uno de los métodos más efectivos a la hora de proteger el entorno de red se basa en la simulación de un ataque real de intrusión por parte de los administradores (*Penetration Test*). [PS-1]. De esta forma podrá saberse hasta donde puede llegar un intruso que posea ciertas herramientas disponibles libremente en Internet.

Cabe comentar que una de las mayores protecciones existentes hoy en día frente al descubrimiento de nuevas vulnerabilidades son los parches software de seguridad. Se trata de una distribución de software del fabricante de un equipo, ya sea de un servicio o de la propia pila TCP/IP del sistema operativo [TC-4], que sustituye los archivos binarios por una nueva implementación que resuelve una vulnerabilidad asociada al mismo (ver referencias URL de cada fabricante). Por tanto, muchos de los problemas de seguridad existentes actualmente se deben a que los administradores de los sistemas no están al corriente de los avisos asociados al descubrimiento de nuevas vulnerabilidades, que suelen ir acompañados de la información del parche que las resuelve, por lo que existen infinidad de equipos con la versión antigua en la que la vulnerabilidad puede ser reproducida.

Tras las protecciones se muestra una visión de futuro respecto a los protocolos asociados a la seguridad en TCP/IP, y a la evolución de ésta mediante la versión 6: IPv6.

Existen herramientas genéricas de análisis de seguridad. Por ejemplo, para los sistemas Unix, existe COPS [CO-1], una utilidad (o colección de programas) para comprobar la configuración del sistema frente a los posibles problemas de las diferentes áreas de seguridad que afectan a Unix. De iguales características es otra utilidad como TIGER [TI-1] y su nueva versión TARA [TI-2], *Tiger Analytical Research Assistant*.

6.1 Footprinting

La primera filosofía de protección aplicable a esta técnica, y extensible a la mayoría de vulnerabilidades, es aplicar la propia técnica sobre los sistemas a defender, para obtener la información que está disponible desde el exterior. Esta información debería controlarse lo más posible en base a las recomendaciones del RFC 2196 [FN-3].

Dentro de los sistemas IDS (ver apartado sobre éstos), se ha implementado la detección de muchas de las técnicas comentadas en la adquisición de información de una red. Una de las herramientas más populares es *Snort* [FN-7]. Es fundamental el deshabilitar los servicios que proporcionan información, como por ejemplo *fingerd*.

La forma directa de protegerse frente a adquisiciones de información de la topología de una red, así como del descubrimiento de los sistemas activos pasa por establecer filtros de paquetes para los protocolos UDP e ICMP. En el caso de DNS, la respuesta que se obtendría de un servidor seguro a la hora de listar los contenidos de un dominio sería similar a:

```
>
> ls -d yahoo.com.
[dnsserver.dominio.com]
*** Can't list domain yahoo.com.: Query refused
>
```

Frente a la identificación de las cadenas de texto representativas de un servicio y su versión, la técnica adecuada es investigar el servicio y el modo de modificar dicho texto, o incluso eliminarlo.

Asimismo, existe una herramienta denominada *RotoRouter* [FN-8] cuyo objetivo es defender una red frente a adquisiciones de información basadas en *traceroute*. Al detectar peticiones de este tipo, genera respuestas falsas con el objetivo de confundir al atacante.

En los *routers* Cisco puede controlarse el filtrado de las respuestas a paquetes *traceroute*, cuando se excede el TTL, mediante listas de acceso:

```
access-list 101 deny icmp any any 11 0
```

6.2 Fingerprinting

La modificación de los fuentes asociados al subsistema de red, es decir la propia implementación, variará el comportamiento habitual del sistema operativo. Asimismo, ciertos parámetros configurables a nivel de *kernel* o de módulo de red (`ndd` en Unix SystemV R4) permiten conseguir este objetivo.

Debido a que las técnicas activas presentadas se basan en enviar paquetes al sistema objetivo, puede ser sencillo para los IDS su detección [FN-23]. Sin embargo, las técnicas pasivas no son detectables, salvo mediante la utilización de herramientas de detección de *sniffers*.

Una herramienta realmente innovadora para ocultar el comportamiento característico de una implementación TCP/IP es *IP Personality* [FN-1].

El parche de *IP personality* implementado en Linux (*kernel 2.4*), añade la posibilidad de que la pila TCP/IP disponga de diferentes personalidades, es decir, que modifique las características de su tráfico de red, dependiendo de distintos parámetros. Para ello puede emplearse cualquier elemento que pueda ser especificado en una regla de *IPtables*: dirección IP fuente y destino, puerto TCP o UDP...

Por tanto pueden modificarse las siguientes características:

- Valor de los *Initial Sequence Numbers* (ISN) de TCP
- Tamaño inicial de ventana de TCP
- Opciones TCP: tanto su tipo como su orden en el paquete
- Las respuestas a ciertos paquetes TCP empleados en el *fingerprinting*
- Las respuestas a ciertos paquetes UDP

Como puede verse es altamente configurable. La implementación del parche se basa en la infraestructura de red *netfilter* creada por *Rusty Russel*. El objetivo de este sistema es defenderse ante el *fingerprinting*, ocultando el comportamiento típico del sistema en ejecución [FN-17]. Asimismo, permite emular el comportamiento de cualquiera de los sistemas contemplados en la base de datos de la utilidad NMAP.

Se ha comprobado que el parche es capaz de engañar a la versión 2.53 de NMAP, y debido a su extensibilidad en la configuración, puede adaptarse para otras herramientas.

Pese a ésto, la vieja regla de la seguridad “*security through obscurity*” no debería ser aplicada como primera medida defensiva, es decir, el que un atacante conozca el sistema operativo no debería facilitarle su acceso, ya que otras medidas se encargarán de limitarlo.

6.3 Escaneo de puertos-vulnerabilidades

Al igual que con la mayoría de los ataques cuyo objetivo es obtener información que permita identificar los sistemas y redes a atacar, la detección pasa por la utilización de sistemas IDS. Dado que NMAP [PT-1] es una de las herramientas de ataque por excelencia, SNORT [FN-7] supone la competencia por parte de la defensa.

Desde el punto de vista de un sistema Unix, existen utilidades como *scanlogd* [FN-12] que permiten la detección de los escáneres de puertos. Dicho ataque debe realizarse con cautela, ya que es muy sencillo determinar que se está produciendo, ya que bajo condiciones normales de funcionamiento no se realiza un acceso secuencial a todos los posibles puertos en un rango determinado[PT-3].

Asimismo, existen aplicaciones como *PortSentry* [PT-4] que pueden tanto detectar el ataque como responder a él, por ejemplo, modificando el módulo de filtrado del sistema atacado para desechar el tráfico proveniente del sistema atacante.

Por otro lado, los *firewalls* suelen incluir un módulo de detección de ataques, aunque no todos tienen la misma prioridad o validez. Por ejemplo, el escaneo basado en SYN puede ser detectado, pero no así el basado en paquetes de FIN. Para poder detectar estos ataques es necesario revisar el resultado de los *logs* de los *firewalls*, cuyo tamaño suele ser elevado, por lo que existen herramientas que facilitan estas tareas [PT-5]. También se han creado utilidades que ejecutan sobre productos de filtrado ya existentes, como la utilidad `alert.sh` [PT-6] preparada para *Firewall-1* de *Checkpoint*.

En el caso de Windows también se dispone de herramientas, como por ejemplo *BlackICE* [PT-7].

Desde un punto de vista más general, la protección frente al escaneo de puertos pasa por deshabilitar todos los servicios que no sean necesarios, en entornos Unix comentándolos en el fichero `/etc/inetd.conf`. A su vez, en éstos sistemas es posible no realizar su activación mediante los *scripts* de arranque, que por ejemplo en Linux se encuentran en `/etc/rc*.d` ó `/etc/rc.d/rc*`. En el caso de Windows la tarea debe realizarse deshabilitando los servicios bajo el *Panel de Control – Services*.

Desde el punto de vista del escaneo de las vulnerabilidades, la mejor defensa es disponer de las herramientas empleadas en una ataque para su descubrimiento y proceder de manera proactiva a solucionar o eliminar la existencia de la vulnerabilidad, ya sea eliminando el servicio o sustituyéndolo por una versión posterior corregida.

Existe disponible una lista generada por SANS y el FBI reflejando las vulnerabilidades más típicas encontradas en Internet:

- “How To Eliminate The Ten Most Critical Internet Security Threats” - junio 2001:
<http://www.sans.org/top10.htm>
- “The Twenty Most Critical Internet Security Vulnerabilities (Updated)” - octubre 2001:
<http://www.sans.org/top20.htm>

6.4 Escaneo basado en el protocolo ICMP

Para evitar las diferentes técnicas basadas en el protocolo ICMP, fundamentadas en los distintos tipos de paquetes existentes en la especificación del protocolo, lo mejor es filtrar todos aquellos tipos que no son necesarios para la funcionalidad asociada a los servicios existentes en la red.

6.5 *Sniffing, eavesdropping y snooping*

La protección básica frente a la extracción mediante *sniffers* de la información que viaja en los paquetes de datos por la red se basa en la encriptación de la información. A lo largo del texto se analizan diferentes protocolos de comunicaciones que hacen uso de ésta: SSL, S/MIME, SSH, IPsec...

Existen herramientas que aunque su uso principal es otro, ayudan a la detección de *sniffers*. *Tripwire*, o sus derivados, [SN-1] genera una huella mediante MD5 de los sistemas de ficheros de un *host*, y permite detectar cualquier modificación realizada sobre los ficheros o directorios. Aunque pudiera

parecer que se trata de una herramienta más propia de la seguridad de un sistema que de la red, permite la detección de ficheros de *log* generados por un *sniffer*.

Asimismo, existe una herramienta denominada CPM [SN-2], proporcionada por el CERT, que permite detectar interfaces en modo promiscuo (signo habitual de la existencia de un *sniffer*, aunque también pueden escuchar “sólo” el tráfico destinado a la máquina en la que residen).

La capacidad intrínseca de los *sniffers* se encuentra en el propio diseño de las redes de difusión, como por ejemplo *ethernet* (CSMA/CD). En las redes *ethernet* el tráfico viaja realmente por el mismo medio físico para todos los equipos conectados al mismo segmento de red (mismo *hub* o serie de *hubs* interconectados), siendo el destinatario de la trama *ethernet* el encargado de “retirarla” del medio y extraer su información. Los interfaces en modo promiscuo realmente actúan como si fueran el destino de todas las tramas, por lo que extraen la información de todas las comunicaciones.

Una protección frente a esta situación se basa en emplear redes conmutadas en lugar de compartidas, donde los elementos de red, en este caso los *switches*, segmentan la red para cada uno de los puertos, es decir, que si se dispone de un equipo por puerto (no existen *hubs*), un *sniffer* será capaz solo de visualizar el tráfico destinado al sistema en el que está ubicado.

Estos equipos son capaces de crear segmentaciones de la red por grupos de equipos, en base a una distribución lógica en lugar de física, denominadas redes locales virtuales, VLANs. En este caso, el *sniffer* solo podría visualizar el tráfico destinado a algún equipo de ese grupo o VLAN. Por último, debe considerarse que los *switches* disponen de la posibilidad de configurar un puerto de monitorización, mediante el cual se volvería a la situación anterior, en la que sería posible visualizar el tráfico de todas las VLANs a través de un punto único.

6.6 Encriptación: *SSL*, *PGP*, *S/MIME*

La protección de la información que circula por la red en claro, sólo puede ser protegida frente a las vulnerabilidades asociadas al *sniffing*, *eavesdropping* o *snooping* a través de técnicas de encriptación.

Se comenzará el análisis de esta técnica con el protocolo de seguridad *Secure Socket Layer* (*SSL*) [SL-1] [SL-2] [SL-3], por ser el más extendido actualmente, principalmente en las comunicaciones encriptadas de conexión a los servidores Web [SS-1] [SS-2], y por ser muy similar a otros protocolos de encriptación que se diferencian principalmente en el protocolo de aplicación hacia el que van enfocados, como por ejemplo *S/MIME* para la transmisión segura de *e-mails*.

Todos ellos se basan en la aplicación de algoritmos de encriptación asimétricos, mediante la utilización de una clave pública y una privada. Además, pueden verse potenciados por el establecimiento de una infraestructura PKI.

A grandes rasgos y desde un punto de vista práctico, los protocolos que funcionan bajo el mismo principio son:

- *SSL*: comunicaciones entre navegadores Web y servidores Web, o entre los propios servidores.

- PGP, *Pretty Good Privacy*: encriptación de correos electrónicos [PG-1] [PG-2].
- S/MIME: envío y recepción de mensajes electrónicos [SM-3] [SM-4].
- SSH: comunicaciones de establecimiento de sesiones (de tipo carácter y gráficas) así como transferencia de ficheros (ver apartado específico).
- IPSec: comunicaciones seguras entre dispositivos de red: clientes, *routers*, *firewalls*... Permiten el establecimiento de redes privadas virtuales (VPNs) (ver apartado específico).

La teoría alrededor de la encriptación es muy extensa, por lo que simplemente se mostrarán las nociones básicas, para analizar más en detalle el propio SSL. Antes de SSL existieron protocolos más específicos orientados a las transacciones Web, como S-HTTP.

Cuando se habla de encriptación debe siempre tenerse en cuenta los problemas de exportación impuestos por EEUU sobre todas las técnicas, algoritmos, software... relacionado con la misma. Por tanto, se habla de “la *inseguridad* de los programas de seguridad estadounidenses”, ya que el objetivo de este gobierno es no permitir el uso externo de algoritmos de encriptación lo suficientemente potentes (normalmente definido por el número de *bits* de la clave) como para impedirles a ellos el análisis de la información enviada. El máximo nivel permitido hasta hace algunos meses (64 bits en el algoritmo DES) perdió confiabilidad desde que se logró vulnerarlo.

El problema original a la hora de establecer una comunicación encriptada entre dos partes era el intercambio de las claves de una forma segura, ya que al utilizarse inicialmente sistemas de clave privada o simétricos, ambas partes debían compartir la misma clave. Una alternativa a este método, es el uso de sistemas de clave pública o asimétricos, que se basan en la existencia de dos claves, una pública y una privada, ambas complementarias.

Asimismo, dan lugar a la creación del concepto de certificado digital (X.509v3): información de encriptación (clave pública) asociada a un nombre o entidad específica, y firmada por un tercero fiable.

En el caso de la Web, los certificados digitales van asociados a una URL concreta, y contienen un número de serie, una fecha de expiración, varias extensiones criptográficas y la clave pública del servidor Web. La validez del certificado se obtiene al estar firmado por la clave privada de una Autoridad de Certificación (CA) reconocida oficialmente [SS-3]. Un certificado permite encriptar la información (confidencialidad o privacidad), firmarla (autenticación), y asegura la integridad de sus contenidos.

El protocolo SSL emplea esta técnica para encriptar la información transmitida entre el cliente y el servidor. Los pasos implicados en el establecimiento de una conexión SSL cuando sólo existen certificados digitales de servidor son:

1. El cliente (navegador Web) solicita al servidor su identificador (certificado digital).

2. El cliente confirma la validez del certificado, comprobando la URL y la fecha, y confirmando la firma de la CA. Para ello el cliente debe poseer la clave pública de la CA, y confiar en ella. El cliente extrae del certificado la clave pública del servidor.
3. El cliente genera una clave de sesión única, utilizada en exclusiva para este cliente y esta conexión, de ahí el concepto de sesión.
4. El cliente encripta la clave de sesión empleando la clave pública del servidor, y se la envía al servidor.
5. El servidor descripta la clave de sesión empleando su clave privada
6. La comunicación a partir de este momento entre el cliente y el servidor se encripta y descripta empleando un sistema de clave privada basado en la clave de sesión intercambiada.
7. El cliente utiliza ese canal de comunicación con el servidor Web para identificarse, normalmente mediante un usuario y una clave, proporcionados previamente por los gestores del servidor.

En el caso de que también existiera en la comunicación certificado digital de cliente, y no solo de servidor, la autenticación del cliente correspondiente al paso 1, sería similar a la autenticación de servidor explicada, pero en el otro sentido de la comunicación.

6.7 IDS: *Intrusion Detection Systems*

Además de los mencionados parches, una de las herramientas existentes hoy en día que proporciona un mayor control sobre la seguridad son los sistemas de detección de intrusos (IDS, *Intrusion Detection System*), también conocidos como NIDS, *Network IDS*, ya que pretenden contemplar dentro de sus comprobaciones todas y cada una de las vulnerabilidades que se van descubriendo a nivel de TCP/IP y de los servicios de red [ID-1]. Asimismo, ciertos IDS permiten la introducción de nuevos patrones (*signatures*), de forma que es posible ampliar la base de datos de vulnerabilidades en el momento en que aparecen, y no tener que esperar a que el fabricante distribuya una actualización. Por ejemplo, los patrones pueden contemplar situaciones como *source-routing*, la obtención del fichero de *passwords* en el contenido de una operación *get* de FTP, fragmentación de paquetes ICMP de gran tamaño...

La alternativa existente previamente a los sistemas IDS pasaba por realizar análisis exhaustivos de los *logs* del *firewall* o grupo de *firewalls*, tratando de interrelacionar los eventos existentes. El problema de este procedimiento es que en un sistema muy accedido, el tamaño de los *logs* es enorme: cientos de *Mbytes*.

Para asimilar la relación de eventos entre el método tradicional y los IDS, basta decir que un ataque de escaneo de puertos, en el *log* del *firewall* generaría unas 65000 entradas (64K puertos posibles) mientras que en el IDS se reflejará 1 sólo evento: “escaneo de puertos”. Lo mismo ocurriría en el caso de un *SYN Flood*, en el *log* aparecerían, por ejemplo, 10.000 conexiones SYN, mientras que en el IDS se vería 1 sólo evento: “ataque *SYN Flood*”.

Los IDSs suelen conformarse mediante un sistema de gestión centralizado y agentes o monitores remotos que se encargan de analizar el tráfico en los puntos remotos de la red en los que están

ubicados. La comunicación entre los agentes y el gestor no se realiza a través del protocolo SNMP como ocurre en los entornos de gestión de red, sino que la comunicación se establece de forma más segura, con métodos de autenticación y codificación. Por ejemplo en el Cisco IDS se realiza a través del protocolo conocido como PO, *Post Office*.

Asimismo, la seguridad de estos sistemas se incrementa al trabajar las interfaces por las que se realiza la monitorización en modo pasivo, es decir, no actúan como destino de ningún tráfico (no disponen de dirección IP), por lo que un atacante no puede detectar su existencia.

Los sistemas IDS, como por ejemplo NFR, *Network Flight Recorder*, *Scanlogd* [FN-12], *IPPL* [FN-13] o *Snort*, disponen de la detección de rastreos basados en *ping* (*ping sweep*). Asimismo, el control de este tipo de reconocimientos puede llevarse a cabo determinando los tipos de paquetes ICMP permitidos en cada segmento de red: para algunos segmentos será necesario permitir los tipos ECHO, REPLY, HOST UNREACHABLE y TIME EXCEEDED, pero no otros. A su vez, a través del control del tráfico en los *routers* de los bordes de la red se puede mitigar la obtención de información basada en ICMP, como la franja horaria o la máscara de subred empleada.

Los dispositivos Cisco disponen de un sistema IDS simplificado en el S.O. propio o IOS. Éste permite el control de diversas vulnerabilidades, disponiendo de la posibilidad de detectarlas y también de responder a ellas: generando una alarma, descartando el paquete o reseteando la conexión. En el IOS los patrones pueden tratarse con carácter informativo o como ataques.

Algunos sistemas más avanzados son capaces de modificar de forma dinámica las listas de control de acceso en los *routers* y *firewalls* en el momento de detectarse el ataque, con el objetivo de filtrar el tráfico asociado al mismo. Por ejemplo, *Cisco Secure IDS* emplea esta técnica junto a los *routers* Cisco y al *firewall* Cisco PIX.

Como regla fundamental debería definirse que los patrones de ataque deberían ser muy generales, de forma que si se produce una pequeña variación en los paquetes de información, por ejemplo la variación de un *flag* TCP, esta nueva modalidad de ataque también sea detectada. Pero, por otro lado, si es muy genérica, se activarán falsas alarmas de ataques cuando en realidad se trata de situaciones normales. Los IDS tienen una disyuntiva frente a la generalidad y particularidad de cada patrón identificativo o *signature*.

A continuación se presentan algunas de las herramientas existentes en el mercado, aunque se recomienda conocer el estado actual y la evolución de esta tecnología [ID-2]:

- Cisco: *Cisco Secure Intrusion Detection* (antiguamente *Cisco Netranger*)

<http://www.cisco.com/go/ids>

Se dispone del listado completo de los ataques que detecta este IDS en:

<http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids1/csidsug/sigs.htm>

- Anzen: *IDS appliances*

<http://www.anzen.com/>

- ISS (Internet Security Systems): *RealSecure*

<http://www.iss.net/>

http://www.iss.net/securing_e-business/security_products/intrusion_detection/

La descripción de las firmas analizadas por RealSecure pueden obtenerse en:

http://documents.iss.net/literature/RealSecure/RS_Signatures_6.0.pdf

- Demon: *Sessionwall-3*

<http://www.demon.net/products/security/sessionwall.shtml>

- Enterasys networks: *Dragon family products*

<http://www.enterasys.com/ids/>

- Network Security Solutions: herramientas de seguridad para Windows y Unix.

<http://www.ns2.co.uk>

- CyberSafe: *Centrax*

<http://www.cybersafe.com/>

- Axent: *NetProwler*

<http://www.axent.com/>

- Intrusion Detection Solutions:

<http://www.intrusion.com/>

6.8 IP spoofing

Mediante la configuración de filtros de conexiones permitidas, especificando a través de qué interfaces, en los dispositivos de interconexión de varias redes (*firewalls* y *screening routers*), se pueden controlar las direcciones IP fuente permitidas, y por tanto evitar la técnica de *IP spoofing*.

Los organismos que realmente pueden evitar el uso de ésta técnica son los propietarios de las redes troncales (*backbones*) de comunicación, normalmente *carriers* u operadores de telecomunicaciones (SP, *Service Providers*), ya que evitan desde su origen que un agresor pueda hacerse pasar por otro usuario. Al actuar de nexo de unión de las diferentes subredes, deben permitir únicamente que el tráfico saliente de cada red al troncal lleve asociada una dirección propia de la red desde la que sale. Así se controla el punto de inicio del *IP spoofing*.

- Linux:

El módulo IPCHAINS de Linux permite la protección frente a vulnerabilidades como por ejemplo el *IP spoofing*. A continuación se muestra el código que permite realizar la verificación de la dirección fuente, mediante un filtro llamado `rp_filter`. Basta con habilitarlo en el arranque de la máquina para que se encuentre activo en todo momento (en el futuro debería estar implementado por defecto):

```
#!/bin/bash
# This is the best method: turn on Source Address Verification and get
# spoof protection on all current and future interfaces.

if [ -e /proc/sys/net/ipv4/conf/all/rp_filter ]; then
    echo -n "Setting up IP spoofing protection..."
    for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
        echo 1 > $f
    done
    echo "done."
else
    echo PROBLEMS SETTING UP IP SPOOFING PROTECTION. BE WORRIED.
    echo "CONTROL-D will exit from this shell and continue system startup."
    echo
    # Start a single user shell on the console
    /sbin/sulogin $CONSOLE
fi
```

- Cisco:

Ver apartado de protección frente a *Smurf* para observar otras características de los equipos Cisco (*Unicast RPF*). La protección del *spoofing* no sólo afecta a la red propia, sino también al resto de Internet, por lo que se trata de una concienciación global.

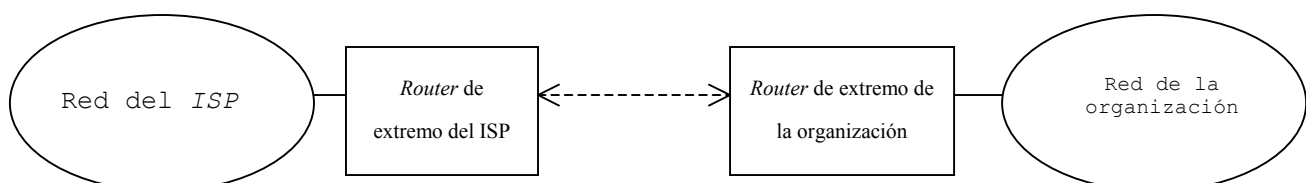
Una protección frente a esta técnica consiste en filtrar en el *router* las redes estándar internas definidas en el RFC 1918 mediante listas de control de acceso del IOS (ACLs):

```
interface X
    ip access-group 101 in

access-list 101 deny ip 10.0.0.0      0.255.255.255    any
access-list 101 deny ip 192.168.0.0  0.0.255.255     any
access-list 101 deny ip 172.16.0.0   0.15.255.255    any
access-list 101 permit ip any any
```

La idea se basa en permitir solo al tráfico destinado para la red interna entrar en ella (*ingress*) y solo al tráfico que se origina en ella (*egress*) salir. Esta regla es sencilla de implementar en los extremos de la red (*ISP, Enterprise*), pero no así en los núcleos centrales (*Service Providers*).

Cisco también recomienda la aplicación de filtros *ingress* y *egress* tal como se especifica en el RFC 2267. Los filtros en el IOS de Cisco se definen mediante ACLs. Los filtros *ingress* se refieren al tráfico destinado a una red (de entrada), mientras que los *egress* se refieren al tráfico de salida de la red (que se origina en ésta). En una configuración como la siguiente:



las recomendaciones serían:

- el *router* de extremo del proveedor, ISP, sólo deberían aceptar tráfico con dirección IP fuente perteneciente a la red de la organización (“REDorg”).
- la red de la organización sólo debería aceptar tráfico con dirección IP fuente perteneciente a una red DISTINTA a la de la organización.

Un ejemplo de ACLs en el *router* de extremo del ISP sería:

```
access-list 190 permit ip {REDorg} {mascara REDorg } any
access-list 190 deny ip any any [log]

interface {ingress interface} {interface #}
ip access-group 190 in
```

Un ejemplo de ACLs en el *router* de extremo de la organización sería:

```
access-list 187 deny ip {REDorg} {mascara REDorg} any
access-list 187 permit ip any any

access-list 188 permit ip {REDorg} {mascara REDorg} any
access-list 188 deny ip any any

interface {egress interface} {interface #}
ip access-group 187 in
ip access-group 188 out
```

En el caso de disponer de la posibilidad de activar CEF, (ver apartado de protección frente a *Smurf*), la longitud de las ACLs se reduciría, por lo que el rendimiento aumentaría.

Desde un punto de vista general debería filtrarse:

- Paquetes destinados a direcciones fuente privadas: RFC 1918 (ya comentado).
- Paquetes con fuente origen de la red de *loopback* (127.0.0.0). (1)
- Paquetes con dirección origen de la red propia.
- Paquetes con dirección origen *multicast*, sí éste no se usa 224.0.0.0). (2)
- Paquetes con dirección fuente de clase E: 240.0.0.0. (3)
- Paquetes con dirección de autoconfiguración DHCP: 169.254.0.0. (4)

Los equipos Cisco implementarían las reglas no comentadas previamente como sigue:

```
access-list 101 deny ip 127.0.0.0 0.255.255.255 any (1)
access-list 101 deny ip 224.0.0.0 15.255.255.255 any (2)
access-list 101 deny ip 240.0.0.0 15.255.255.255 any (3)
access-list 101 deny ip 169.254.0.0 0.0.255.255 any (4)
```

6.9 SMTP spamming

Los dispositivos de red permiten limitar el número de direcciones de correo en el campo destino. Por ejemplo, en el módulo IOS IDS de los equipos Cisco puede configurarse un límite de “N” usuarios mediante la siguiente sentencia:

```
ip audit smtp spam N
```

Asimismo, los servidores de correo, como *sendmail*, en sus últimas versiones han incluido reglas en el fichero de configuración para evitar esta técnica [SM-2]. Existe un servidor de correo, denominado *qmail* que fue diseñado con la seguridad en mente, por lo que es muy estable [SM-5].

6.10 DoS y DDoS

La defensa ante los ataques de denegación de servicio no es directa, ya que se basan en vulnerabilidades inherentes al diseño del protocolo TPC/IP. Pese a eso, existen herramientas de detección de intrusos [DD-8] que son capaces de reconocer las huellas dejadas por las herramientas de DDoS, por ejemplo al llevarse a cabo la comunicación desde el atacante a la hora de invocar la ejecución de ciertos comandos en remoto. De esta forma se podrá tener constancia de la infección antes de que se lleve a cabo el ataque masivo desde esos sistemas.

Podría ser posible el identificar a los atacantes mediante una combinación de técnicas de auditoria y referencias cruzadas de los ficheros de *log* [DD-7], junto a la escucha en los IRCs de *hacking*. Las penas impuestas por el FBI para este tipo de ilegalidad pueden ir de 5 a 10 años de cárcel [DD-3].

- *Net Flood*:

En este caso la red objetivo no puede hacer nada, ya que aunque se filtre el tráfico, las líneas estarán realmente saturadas con tráfico del atacante, por lo que estarán incapacitadas para cursar tráfico útil.

El método de actuación [DD-6] debería basarse en contactar con el operador que suministra el servicio de conexión a Internet para determinar la fuente del ataque, y de ser posible, para que lo filtre en su extremo de la conexión (*upstream*) de mayor capacidad. Una vez localizada la fuente, y siempre que no se haya combinado con la técnica de *IP spoofing*, será necesario informar a sus administradores de este hecho, ya que lo más probable es que no sean conscientes de que desde sus sistemas se está desencadenando este ataque. Esta comunicación no debería realizarse a través de e-mail, ya que el servidor de correo podría estar comprometido, sino mediante comunicación telefónica. El procedimiento debería basarse en el servicio WHOIS (también disponible vía WEB: http://www.nic.com/nic_info/whois.htm):

Se precisa buscar la información de contacto de “dominio.com”:

```
$ whois dominio.com
```

El resultado devolverá el *Registrar* (<http://www.internic.net> o <http://rs.internic.net>) que registró este dominio en el DNS, así como su servicio *whois* asociado, por ejemplo “whois.registrar1.com”:

```
$ whois dominio.com@whois.registrar1.com
```

De esta forma se podrá obtener la organización “global” bajo la que se registró el dominio [WI-1]: ICANN - IANA (*Internet Assigned Numbers Authority*): ARIN, RIPE, APNIC.

Asimismo, es posible buscar información sobre el propietario de la dirección de una red, como del dominio, de nuevo empleando el servicio WHOIS directamente, o a través de los servicios

disponibles en las páginas Web de los organismos encargados de los registros en los diferentes continentes [WI-1]. Por ejemplo, en Europa se encarga el RIPE:

```
$ whois 30.1.1.1@whois.ripe.net
```

En el caso de que se consulte la dirección IP de una red situada en un continente a un organismo que gestiona las direcciones de otro continente, éste devolverá a qué organización recurrir. Finalmente, el servicio de identificación indicará quién registró la red buscada, es decir, el propietario de la misma.

Debido a que los sistemas atacantes a su vez son víctimas, es necesario realizar un seguimiento complejo y lento, en el que deben colaborar los responsables de redes troncales, por lo que si el ataque dura solo un par de horas, puede ser imposible movilizar todos los recursos necesarios para determinar la fuente real del mismo.

(Ver apartado específico para la protección de este ataque en equipos Cisco).

- *Smurf*: (ver apartado específico).
- *TCP Syn Flood*:
Las versiones actuales de los sistemas operativos, tanto de los sistemas como de los dispositivos de red, contemplan parámetros de configuración para evitar este tipo de ataque (ver apartado específico).
- *Connection Flood*:
Mediante un sistema de detección eficiente y haciendo uso de filtros de paquetes es posible eliminar este tipo de ataque, siempre eso sí, que el número de equipos atacantes sea reducido, ya que es inviable añadir 1000 reglas al *firewall*.

Los ataques DDoS reflejan la necesidad de proteger todos los sistemas de una organización, aunque no contengan información valiosa, ya que pueden ser empleados como fuente de origen para la realización de ataques posteriores. Por ejemplo, para detectar este hecho debe tenerse en cuenta que un puerto habitual empleado para el control remoto de un sistema comprometido es el 37337.

La mejor defensa ante este tipo de ataques es seguir las recomendaciones y prácticas de seguridad genéricas: deshabilitar todos los servicios innecesarios, mantener el software actualizado, y suscribirse a las listas de correo de seguridad [DD-4].

Igualmente, es peligroso disponer de servicios tipo *proxy* (Squid, Socks, Proxies...) abiertos a todo el mundo en Internet, ya que pueden ser utilizados como red intermedia para atacar cualquier otro sistema. Lo lógico es ofrecérselos solo a los usuarios internos de la organización a la que pertenecen.

Las organizaciones suelen disponer de un enlace único a Internet, que será el objetivo del ataque para inutilizar todos los servicios de golpe: el extremo o punto de acceso de esa red. Por tanto, existen ciertas medidas [DD-5] para proteger los *routers* externos, precauciones en el filtrado (para evitar el

spoofing) y para proteger los mecanismos de envío y recepción de información de enrutado: BGP, OSPF...

También existen ciertos servicios básicos que serán el objetivo principal de un DoS, como puede ser el DNS, imposibilitando el acceso a ningún sistema mediante su nombre, o el LDAP y las PKIs, de forma que no se puedan llevar a cabo las comprobaciones de autenticación pertinentes.

6.11 Net Flood

Es posible controlar el ratio (número de paquetes por unidad de tiempo) de paquetes ICMP en los equipos Cisco mediante una característica denominada CAR. Por ejemplo:

```
interface X
rate-limit output access-group 2020 3000000 512000 786000 conform-action
transmit exceed-action drop

access-list 2020 permit icmp any any echo-reply
```

Para más información ver *IOS Essential Features* [NF-1].

6.12 Smurf

Para evitarlo [SM-1] basta con restringir el tráfico de *broadcast* desde fuera de la red hacia dentro, evitando así ser usados como multiplicadores de tráfico. Sólo tiene sentido permitir paquetes, por ejemplo ICMP con dirección destino la de *broadcast* dentro de la red, por ejemplo para que ciertos sistemas de gestión determinen que IPs hay activas, pero no desde el exterior, por lo que debe bloquearse en los *routers* o *firewalls* de los extremos. Éstos deben ser los principales encargados de no propagar los *directed broadcast*.

Las siguientes direcciones Web permiten comprobar si una red es vulnerable a este ataque basado en ICMPs, simplemente introduciendo la dirección de la red a chequear:

<http://netscan.org/index.html>

<http://www.powertech.no/smurf/>

En los equipos Cisco, este tipo de ataques puede controlarse mediante una característica denominada *Unicast RPF*. Esta funcionalidad también elimina otros ataques basados en *IP Spoofing*. Para ello es necesario configurar el siguiente comando de interfaz:

```
(Router-if)# ip verify unicast reverse-path
```

Esta característica comprueba cada paquete que se enruta hacia el encaminador; si la dirección IP fuente no posee una ruta en la tabla interna CEF (*Cisco Express Forwarding*), que apunte al mismo interfaz por el que llegó el paquete, éste es descartado. Por tanto, es una funcionalidad de entrada que actúa sobre los paquetes recibidos por el *router*. Para poder utilizar *Unicast RPF* es necesario disponer

en el *router* de *CEF switching* o *CEF distributed switching* habilitado de forma global, por lo que no afecta a otros modos de *switching* propios del interfaz o subinterfaz:

```
(Router)# ip cef
```

Por otro lado, puede ser controlado en las pilas TCP/IP de los sistemas, configurándose que no se responda a estos paquetes:

- HP-UX:

Los parámetros de red a modificar en el fichero */etc/rc.config.d/nddconf* son:

Para evitar el reenvío de *broadcast* directo

```
TRANSPORT_NAME[1]=ip
NDD_NAME[1]=ip_forward_directed_broadcasts
NDD_VALUE[1]=0
```

Para evitar que la máquina responda a paquetes *broadcast ICMP*

```
TRANSPORT_NAME[2]=ip
NDD_NAME[2]=ip_respond_to_echo_broadcast
NDD_VALUE[2]=0
```

- Solaris:

Para evitar el reenvío de *broadcast* directos se debe configurar el siguiente parámetro:

```
ndd -set /dev/ip ip_forward_directed_broadcasts 0
```

Para evitar que la máquina responda a paquetes *broadcast ICMP* se debe incluir la siguiente línea excepto en máquinas configuradas en entornos de alta disponibilidad:

```
ndd -set /dev/ip ip_respond_to_echo_broadcast 0
```

- Linux:

Puede controlarse mediante un parámetro de *kernel* en */proc/sys/net/ipv4* el ignorar los paquetes ICMP con dirección de *broadcast* o *multicast*:

```
icmp_echo_ignore_broadcast
```

En el caso de ser necesario también es posible deshabilitar la capacidad de responder al *ping* (*ICMP echo*):

```
icmp_echo_ignore_all
```

- Windows:

Los sistemas Windows no son vulnerables a este tipo de ataque, ya que siguen el RFC1122, y por tanto no responden a los ICMP *broadcast*: “*if we send an ICMP ECHO request to an IP Broadcast or IP Multicast addresses it may be silently discarded by a host*”.

6.13 TCP Syn Flood

Para el control de este ataque, en los equipos Cisco, el límite de ratio admitido para paquetes SYN puede definirse como sigue:

```
interface {int}
rate-limit output access-group 153 34000000 100000 100000 conform-action
    transmit exceed-action drop
rate-limit output access-group 152 1000000 100000 100000 conform-action
    transmit exceed-action drop

access-list 152 permit tcp any host eq www
access-list 153 permit tcp any host eq www established
```

El valor 34000000 representa el valor máximo del ancho de banda del enlace (34 Mbps), y el valor 1000000 se establece con un valor que esté comprendido entre un 50% y un 30% del ratio de *SYN Floods*. Asimismo, deben reemplazarse los ratios burst normal y burst max con valores precisos. Esta parametrización es fundamental para diferenciar situaciones de tráfico habituales de un ataque *SYN Flood* real: si el ratio de pico se fija a más del 30%, muchos SYN legítimos se descartarán. Para obtener el valor del ratio debe emplearse el comando “show interface rate-limit”, visualizándose los valores típicos y los excesos del *interface*. Por tanto, deben obtenerse los valores habituales en un funcionamiento normal (antes de que ocurra un ataque) y emplear éstos como límite.

Los equipos Cisco disponen de un sistema más avanzado de control de estos ataques conocido como *TCP Intercept*. Existen dos modos de funcionamiento:

1. *Watch*:

```
ip tcp intercept list <<ACL>>
ip tcp intercept mode watch
```

Se permite que las conexiones pasen a través del *router*, pero son analizadas (construyendo una tabla interna con las conexiones TCP existentes) hasta que se establecen. En el caso de que no se establezcan en 30 segundos (puede configurarse con “ip tcp intercept watch-timeout N”), el *router* envía un *reset* al servidor para liberar la conexión.

Si el número de conexiones es muy elevado, no se podrán resetear en un tiempo suficiente.

2. *Intercept*:

```
ip tcp intercept list <<ACL>>
ip tcp intercept mode intercept
```

El *router* de forma activa intercepta cada conexión, paquete SYN, y responde en representación del servidor con un SYN-ACK, esperando mientras tanto a la llegada del ACK. Cuando se recibe, el SYN original se envía al servidor y el *router* realiza el establecimientos en 3 pasos con el servidor. Cuando este proceso se completa, las dos conexiones (cliente-*router* y *router*-servidor) se enlazan entre sí. Para ello el *router* debe actuar continuamente adecuando los valores de los números de secuencia de ambos enlaces, situación que afecta al rendimiento y que no puede emplearse ni con NAT ni con *routing* asimétrico.

Asimismo, si el ataque se concentra en un único equipo, debería considerarse la posibilidad de instalar un filtro de paquetes software asociado a éste, como por ejemplo *IP Filter* [SF-1].

En el caso de HP-UX, el control de las conexiones admitidas se realiza en el módulo TCP/IP del *kernel*. Los sistemas Unix emplean dos colas diferenciadas para las conexiones entrantes para defenderse ante este tipo de ataques: una para gestionar las conexiones iniciadas (se ha recibido el SYN y se ha enviado el SYN-ACK), y otra para las conexiones establecidas (*three-way handshake*), cuyo socket está esperando una “accept()” por parte de la aplicación. El ampliar los valores permitirá el poder ofrecer el servicio únicamente ante ataques DoS reducidos. Para ello deben modificarse los dos siguientes parámetros en el fichero “*nddconf*”:

En servidores de red con un número alto de conexiones es necesario aumentar el número máximo de conexiones TCP (conexión establecida)

```
TRANSPORT_NAME[1]=tcp
NDD_NAME[1]=tcp_conn_request_max
NDD_VALUE[1]=1024
```


En servidores de red con un número alto de conexiones es necesario aumentar el tamaño de la cola de conexiones TCP incompletas (conexión iniciada)

```
TRANSPORT_NAME[2]=tcp
NDD_NAME[2]=tcp_syn_rcvd_max
NDD_VALUE[2]=4096
```

Estas dos características aumentan los requisitos de memoria de la máquina. Este cambio es conveniente en el caso de servidores de red, ya que los valores por defecto están pensados normalmente para una estación de trabajo, no para un servidor de red con gran número de conexiones.

Igualmente, en Solaris, en servidores de red con un número alto de conexiones es necesario aumentar el tamaño de la cola de conexiones TCP incompletas, para ello se debe incluir la siguiente línea:

```
ndd -set /dev/tcp tcp_conn_req_max_q0 4096
```

En servidores de red con un número alto de conexiones es necesario aumentar el número máximo de conexiones TCP, para ello se debe incluir la siguiente línea:

```
ndd -set /dev/tcp tcp_conn_req_max_q 1024
```

De igual forma, en Linux, *kernel* 2.2 (mediante el comando “sysctl”), existe un parámetro para controlar la cola de recepción de conexiones iniciadas:

```
/sbin/sysctl -w net.ipv4.tcp_max_syn_backlog=4096
```

Para cambiarlo en el *kernel* 2.4 bastaría con configurar el valor mediante el comando:

```
# echo 4096 > /proc/sys/net/ipv4/tcp_max_syn_backlog
```

Asimismo, en Linux, la gestión la puede también realizar un parámetro de *kernel* en `/proc/sys/net/ipv4`, que permite enviar *SYN cookies* [TC-5] cuando la cola de conexiones (*backlog*) de un *socket* es excedida (*overflow*): `tcp_syncookies/sbin/sysctl -w net.ipv4.tcp_syncookies=1`

En Windows se añadió esta característica, ver artículo [Q142641](#). Se recomienda consultar el artículo en “<http://support.microsoft.com>” para analizar los parámetros relacionados con el aquí mostrado. A través del editor del registro (`regedt32.exe`) debe localizarse la clave:
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`

Bajo la misma es necesario añadir el valor:

Value Name: SynAttackProtect

Data Type: REG_DWORD

Value: 0, 1, 2

0: sin protección frente a ataques SYN Flood.

1: los intentos de retransmisión utilizados se reducen y se introduce un retardo en la creación de entradas en la RCE (route cache entry) si están configurados `TcpMaxHalfOpen` y `TcpMaxHalfOpenRetried`.

2: igual que “1” y además se introduce un retardo en Winsock.

6.14 Connection Flood

Para evitar la existencia de un ataque basado en el cierre incorrecto de las conexiones, el sistema servidor puede controlar el tiempo que un *socket* TCP puede permanecer en el estado `TIME_WAIT`, evitando así un consumo de recursos excesivo.

- HP-UX y Solaris:

Para ello se emplea el siguiente comando limitando el tiempo a 60 segundos:

```
ndd -set /dev/tcp tcp_time_wait_interval 60000
```

- Linux (2.2):

Igualmente, se limita el tiempo de vida del *socket* en este estado:

```
/sbin/sysctl -w net.ipv4.vs.timeout_timewait 60000
```

- Linux (2.4):

En este caso, se limita el número de *sockets* en este estado. En el caso de que se supere el número, se destruirán *sockets* en ese estado, generándose un *warning*:

```
# echo 512 > /proc/sys/net/ipv4/tcp_max_tw_buckets
```

- Windows NT, 2000, XP:

A través del editor del registro (`regedt32.exe`) debe localizarse la clave:

`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`

Bajo la misma es necesario añadir el valor:

Value Name: `TcpTimedWaitDelay`

Data Type: REG_DWORD

Value: 30-300 segundos (defecto: 240 segundos)

6.15 Bastion hosts

La primera regla de oro a la hora de securizar un sistema (*hardening* o *armoring*) [BH-5], tanto Unix [UX-1] como Windows, pasa por deshabilitar todos los servicios TCP/IP innecesarios. A su vez, son múltiples los controles que pueden llevarse a cabo, tanto desde el punto de vista del sistema (no analizados en este documento centrado en TCP/IP) como de la red, en los que se profundiza en este documento.

Las recomendaciones de configuración respecto a la pila TCP/IP que se presentan a continuación no han sido incluidas en referencias específicas para la protección de una vulnerabilidad concreta comentada.

- HP-UX:

Existen dos documentos de referencia a la hora de securizar el sistema operativo HP-UX, para la versión 11.00 [BH-2] y para la 10.X [BH-3].

Concretamente, para ajustar la seguridad mediante los parámetros de red de TCP/IP se deben añadir las siguientes líneas al fichero */etc/rc.config.d/nddconf*.

```
# Para evitar que se propaguen paquetes de una interfaz a otra (uso de routing)
TRANSPORT_NAME[1]=ip
NDD_NAME[1]=ip_forwarding
NDD_VALUE[1]=0
#
# Para evitar el chequeo de gateway muerto (éste no funciona bien con firewalls)
TRANSPORT_NAME[2]=ip
NDD_NAME[2]=ip_ire_gw_probe
NDD_VALUE[2]=0
#
# Para evitar el uso de la estrategia PMTU discovery (echo-request).
TRANSPORT_NAME[3]=ip
NDD_NAME[3]=ip_pmtu_strategy
NDD_VALUE[3]=1
#
# Para no enviar ICMP source quench
TRANSPORT_NAME[4]=ip
NDD_NAME[4]=ip_send_source_quench
NDD_VALUE[4]=0
#
# Para evitar que se responda a peticiones Timestamp
TRANSPORT_NAME[5]=ip
NDD_NAME[5]=ip_respond_to_timestamp
NDD_VALUE[5]=0
#
# Para no enviar mensajes en paquetes de reset de TCP
TRANSPORT_NAME[6]=tcp
NDD_NAME[6]=tcp_text_in_resets
NDD_VALUE[6]=0
```

#

- Solaris: [BH-4]

Los parámetros de configuración generales recomendados en este S.O. son aplicables a un fichero de arranque del sistema, por ejemplo */etc/rc2.d/S69inet*. Para que todos estos cambios tengan efecto se deben ejecutar los siguientes comandos:

```
# /etc/rc2.d/S69inet stop
# /etc/rc2.d/S69inet start
```

Para evitar que la máquina encamine paquetes desde una interfaz a otra se debe incluir la siguiente línea:

```
ndd -set /dev/ip ip_forwarding 0
```

Para evitar que por cualquier interfaz de una máquina se reciban paquetes con destino a direcciones IP correspondientes a otros interfaces de la máquina, se debe incluir la línea:

```
ndd -set /dev/ip ip_strict_dst_multihoming 1
```

Para evitar que se responda a peticiones *Timestamp* se debe incluir la siguiente línea:

```
ndd -set /dev/ip ip_respond_to_timestamp 0
```

- Linux:

Al igual que los dos sistemas Unix previos, existen guías similares para Linux [BH-6] [BH-8]. Asimismo, existen versiones completas derivadas de Linux enfocadas desde el punto de vista de la seguridad, como Trinux (<http://www.trinux.org/>) u Openwall Owl (<http://www.openwall.com/Owl/>).

Guía para securizar un sistema Linux: “Linux Administrator's Security Guide” [LI-1].

Un proyecto que engloba a los anteriores es *GRSecurity*: <http://www.grsecurity.net>

- Windows:

Existen documentos de referencia [BH-1] [BH-7] para securizar este sistema operativo.

Chequeo del *gateway* activo o no:

A través del editor del registro (*regedt32.exe*) debe localizarse la clave:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
```

Bajo la misma es necesario añadir el valor:

Value Name: EnableDeadGWDetect

Data Type: REG_DWORD

Value: 0 (desactivarlo)

Las referencias respecto a los diferentes parámetros configurables de las pilas TCP/IP de los distintos sistemas operativos son múltiples. A modo de ejemplo se muestran las principales:

- “UNIX IP Stack Tuning Guide v2.7”:
<http://www.cymru.com/~robt/Docs/Articles/ip-stack-tuning.html>
- “Enabling High Performance Data Transfers on Hosts”:

http://www.psc.edu/networking/perf_tune.html

- Linux kernel 2.4:
<http://www.linux.org/docs/ldp/howto/Adv-Routing-HOWTO-14.html>
- Windows NT, 2000: <http://support.microsoft.com/>
“TCP/IP and NBT Configuration Parameters for Windows (Q120642)”
- Windows XP: <http://support.microsoft.com/>
“TCP/IP and NBT Configuration Parameters for Windows XP (Q314053)”
- “Solaris 2.X - Tuning Your TCP/IP Stack and more”:
<http://www.sean.de/Solaris/tune.html>

6.16 Bastion routers

Al igual que ocurre con los sistemas, los dispositivos de red deben ser configurados desde un punto de vista restrictivo, de forma que imposibiliten explotar la mayoría de vulnerabilidades comentadas. Para ello se configuran como un *bastion router* [BR-1].

Las características vulnerables de un *router* considerándolo como un *host* son:

- Existencia de claves en claro en la configuración.
- Servicios TCP y UDP simples activos: *echo*, *discard*, *daytime*...
- Protocolos de *routing* sin autenticación y/o encriptación.
- Protocolos AAA sin encriptación.
- Aceptación de paquetes *source routing*.
- Redirecciones IP.
- *Proxy ARP*.
- CDP, *Cisco Discovery Protocol*.
- Servidor *HTTP* activo.

A continuación se comentan algunos de los comandos que permiten securizar (*hardening* o *armoring*) desde el punto de vista de TCP/IP un *router* Cisco. Otros se han comentado en la sección asociada a la defensa de un ataque específico.

Deshabilitar los siguientes servicios:

- Finger:
`no service finger`
- Servicios simples: *echo*, *discard*, *daytime*.
`no service tcp-small-servers`
`no service udp-small-servers`
- HTTP server:
`no ip http server`
- Proxy ARP: en algunos escenarios, el uso de un *Proxy ARP* puede condicionar a que el tráfico circule por un camino que no es el impuesto por los protocolos de *routing*.
`no ip proxy-arp`
- Deshabilitar el Protocolo CDP:
`no cdp run`

6.17 Trinoo, Tribe Flood Network, TFN2K, Stacheldraht

Mediante la identificación de los puertos empleados por estas herramientas pueden evitarse sus consecuencias y detectarse su existencia. Otras herramientas genéricas de control de DDoS también las detectan [TF-3], aunque como norma general, la securización del sistema es la clave.

El deshabilitar el tráfico (típicamente ICMP) a través del que se comunican los componentes de la herramienta, también anula su ejecución. Para el caso de *Stacheldraht* se aplican las mismas consideraciones que anteriormente, centrándose también éstas en el filtrado del tráfico TCP asociado a la misma.

En el caso de *Trinoo* la protección puede realizarse ejecutando un *crack* de claves sobre el binario maestro, de forma que sea posible monitorizar la red en busca del valor en claro de la clave. Los paquetes con ese valor indicarán el comienzo de una conexión cuyo objetivo es enviar órdenes al demonio. Mediante este método podrá investigarse el origen del ataque y descubrir al atacante.

Existen numerosas utilidades para obtener las claves empleadas por las más conocidas aplicaciones de software: <http://members.aol.com/jpeschel/crack.htm>

6.18 NAT: Network Address Traslation

La técnica basada en la traducción de direcciones de red o NAT, *Network Address Traslation*, así como su variante que permite la traducción adicional de los puertos, PAT, *Port Address Traslation*, no se pueden considerar herramientas directamente relacionadas con la seguridad, pero aportan ciertas características de ocultación: “*security through obscurity*”.

Estas técnicas permiten no conocer desde el exterior las direcciones IP reales empleadas en la red interna, mostrando una visión particular y concreta de los sistemas y servicios visibles desde el exterior (direcciones IP y puertos). Por tanto, dificultarán la identificación y obtención de información de técnicas como *footprinting*.

6.19 Screening routers

Los *routers* de entrada a una red, ya sean propiedad del ISP, conocidos como CPE, *Customer Premise Equipment*, o de la organización, pueden ser empleados para realizar un filtrado inicial del tráfico IP, y se conocen como *screening routers*.

Típicamente se basan en tecnologías de filtrado de paquetes (ver los tipos de *firewalls* existentes), aunque en algunos casos se dispone de implementaciones con estado. Uno de los paradigmas en el filtrado de paquetes se basa en realizarlo lo antes que sea posible, para no desperdiciar capacidad y recursos en manejar paquetes que serán filtrados posteriormente.

La primera tecnología es empleada en los equipos Cisco a través de las ACLs del IOS: éstas son aplicables a IP, TCP, UDP e ICMP. Asimismo, Cisco dispone de una nueva implementación con muy alto rendimiento denominada *Cisco IOS Turbo ACLs*.

La segunda (filtros con estado) se implementa en Cisco a través de la funcionalidad conocida como *IOS Firewall*, previamente *IOS FFS*, *Firewall Feature Set*. Los equipos Cisco disponen de un motor de estado, denominado CBAC, que permite controlar el tráfico en base a la tecnología de *stateful filter* (ver apartado de *firewalls*). Esta tecnología contempla numerosas vulnerabilidades como *IP Spoofing*, fragmentación de paquetes, ataques DoS...

6.20 Ping of death

La solución generada por los fabricantes de las implementaciones se basó en la generación de modificaciones en la pila TCP/IP. En el caso de no disponer de un parche asociado al S.O., el ataque puede evitarse filtrando los paquetes ICMP en el *firewall* de entrada. Una solución aún más detallada es filtrar únicamente los paquetes ICMP fragmentados y no todos, ya que otros protocolos o procedimientos pueden emplear paquetes ICMP para otros propósitos.

6.21 Firewalls

Un *firewall* o cortafuegos [L-14] [L-15] es un sistema o grupo de sistemas que refuerzan la política de control de acceso entre dos redes, por tanto, cualquier mecanismo de control de acceso puede considerarse como tal, aunque normalmente el término se aplica a dispositivos especializados en esta función.

En toda implementación de un *firewall* debe balancearse entre la seguridad y la accesibilidad, siendo más o menos restrictivo en función de las necesidades. Un aspecto importante desde el punto de vista de la seguridad es que la comunicación con la consola de gestión del *firewall* debe viajar de forma encriptada.

Existen tres tipos de *firewalls*:

- Filtro de paquetes (*stateless*):

Fue la primer tecnología empleada en el control de acceso de las redes, y suele ser implementada mediante listas de control de acceso, que especifican el tráfico que puede pasar a través del filtro a través de los valores existentes en las cabeceras TCP, UDP e IP: direcciones IP y puertos UDP y TCP.

Suelen caracterizarse por una gestión muy compleja debido al elevado número de reglas que deben especificarse, ya que la orientación de su diseño se basa en el análisis de cada paquete individual que pasa por la red.

El control de las conexiones es sencillo, ya que por ejemplo, si se especifica una regla que permita tráfico de las conexiones *established*, por ejemplo en las ACLs del IOS de Cisco, sólo se comprobará que el paquete tiene activo el *bit* ACK.

Asimismo, no se adaptan a protocolos dinámicos como puede ser FTP, en los que los puertos por los que se realizarán ciertas comunicaciones se determinan en el transcurso de la conexión; no están predefinidos inicialmente.

Por otro lado, no pueden manejar de forma segura protocolos basados en UDP, ya que no se dispone en el momento de su configuración de los puertos UDP aleatorios elegidos por los clientes internos de la red, por lo que para habilitar la comunicación es necesario abrir todos los puertos mayores al 1023.

- Proxy de aplicación:

Un caso particular de este tipo de *firewalls* basados en *proxies*, y el más sencillo, es el de traductores de protocolos genéricos, como SOCKS. Éstos se encargan de reenviar las peticiones de red (mediante llamadas *socket*) haciendo de intermediarios, pero sin aplicar ningún filtro a los datos.

Por otro lado, los *proxies* de aplicación específicos, disponen de información propia de la aplicación, por lo que son capaces de aplicar restricciones muy particulares, como por ejemplo en el caso del protocolo HTTP no permitir la obtención de *applets* Java, o en el caso de FTP, permitir enviar ficheros (*put*) pero no recibirlos (*get*).

Debido a este conocimiento detallado de la aplicación se trata de sistemas más seguros, siempre que se disponga de un filtro específico asociado a la aplicación o protocolo que se desea filtrar. En caso contrario suponen una desventaja, ya que en el caso de nuevas aplicaciones no se dispondrá de éste, por lo que se deberá aplicar un *proxy* genérico.

Realizan un análisis a muy bajo nivel, tienen funcionalidades de contabilidad y auditoria muy detalladas, pero por el contrario el rendimiento es menor debido a su complejidad.

- Filtros con estado (*stateful filters*):

Los filtros con estado trabajan, por diseño, con las conexiones o sesiones, no con los paquetes individualmente, y son inteligentes desde el punto de vista de que controlan cada paquete asociado a cada conexión que transcurre a través de ellos, siendo conscientes de los pasos posibles a realizar por la aplicación o el protocolo.

La configuración es más sencilla, ya que debido al conocimiento intrínseco al *firewall*, con especificar el tráfico deseado, se configurarán automáticamente los caminos necesarios para el tráfico de retorno. Por ésto, es capaz de trabajar con protocolos dinámicos como FTP.

La seguridad que añaden y el rendimiento obtenido es muy positivo. Como desventaja es posible señalar la dificultad para analizar los contenidos de las aplicaciones, centrándose más en los protocolos, como TCP y UDP.

La aplicación y configuración de *firewalls* para Internet, incluyendo las características de los servicios a proteger puede estudiarse mediante [L-14]. Como reglas generales a aplicar pueden remarcarse la sencillez de la configuración y la aplicación de arquitecturas multicapa, de ahí la existencia de segmentos de red diferentes: interno, externo (típicamente Internet) y zonas intermedias, conocidas como DMZ, *DeMilitarized Zones*.

- *Firewalls* en Linux:

El S.O. Linux dispone de soporte de *firewalling*, denominado IPCHAINS en los *kernels* 2.2 [FW-1] e IPTABLES en los *kernel* 2.4. Mediante éste se pueden bloquear ciertos paquetes TCP/IP hostiles, y realizar un filtrado de paquetes avanzado. También permite la protección frente a otras vulnerabilidades, por ejemplo, *IP spoofing* (ver apartado de protecciones). Existen diferentes utilidades para una gestión más sencilla de los módulos [FW-2] [FW-3].

- Filtro IPF en Unix:

Existe una implementación de un filtro de paquetes con estado (*stateful*) de carácter *open source* para sistemas Unix abiertos BSD, que también está portada para HP-UX y Solaris [FW-7], denominada *Internet Packet Filtering*, *IPF*.

- *Firewalls* en Windows NT 4.0 y 2000:

Aunque NT dispone de un módulo interno de *firewall* no es 100% fiable, por lo que es recomendable no utilizarlo como la única medida de seguridad para el filtrado de paquetes.

Las capacidades de filtro de paquetes incluidas en Windows 2000 han mejorado enormemente las de Windows NT, y podrían considerarse adecuadas para sistemas de usuario final o servidores intermedios. La gestión se realiza a través de la MMC (*MS Management Console*) y puede realizarse de forma remota.

- *Firewalls* comerciales:

La variedad y diversidad de *firewalls* comerciales existentes hoy en día hace imposible un análisis exhaustivo de éstos y sus características. A continuación se presentan algunos orientados a redes corporativas de gran tamaño y con carácter crítico:

- Firewall 1: Checkpoint [FW-4] (*stateful*)
- Raptor y VelociRaptor: Axent- Symantec [FW-5] (*proxy*)
- StoneGate: StoneSoft [FW-6] (*stateful*)
- Cisco PIX [FW-8] (*stateful*)
- Gauntlet [FW-9] (*stateful* + *proxies*)

Existen dispositivos o *appliances* como los de Nokia, CyberGuard, SonicWALL o Cisco.

- *Firewalls* personales:

Los *firewalls* personales permiten securizar cada uno de los equipos que se conecta a la red de forma personalizada, evitando posibles vulnerabilidades existentes en *firewalls* más externos, corporativos, o en conexiones directas (no protegidas) a Internet:

- Network ICE BlackICE Defender: <http://www.networkice.com/> o <http://www.iss.com/>
- ZoneAlarm PRO: <http://www.zonelabs.com/>
- Norton Personal Firewall (NPF): <http://www.symantec.com/>

Existe una comparativa de todos ellos en:

<http://www.agnitum.com/products/outpost/compare.html>

6.22 Land

La primera protección frente a este tipo de ataque pasa por actualizar la pila TCP/IP del sistema operativo con los parches recomendados por el fabricante [LN-3].

La otra solución se centra en filtrar los paquetes IP *spoofeados*. Como es sabido, con la tecnología IP actual no es posible eliminar este tipo de paquetes, por lo que sólo pueden ser bloqueados. Esta técnica se describe en el RFC 2267.

6.23 Routing protocols

Los protocolos de enrutamiento pueden protegerse garantizando la autenticación del sistema de red que envía la actualización de la tabla de rutas así como encriptando los datos, con el objetivo de prevenir la inserción de actualizaciones falsas.

Los protocolos BGP, IS-IS y OSPF contemplan la posibilidad de autenticación e integridad en base al algoritmo de *hashing* MD5. El método empleado se basa en el conocimiento mutuo de una clave entre los dispositivos que intercambian rutas, enviándose cada actualización acompañada de un *fingerprint* o firma, obtenido a partir de la clave y el propio contenido de la actualización.

Como recomendación de seguridad conviene deshabilitar protocolos no seguros como RIP, y permitir otros más seguros como los arriba comentados. Asimismo, es conveniente deshabilitar los paquetes RIP de entrada en los *routers* externos, pudiendo emplear el protocolo internamente en la red.

Por ejemplo, para configurar esta funcionalidad en Cisco, para el protocolo BGP y OSPF:

```
router bgp as
neighbor 10.10.10.10 password SeCrEto

router ospf 10
area 50 authentication message-digest
interface ser 0/0
    ip ospf authentication message-digest
    ip ospf message-digest-key 1 key SeCrEto
```


6.24 Session hijacking

Los protocolos de encriptación, al igual que las técnicas de *sniffing*, como IPSec o SSH eliminan la posibilidad de obtener toda la información necesaria para reemplazar una sesión con otra.

6.25 Source routing

Las diferentes implementaciones disponen de mecanismos para deshabilitar ésta característica, de forma que no se acepten paquetes con *source routing*:

- Cisco:
Mediante el comando “no ip source-route” el *router* no admitirá paquetes con los datos de ruta implícitos.
- HP-UX:
Para ajustar este parámetro de red se deben añadir la siguiente línea al fichero `/etc/rc.config.d/nddconf`. Es importante si el sistema está enrutando, como por ejemplo un *firewall*:

```
# Para evitar que la máquina propague paquetes IP con la opción source_route
TRANSPORT_NAME[1]=ip
NDD_NAME[1]=ip_forward_src_routed
NDD_VALUE[1]=0
```

- Solaris:
Para ajustar el parámetro de red se deben añadir la siguiente línea al fichero `/etc/rc2.d/S69inet`:

Para evitar que la máquina propague paquetes IP con la opción *source_route*:

```
ndd -set /dev/ip ip_forward_src_routed 0
```

- Windows:
A través del editor del registro (`regedt32.exe`) debe localizarse la clave:
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`

Bajo la misma es necesario añadir el valor:

Value Name: DisableIPSourceRouting

Data Type: REG_DWORD

Value: 0, 1 or 2

Nota: 0 – permite habilitar *source routing*

1 - deshabilita *source routing* cuando *IP forwarding* está activo

2 - deshabilita *source routing* completamente (más seguro)

- Linux (*kernel 2.2*):
Se controla mediante un parámetro de *kernel* en `/proc/sys/net/ipv4` que determina si todos los paquetes de este tipo se aceptan o no: `accept_source_route`
`/sbin/sysctl -w net.ipv4.conf.all.accept_source_route=0`
Además puede controlarse sólo si se reenvían o no:
`/sbin/sysctl -w net.ipv4.conf.all.forwarding=0`

```
/sbin/sysctl -w net.ipv4.conf.all.mc_forwarding=0
```

- Linux (*kernel 2.4*):

Algunos son parámetros que deben configurarse por interfaz de red (<if>=eth0, eth1...) y no de forma global:

```
# echo 0 > /proc/sys/net/ipv4/conf/<if>/accept_source_route
# echo 0 > /proc/sys/net/ipv4/conf/<if>/forwarding
# echo 0 > /proc/sys/net/ipv4/conf/<if>/mc_forwarding
```

6.26 ICMP redirects

Las implementaciones TCP/IP disponen de mecanismos para controlar y deshabilitar la gestión de este tipo de paquetes, ya que mediante los mismos se podría modificar la tabla de rutas del *host* destino. En el caso de sistemas finales, no equipos de red, no es necesario ni el envío ni la recepción de estos paquetes:

- Cisco:
Mediante el comando “no ip redirects” se evita que los *routers* envíen estos paquetes. Por defecto los *routers* Cisco no escuchan estas notificaciones.

- HP-UX:

De nuevo en el fichero `/etc/rc.config.d/nddconf` puede configurarse este parámetro:

```
# Para evitar que se envíen paquetes ICMP de error de redirección
```

```
TRANSPORT_NAME[1]=ip
NDD_NAME[1]=ip_send_redirects
NDD_VALUE[1]=0
```

- Solaris:

Debe modificarse en la pila TCP/IP los siguientes parámetros:

Para ignorar los paquetes ICMP de error de redirección se debe incluir la siguiente línea:

```
ndd -set /dev/ip ip_ignore_redirect 1
```

Para evitar que se envíen paquetes ICMP de error de redirección se debe incluir la siguiente línea:

```
ndd -set /dev/ip ip_send_redirects 0
```

- Windows:

A través del editor del registro (`regedt32.exe`) debe localizarse la clave:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
```

Bajo la misma es necesario añadir el valor:

Value Name: EnableICMPRedirects

Data Type: REG_DWORD

Value: 0

NOTA: debe tenerse en cuenta que en algunas versiones de Windows el parámetro era en singular. *Ver fix:* Q293626.

- Linux:

En el *kernel* 2.4 se deshabilitan poniendo a cero los parámetros:

```
/proc/sys/net/ipv4/conf/<if>/accept_redirects  
/proc/sys/net/ipv4/conf/<if>/send_redirects
```

En el *kernel* 2.2 mediante los siguientes:

```
/sbin/sysctl -w net.ipv4.conf.all.send_redirects=0  
/sbin/sysctl -w net.ipv4.conf.all.accept_redirects=0
```

Asimismo, Linux dispone de un parámetro de *kernel* en `/proc/sys/net/ipv4` que permite controlar la aceptación de mensajes ICMP sólo de los *gateways* especificados en la lista de *gateways* por defecto:

```
secure_redirects
```

6.27 Directed broadcast

Las implementaciones TCP/IP disponen de mecanismos para controlar y deshabilitar la gestión y reenvío de este tipo de paquetes genéricos, normalmente de tipo ICMP (como en los ataques *Smurf*) dirigidos a la dirección de la subred:

- Cisco:
Mediante el comando “no ip directed-broadcast” se evita que los *routers* reenvíen estos paquetes, que dan lugar a ataques DoS, como *Smurf*. Por defecto los *routers* Cisco en versiones de IOS 12.0 o mayores no los envían.

Asimismo, suelen añadirse filtros para omitir el tráfico de redes amplificadoras que sí admiten estos paquetes IP. Existen en Internet listas de las redes activas de este tipo.

En el caso de paquetes de *broadcast* para la obtención de información, como los de máscara de red o *timestamp*, se pueden evitar empleando los siguientes comandos:

- HP-UX y Solaris:
Evita la respuesta a preguntas sobre la máscara de red:

```
ndd -set /dev/ip ip_respond_to_address_mask_broadcast 0
```


Evita responder a las preguntas de *timestamp*:

```
ndd -set /dev/ip ip_respond_to_timestamp_broadcast 0
```
- Windows:
Aunque de Nuevo el RFC1122 determina que no debería responderse a este tipo de paquetes, la implementación de cada fabricante actúa de forma particular. En el caso de Windows, cada versión tiene su propio comportamiento [IC-1].

6.28 SNMP

Como normal general, las redes de gestión deberían implementarse separadas de las redes de servicio (*out of band*). La primera recomendación concreta se basa en no emplear para los agentes SNMP las claves o nombres por defecto para los permisos de lectura (RO) y escritura (RW), respectivamente “*public*” y “*private*”.

Las implementaciones de los agentes SNMP disponen de mecanismos para modificar el valor de los nombres empleados por defecto.

- Cisco:
Los comandos que permiten cambiar los valores son:

```
snmp-server community CLAVE1 rw 10
snmp-server community CLAVE2 ro 11
```

No debería emplearse RW si la función del agente SNMP es sólo de monitorización.

- HP-UX:

El objetivo principal en los sistema Unix es configurar el servicio SNMP para que cumpla ciertas restricciones: sólo se permiten accesos de lectura, la comunidad SNMP debe considerarse como una clave de modo que se configurará a un valor difícil de adivinar, ésto es, distinta de *public* y *private*. Por último, el agente SNMP sólo permitirá accesos desde una lista determinada de máquinas gestoras.

Para conseguir este objetivo es necesario modificar el fichero `/etc/SnmpAgent.d/snmpd.conf` para incluir la configuración que se indica a continuación: se define la comunidad SNMP de lectura que corresponda; se permite sólo consultas desde las máquinas colectoras, se envían *traps* a éstas y la comunidad de escritura no se usa:

```
location: <Lugar donde está la máquina>
contact: <persona de contacto>
sys-descr: <descripción de la máquina>
get-community-name: <comunidad_lectura> IP: <máquina colectora>
#set-community-name: : <comunidad_escritura> IP: <máquina colectora>
trap-dest: <máquina colectora de traps>
```

En el caso de que sea necesaria la comunidad de escritura, se debe eliminar el comentario de la línea correspondiente.

- Solaris:

Para conseguir esta configuración, en base a las pautas comentadas en HP-UX, en el agente SNMP estándar de Solaris es necesario modificar varios ficheros tal como se indica a continuación. Una vez realizadas estas operaciones es necesario rearrancar el servicio SNMP. Para ello se debe ejecutar:

```
# /etc/rc3.d/S76snmpdx stop
# /etc/rc3.d/S76snmpdx start
```

El fichero `/etc/snmp/conf/snmpd.conf` especifica la configuración del subagente SNMP *mib1sa*. En este fichero se incluirá la configuración que se indica a continuación: se define la comunidad SNMP de lectura que corresponda; la comunidad de escritura no se usa y el único gestor permitido es *localhost*. Ésto es debido a que el subagente se comunica únicamente con el agente principal *snmpdx* en la propia máquina.

```
system-group-read-community <comunidad_lectura>
#system-group-write-community <comunidad_escritura>
#
read-community <comunidad_lectura>
#write-community <comunidad_escritura>
#
trap localhost
trap-community <comunidad_trap>
#
#kernel-file /vmunix
#
managers localhost
```

El fichero `/etc/snmp/conf/snmpdx.acl` especifica la lista de control de acceso para las ramas de la MIB soportadas por el agente principal *snmpdx*. Se incluirán las líneas de configuración siguientes. En la línea *managers* se incluirán nombres de máquinas separados por comas.

```
acl = {
{
communities = <comunidad_lectura>
access = read-only
managers = <IP_gestión_colectora>
}
}
```

Para aplicar una lista de control de acceso para el subagente SNMP *mibiisa* es necesario modificar el fichero `mibiisa.rsrc` y crear un nuevo fichero `mibiisa.acl`. Modificar el fichero `/etc/snmp/conf/mibiisa.rsrc` de modo que contenga las líneas siguientes:

```
resource =
{
{
security = "/etc/snmp/conf/mibiisa.acl"
registration_file = "/etc/snmp/conf/mibiisa.reg"
policy = "spawn"
type = "legacy"
command = "/usr/lib/snmp/mibiisa -r -p $PORT"
}
}
```

Crear un fichero `/etc/snmp/conf/mibiisa.acl` con permisos 600, propietario *root* y grupo *sys* y el contenido siguiente:

```
# Fichero de control de acceso para mibiisa
#
acl = {
{
communities = <comunidad_lectura>
access = read-only
managers = <IP_gestión_colectora>
}
}
trap = {
}
```

Dónde en general, *<comunidad_lectura>* es el nombre de la comunidad que siendo distinta de *public*, deberá ser tratado como si de una clave se tratara, *<comunidad_escritura>* es el nombre de la comunidad que siendo distinta de *private*, deberá tener tratamiento de clave, *<comunidad_trap>* es el nombre de la comunidad de *trap*, puede utilizarse el mismo que para la comunidad de lectura. Asimismo, *<IP_gestión_colectora>* es la dirección IP de la máquina o máquinas de gestión encargada de recoger todos los eventos producidos.

Asimismo, debería filtrarse las fuentes desde las que se admiten conexiones SNMP. Por ejemplo mediante ACLs en Cisco:

```
access-list 101 deny udp any any eq 161
```

Por último recomendar la implantación de la versión 3, preferiblemente, del protocolo de cara a la seguridad, y a evitar el envío en claro de los *community strings* por la red, u optar por alternativas de encriptación como incluir el tráfico SNMP en su propia VPN.

6.29 TCP Initial Sequence Numbers

Los sistemas pueden protegerse de este ataque mediante la encriptación del protocolo, con IPSec, o siguiendo las recomendaciones del RFC 1948.

En HP-UX, versiones 11 y 11i, debe aplicarse el parche de red PHNE_22397: por defecto emplea ISN aleatorios. En versiones anteriores, por ejemplo 10.20, existe un parámetro para elegir el nivel de aleatoriedad en la creación de los ISNs. Se introdujo con el parche PHNE_5361. Mediante la utilidad “*net tune*” es posible modificarlo:

```
tcp_random_seq set to 0    (Standard TCP sequencing)
tcp_random_seq set to 1    (Random TCP sequencing)
tcp_random_seq set to 2    (Increased Random TCP sequencing)
```

En la implementación TCP de Solaris, para cambiar el método utilizado en la generación de números de secuencia de paquetes TCP (ISNs) y evitar la suplantación de la dirección IP (*IP Spoofing*) se debe modificar el fichero */etc/default/inetinit* para que indique lo siguiente:

```
set TCP_STRONG_ISS=2
```

En el caso de Windows NT, hasta la versión 4.0 SP6 no se ha implementado un generador realmente aleatorio de ISNs en la pila TCP.

Existen sistemas como el Cisco PIX que permiten aleatorizar los ISNs [IS-5] de los servidores que protege, luego en el caso de que el algoritmo de generación de éstos no sea todo lo bueno que debería, se soluciona la situación.

El *kernel* de Linux implementó una variante del RFC1948 por defecto desde 1996 [IS-4] [IS-6].

6.30 *Tiny Fragment Attack*

Los sistemas de filtrado deben asegurar un tamaño mínimo de fragmento, ya que sino, el paquete no será descartado en el filtro. El RFC 791 indica: “todo módulo de Internet debe ser capaz de reenviar un datagrama de 68 *bytes* sin fragmentación posterior. Basado en que la cabecera puede ser de 60 *bytes* y el fragmento mínimo es de 8 *bytes*.” Desde el punto de vista de la seguridad, estos 8 *bytes* no aseguran la inclusión de todos los campos de la cabecera TCP.

Este ataque puede prevenirse restringiendo los límites de los fragmentos que atraviesan un filtro, de forma que contengan la información suficiente. En el caso de Cisco, se controla que el tamaño de fragmento sea mayor de 40 *bytes*.

Existen dos formas de afrontarlo:

- Método directo:
Existe un valor TMIN que indica la longitud mínima de la cabecera TCP requerida para contener toda la información de transporte relevante, desde el punto de vista de los filtros de paquetes. La medida se toma desde el comienzo de la cabecera TCP en el paquete original (sin fragmentar). El control se basa en analizar los paquetes con un *offset* de cero frente a este valor, para no permitir paquetes con un valor de TMIN menor.
- Método indirecto:
Este se basa en el análisis de un paquete TCP, de forma que cuando es fragmentado, si los campos que definen los *flags* no se encuentran en el paquete inicial, éste se deja pasar, pero al recibirse el siguiente fragmento, en base al campo FO, *Fragment Offset*, se descarta, con lo cual se bloquea el proceso de reconstrucción del paquete original.

6.31 *Winnuke*

La protección frente al *Winnuke* pasa por actualizar la pila de TCP/IP de Windows [WN-2] [WN-7]. Asimismo se dispone de software cuyo objetivo es controlar la llegada de éste ataque [WN-3]. Lo que hace básicamente es monitorizar el puerto 139 y mostrar la dirección IP del atacante.

6.32 *Teardrop*

Microsoft generó un *fix* para solucionar el problema modificando el comportamiento de la pila TCP/IP [TD-6]. En el caso de *teardrop2* también se dispone de otro *fix* [TD-8].

6.33 *DNS*

Las protecciones de seguridad del servicio de nombres [BN-2] permiten controlar desde el fichero de configuración el comportamiento del servidor frente a los supuestos clientes y servidores, tanto secundarios como los asociados a otros dominios (ver apartado de vulnerabilidades específicas).

Una de las medidas más sencillas para controlar las transferencias de zona desde el punto de vista del *firewall* en lugar de desde la configuración del servicio, es permitir conexiones externas al DNS a través del puerto 53 solo para el protocolo UDP, asociado a las consultas individuales, y no a través de TCP, correspondiente a las transferencias de zona.

Existen herramientas, principalmente IDSs, que son capaces de detectar las peticiones efectuadas por DIG y avisar de éstas. Incluso algunas facilitan el explotar DIG a través de un *buffer-overflow*.

Mediante la configuración del propio DNS se pueden restringir las transferencias de zona sólo a ciertos DNS esclavos, por ejemplo 192.168.20.20, y las peticiones de información de nombres a los clientes de la red, por ejemplo 192.168.100.0:

```
zone "dominio.com" {
    type master;
    file "dir/db.dominio.com";
    allow-transfer { 192.168.20.20; };
    allow-query { 192.168.100.0/24; };
}
```

Otra forma de especificar conjuntos de sistemas o redes es mediante ACLs. Por ejemplo:

```
acl "red_interna" {
    {192.168.100.0/24;      192.168.200.0/24; };
}
```

Pudiéndose emplear entonces la nueva ACL en lugar de las redes numéricas:

```
allow-query { "red-interna"; };
```

Además, es posible denegar la petición de información relativa a la versión de BIND que se está ejecutando, excepto para el propio servidor, ya que esta información permitiría a un atacante buscar las vulnerabilidades que afectan a la misma:

```
zone "bind" chaos {
    type master;
    file "master/bind";
    allow-query { localhost; };
}
```

El fichero `“./master/bind”` debería contener:

```
$TTL 1d
@      CHAOS SOA      localhost.  root.localhost. (
    1      ; serial
    3H     ; refresh
    15M    ; retry
    1W     ; expire
    1D )   ; minimum
NS     localhost
```

Pueden restringirse los interfaces del sistema por los que escuchará el proceso de DNS: `named`

```
listen-on { 192.168.20.1; };
```


Una mayor protección frente a posibles ataques de *buffer-overflow* puede llevarse a cabo ejecutando `named` con un usuario normal y no como *root*. Los ficheros de BIND deben pertenecer a este usuario. Una vez arrancado el servicio, si se dispone de un usuario *bind* del grupo *dns*, se puede modificar el *script* de arranque “`/etc/init.d/named`” (Linux-Debian) o “`/etc/rc.d/init.d/named`” (Linux-Red Hat):

```
/usr/sbin/named -u bind -g dns
```

Asimismo, puede securizarse más el entorno configurando el directorio raíz del servicio a su directorio propio, mediante `chroot`.

6.34 NTP

Es posible restringir los sistemas desde los que admitimos modificaciones en la hora de un equipo mediante el fichero de configuración del servicio NTP, denominado `/etc/ntp.conf` en Unix:

```
# Servidores desde los cuales se aceptan paquetes NTP
# Permitir acceso al servidor o servidores
restrict <dirección_ip_del_servidor_NTP> mask <mascara_servidor_NTP>

# Permitir acceso a localhost
restrict 127.0.0.1

# Denegar cualquier otro acceso
restrict default notrust nomodify
```

6.35 Caballos de Troya o Trojanos

Las medidas de control asociadas a este software se basan en un análisis exhaustivo de los puertos empleados para la comunicación exterior [TJ-5], así como en detectar huellas que generan en los sistemas de ficheros, registros, en el caso de Windows, o ficheros de configuración. Respecto al software de control remoto, la protección de las *passwords* es la medida primordial a considerar.

Debido a que cada herramienta presenta sus particularidades, debe analizarse en profundidad el objetivo y las acciones que lleva a cabo: [TJ-1] [TJ-2] [TJ-3] [TJ-4]. Por ejemplo, en el caso de *BackOrifice*, se dispone de numerosas herramientas para eliminarlo *BackOfficer Friendly* [TJ-6], *Back Orifice Eliminator*, *Back Orifice Eradicator v1.0*.

6.36 IPSec

El estándar para el protocolo IPSec [IP-4] [IP-5] según el IETF [IP-1] se contempla en 3 RFCs más uno adicional [FW-3] que describe el protocolo de intercambio y gestión de claves (IKE, *Internet Key Exchange*) empleado por IPSec para obtener una clave maestra para la autenticación. IPSec proporciona encriptación a nivel de la capa de red, es decir, IP.

Éste es uno de los protocolos de seguridad sobre los que se está desarrollando más actualmente [IP-2], aunque fue diseñado inicialmente para IPv6, portándose posteriormente a IPv4. Los fabricantes de equipos de red, por ejemplo, Cisco [IP-6] o Checkpoint, proporcionan sistemas para el establecimiento

de redes privadas virtuales o VPN, *Virtual Private Networks*, mediante IPSec, tanto software cliente como servidor

En IPSec, mediante el uso de *fingerprints* en los paquetes IP, generados a través de una función *hash* basada en el uso de una clave compartida por ambos extremos de la comunicación, se asegura la identificación (autenticación) del emisor, por lo que permite erradicar la técnica de *IP Spoofing*.

Asimismo soluciona otros problemas de seguridad, además de la autenticación, como los ataques DoS, por ejemplo *SYN Floods*, y los problemas de confidencialidad o privacidad basándose en la encriptación, y la integridad de los datos intercambiados mediante funciones *hash*.

Por otro lado, todos los protocolos de encriptación requieren una utilización mucho mayor de CPU, para ejecutar los algoritmos asociados a estos procesos, por lo que facilitan la ejecución de ataques DoS. La utilización de tarjetas específicas de encriptación libera a la CPU general de este proceso, pero simplemente desplaza el punto débil asociado al consumo de procesamiento a otro componente. Este hecho ha dado lugar a que no se empleen en IPSec firmas digitales completas.

IPSec dispone de dos servicios:

- AH: *Authentication Header* – asegura la autenticación y la integridad. RFC 2402.
- ESP: *Encapsulating Security Payload* – asegura confidencialidad, autenticación e integridad. RFC 2406.

Los algoritmos de encriptación principales que se implementan son SHA-1, MD-5, DES y 3DES, así como otros adicionales: IDEA, Blowfish y RC4.

Ambos servicios pueden trabajar en modo transporte, cuando los sistemas finales de la comunicación manejan IPSec, o en modo túnel, cuando son los equipos intermedios de entrada a la red (*routers*, *firewalls*...) los que gestionan el tráfico IPSec.

Las relaciones entre dos equipos que desean hablar mediante el protocolo IPSec vienen caracterizadas por el conjunto de parámetros que define todos los aspectos a tener en cuenta, como los algoritmos de autenticación y encriptación, la longitud de las claves, el servicio (AH o ESP) a emplear... Este conjunto de parámetros se conoce como SA, *Security Association*.

Dentro de las tecnologías empleadas para el establecimiento de VPNs, además de IPSec existen protocolos de creación de túneles, como L2TP, *Layer 2 Tunneling Protocol*, o PPTP, *Point-to-Point Tunneling Protocol*. La especificación de los mismos está disponible en el IETF [VP-1]. Asimismo, existen especificaciones de securización para éstos [VP-6].

Un túnel determina la encapsulación de un protocolo en otro, ya sea de forma encriptada o no:

- IPsec: IP en IP.
- PPTP: PPP en GRE.

Frente a ciertas vulnerabilidades encontradas en el protocolo PPTP [VP-4], Microsoft distribuyó los parches de seguridad asociados a su resolución [VP-5].

6.37 Finger Bomb

La forma de protegerse ante este ataque es deshabilitar el servicio de *finger*, o disponer de una versión del mismo no vulnerable.

6.38 TCP wrappers

Los *TCP wrappers* (`tcpd` o `tcpwrapper`) [TP-1] [TP-2] [TP-3] son una utilidad 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*), el demonio que implementa los servicios TCP/IP en Unix.

Con esta herramienta el administrador posee un mayor control de las conexiones que entran y salen hacia y desde una máquina, facilitando el filtrado de servicios, ya sea anulándolos completamente o controlando desde dónde se permite su acceso. La función de registro permite informar al administrador en todo momento y con todo lujo de detalles de las conexiones asociadas a los servicios de red que se han hecho desde y hacia el sistema.

Algunos de los ficheros afectados en Unix son:

- `/etc/inetd.conf`: definición de los servicios TCP/IP activos. Especifica el programa que debe emplearse para dar el servicio. En el caso de emplear un *wrapper*, será éste el invocado en lugar del servidor original. Posteriormente las peticiones recibidas pasarán del *wrapper* al servidor. Los servicios no deseados deben comentarse mediante el uso de una “#” al comienzo de la línea.
- `/etc/services`: definición de los servicios TCP/IP. Su carácter es principalmente informativo, ya que asocia al nombre del servicio, el protocolo (UDP o TCP) y el puerto empleado.
- `/etc/host.allow` y `/etc/host.deny`: especificación de los sistemas desde los que se permite y deniega el acceso.

6.39 MPLS

El protocolo MPLS, *Multiprotocol Label Switching*, permite crear redes privadas virtuales en las redes IP tradicionales. Este tipo de redes no encriptan los datos, a diferencia de las creadas empleando IPSec.

Ambas tecnologías pueden ser empleadas de forma conjunta añadiendo un nivel de seguridad adicional [MP-1].

6.40 SSH: *Secure Shell*

El diseño de SSH surgió en una Universidad de Finlandia, y permite encriptar las comunicaciones de otros protocolos TCP/IP. Su uso principal se centra en las comunicaciones que permiten establecer sesiones y realizar transferencia de ficheros, es decir, telnet, ftp, los comandos-*r* (*rsh*, *rlogin*, *rcp*...) así como sesiones X-Windows. Actualmente están implementadas dos versiones (versión 1 y 2) [SS-4] [SS-5] [SS-6]. Se dispone de desarrollos tanto para la plataforma Unix como para Windows.

SSH es una *suite* de encriptación que pretende sustituir los programas de conexión mencionados previamente. Mediante el uso de claves asimétricas, permite realizar la encriptación y la autenticación entre dos sistemas, por lo que evita ataques “*man-in-the-middle*” (*session hijacking*) y *DNS spoofing*. Asimismo aplica ciertos algoritmos de compresión a los datos transmitidos.

Para generar las claves asociadas a un sistema, tanto la privada, gestionada localmente, como la pública, que deberá compartirse con el resto de sistemas involucrados en la comunicación SSH, se dispone del comando `ssh-keygen`. Durante su ejecución debe especificarse el fichero en el que almacenar la clave privada (`$HOME/.ssh/identity`) así como la *passphrase* o clave asociada a la misma. A su vez, se generará un fichero con la clave pública (`$HOME/.ssh/identity.pub`).

Es necesario por tanto distribuir la clave pública a los sistemas remotos. Para ello basta con incluir el contenido de “`identity.pub`” en el fichero remoto “`$HOME/.ssh/authorized_keys`”.

Una vez se hayan distribuido las claves entre los sistemas manualmente o en base a una infraestructura PKI, se podrá acceder a ellos (`ssh`) o transferir ficheros (`scp`). En ambos casos, al emplearse la clave privada propia se solicitará al usuario la *passphrase*. Ejemplo de uso:

```
$ ssh sistema_remoto
$ scp fichero_local sistema_remoto:/tmp
```

6.41 Programación segura: *Buffer Overflows, Format Strings*

La programación de aplicaciones de forma segura, para evitar vulnerabilidades que permitan la ejecución de código por parte de un atacante, es un área de la seguridad que prácticamente no ha evolucionado desde hace 13 años, cuando en 1988 el gusano de Internet que colapsó la red mediante una vulnerabilidad del demonio “*fingerd*” explotó la posibilidad de ejecutar código en el sistema.

Actualmente, en el año 2001, se siguen sufriendo ataques basado en que no se realiza la comprobación de estructuras de memoria, permitiendo introducir código no deseado en la misma, como por ejemplo, gusanos como el Code Red sobre los servidores Web IIS de Microsoft.

Existen diferentes proyectos con el objetivo de concienciar a los programadores de la necesidad de comprobar los límites de las estructuras de datos en memoria y de la escritura sobre éstas [PS-2].

6.42 PKI: *Public Key Infrastructure*

La simple generación e instalación de los certificados en los navegadores Web y en el servidor Web, como se explica en el protocolo SSL no es suficiente para confiar en la seguridad y creer que no existen riesgos desde el punto de vista de la seguridad. Tomando de nuevo como ejemplo las comunicaciones Web, pero teniendo en cuenta que esta técnica es aplicable a cualquier protocolo en el que se utilicen certificados digitales, es necesario considerar el procedimiento necesario para comprobar la validez de un certificado (o clave pública) así como para realizar la gestión completa de los mismos: generación, validación, revocación... Toda la gestión global de una arquitectura basada en un sistema de clave pública requiere el establecimiento de una infraestructura PKI, *Public-Key Infrastructure*.

La tecnología e implantación asociada a una PKI es realmente compleja, por lo que se remite al lector a consultar las referencias asociadas a su estudio, así como las soluciones ofrecidas por los diferentes fabricantes [PK-1] [PK-2].

6.43 Tabla ARP

Mediante el envío de paquetes ARP falsos, es posible que un atacante degrade el rendimiento de un sistema (DoS) llenando la caché de rutas de IP con entradas ARP incorrectas. En algunos sistemas, el intervalo de expiración de la tabla ARP no puede ser modificado, pero sí en otros:

- Solaris:

El intervalo de expiración de la caché de IPs lo controlan dos parámetros, el de entradas solicitadas y no solicitadas. Desde el punto de vista de la seguridad, solo el segundo es de interés:

```
/usr/sbin/ndd -set /dev/arp arp_cleanup_interval 60000
```

- HP-UX:

El valor de expiración por defecto es de 5 minutos.

- Windows:

A través del editor del registro (*regedt32.exe*) debe localizarse la clave:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
```

Bajo la misma es necesario añadir el valor:

Value Name: *ArpCacheLife*

Data Type: *REG_DWORD*

Value: 600 (segundos), por defecto: 10 minutos.

6.44 Parches software de seguridad

Muchas de las vulnerabilidades comentadas existen o existieron debido a un error de diseño, programación o implementación de un determinado software, ya sea el propio sistema operativo o una aplicación servidor. Una vez conocida la vulnerabilidad, el fabricante del producto procede a generar y

distribuir lo que se conoce como un parche software de seguridad, es decir, una revisión o nueva versión de los binarios de software afectados por la vulnerabilidad, retocados de forma que se elimina el error descubierto. Los anuncios de las diferentes vulnerabilidades, así como la información y el procedimiento para subsanarlas se publican en numerosos portales Web dedicados a la seguridad, así como en las páginas Web de los fabricantes [W-1].

Por tanto, si se desea disponer de un entorno lo más seguro posible, es necesario mantener tanto los sistemas como los dispositivos de red actualizados con todos los parches software de seguridad liberados por el fabricante.

6.45 *Computer Emergency Response Team: CERT*

Finalmente, en el caso de sufrir una vulnerabilidad, al buscar ayuda legal al respecto, es probable que el departamento de policía local o estatal no esté preparado para enfrentarse a ella, por lo que es preferible recurrir a los grupos nacionales encargados de temas de seguridad tecnológica.

Por otro lado, existen numerosas organizaciones CERT mundiales a las que puede recurrirse en busca de asistencia [CE-1]. El CERT principal se encuentra en <http://www.cert.org>, junto al repositorio de información mantenido por éstos: <ftp://ftp.cert.org>. Algunos ejemplos son:

USA

FBI:

<http://www.fbi.gov/> - Página principal

<http://www.fbi.gov/contact/fo/fo.htm> - Lista de oficinas locales

1-202-324-3000 - Números de teléfono de la oficina principal

CERT:

<http://www.cert.org/> - Página principal

http://www.cert.org/contact_cert/contactinfo.html - Información de contacto

1-412-268-7090 - Línea del CERT 24 horas

Australia

AUSCERT:

<http://www.auscert.org.au/> - Página principal

http://www.auscert.org.au/Information/Auscert_info/contact.html - Información de contacto

+61 7 3365 4417 - Números de teléfono de incidentes

Europa:

Europa disponía de un CERT común que fue dividido en los diferentes países. Aquellos países que disponen de organizaciones que proporcionan servicios de CERT se encuentran disponibles en: <http://www.eurocert.net/>.

España

esCERT:

<http://escert.upc.es/> - Página principal

cert@escert.upc.es - Dirección de correo de incidentes

93-4015795, 93-4016984 – Números de teléfono de incidentes

Alemania

DFN-CERT:

<http://www.cert.dfn.de/> - Página principal

dfncert@cert.dfn.de - Dirección de correo de incidentes

+49 (0)40-42883-2262 - Números de teléfono de incidentes

Bundeskriminalamt BKA:

<http://www.bka.de/> - Página principal

+49 (0) 611-55-0

Inglaterra

JANET CERT:

<http://www.ja.net/CERT/cert.html> - Página principal

01235 822340 - Números de teléfono de incidentes

Varios servicios de policía:

<http://www.police.uk/> - Página principal

7 EJEMPLOS DE VULNERABILIDADES Y PROTECCIONES PARTICULARES

A modo de ejemplo se ha considerado interesante presentar algunos casos de vulnerabilidades concretas existentes (o que han existido y ya han sido resueltas) en el mundo real. Estos ejemplos sirven como muestra detallada de las posibilidades asociadas a las vulnerabilidades y protecciones mencionadas en los apartados anteriores.

Ni mucho menos pretenden ofrecer una muestra exhaustiva de ejemplos reales en las implementaciones de los fabricantes de equipos con pilas TCP/IP, ya que el número de éstas se cuenta por cientos de miles y además aumenta a pasos agigantados cada día [VU-1] [VU-2]. Asimismo, los equipos, servicios o S.O. reflejados no denotan un mayor índice de vulnerabilidad, sino que simplemente se han tomado como muestra representativa de ejemplo.

Las vulnerabilidades encontradas son clasificadas con un código identificativo o CVE [VU-2]. Previamente a esta clasificación pueden ser contempladas con un identificador CAN, representando vulnerabilidades candidatas a ser CVEs, pero que no han sido completamente comprobadas.

El único método que puede permitir un acercamiento mayor a las vulnerabilidades existentes, así como a las nuevas que van surgiendo, y a las ya solucionadas, es la suscripción a listas de distribución, grupos de *news*, servidores Web (portales) de organizaciones dedicadas a la seguridad o de los propios fabricantes [W-1].

Estos sistemas son actualizados diariamente para mantener informados tanto a la comunidad de administradores de sistemas y redes en Internet, como a la comunidad de *hackers*. Asimismo, cuentan con grandes bases de datos en las que almacenan todas las referencias pasadas.

7.1 Dispositivos Cisco: puerto de identificación 1999

Los dispositivos de red Cisco ejecutando su sistema operativo IOS utilizan el puerto 1999 como puerto de identificación [VU-3]. En algunos documentos aparece clasificado como sigue:

<code>tcp-id-port</code>	<code>1999/tcp</code>	<code>cisco identification port</code>
<code>tcp-id-port</code>	<code>1999/udp</code>	<code>cisco identification port</code>

Las peticiones recibidas en dicho puerto se manejan de forma diferente a las de los otros puertos; en este caso, al recibir un paquete TCP de SYN, responden inmediatamente con un paquete RST (en lugar de con el típico SYN-ACK), incluyendo el término “cisco” en el *payload* del paquete.

La implicación de este comportamiento es la sencillez a la hora de identificar dispositivos de este fabricante cuando se realiza un escaneo de puertos. Por tanto, aunque el dispositivo no permita el acceso a su puerto de *telnet*, es posible identificarlo.

Para protegerse de este ataque basta con impedir conexiones TCP externas al puerto en cuestión.

7.2 DoS en los puertos de acceso de los *routers*

Los puertos de acceso o configuración de los *routers*, habitualmente 23, 2001, 4001, 6001 y 9001, pueden ser atacados enviando unos pocos miles de caracteres (en ocasiones basta con enviarlos un par de veces), consiguiendo que el *router* no proporcione servicio durante el periodo de tratamiento de esa información [VU-4]. La denegación del servicio lógicamente perdurará más al enviarse un mayor número de paquetes.

Algunos dispositivos solo pueden ser restaurados mediante un *reset* manual del equipo, mientras que otros se recuperan a los pocos minutos, horas o incluso días. Este ataque ha afectado tanto al sistema operativo (S.O.) IOS de los Cisco como al ComOS de 3Com.

El impacto de este ataque es que condiciona al administrador a disponer de acceso físico al sistema, condición que no siempre se cumple, debido a la comodidad de las tareas de administración remotas. Un ejemplo del ataque es el envío de varias (3-6) ráfagas de información como sigue:

```
$ perl -e ` print 0xFF x 10000 ` | telnet router.dominio.com 4001
```

Tras la desconexión, un nuevo intento de conexión al puerto generará un mensaje de “*Connection refused*”, y el tiempo que se mantendrá el *router* en esta situación depende del modelo y la versión de S.O., pero puede variar entre 30 seg. y varios días.

7.3 NetBIOS

NetBIOS es un protocolo de transporte de datos (basado en sesiones) característico de los entornos Windows. Asimismo añade la funcionalidad de resolución de nombres en este entorno, a través del puerto UDP 137.

En Windows 2000 es posible ejecutar TCP/IP de forma nativa, pero no ocurre lo mismo en las versiones anteriores de Windows, por ejemplo NT. Las vulnerabilidades entorno a este protocolo son numerosas [L-1].

Un ejemplo concreto se centra en la gestión de nombres. Cuando un ordenador quiere dar a conocer alguno de sus recursos a la red, para compartirlo, éste debe reservar un nombre único para que el resto de equipos sepan identificarlo. De esta manera, el protocolo envía un mensaje *broadcast* - a todos los ordenadores de la red - "preguntando" si la denominación que quiere reservar está o no ocupada. En el supuesto de que no esté, el nombre del nuevo recurso se almacena en una lista en este equipo que queda visible al resto de clientes de la red para su acceso. Por el contrario, en el caso de que ya esté reservado por otro cliente, éste, al recibir el mensaje de petición, responde con otro de "conflicto". De esta manera, esta denominación queda "tachado" en la lista del demandante como ya en uso.

Supóngase qué, aunque el protocolo de bajo nivel (en este caso TCP/IP) funcione correctamente, *NetBIOS* deje de funcionar en un sistema *A*. En el momento en que cualquier cliente *B* quiera acceder a un recurso almacenado en el sistema *A*, éste utiliza el nombre del recurso compartido para llegar hasta él. *B* envía un *broadcast* (en el caso de que la dirección no se encuentre en la caché del sistema) preguntando quién es *A*. Como este último no puede responder a la petición del nombre, *B* no encuentra el equipo en la red. Lo mismo ocurre cuando el ordenador *A* intenta acceder a otro recurso en la red. Pese a que el resto de equipos si son capaces de responder, no saben a quien hacerlo con lo que *A* queda completamente aislado de la red. Debe tenerse en cuenta que Windows utiliza *NetBIOS* no sólo para resolver nombres, sino además se emplea para establecer ciertas sesiones de autenticación y enviar los contenidos de los recursos compartidos.

El fallo descubierto, cuya demostración se encuentra en un pequeño programa del grupo *hacker* “*Cult Of The Dead Cow*” [NB-1] [NB-2], intenta eliminar el nombre de un recurso en la red. Cuando se reconoce un nuevo recurso, el equipo que quiere registrar el nombre (que tiene formato de 15 caracteres y 1 *byte* que determina el tipo de servicio, algo así como un puerto UDP o TCP) envía un mensaje *broadcast* a la red del tipo *NAME REGISTRATION REQUEST*. Si otra máquina ya tiene registrado este nombre en su lista interna, envía de forma inmediata un mensaje al que realiza la petición de tipo *DENY*. El dispositivo que realizó la petición marca este nombre como "en conflicto" y deja de responder a cualquier tipo de mensajes dirigidos a este recurso. Lo mismo ocurre cuando un equipo "libera" uno de estos nombres, ya que pasará a ser marcado o directamente borrado de esta lista interna. ¿Qué es lo que ocurre cuando cualquier cliente envía, por amor al arte, uno de estos mensajes a otro equipo *DENY*?

En las implementaciones de *NetBIOS* (afecta desde Windows NT a 2000), este mensaje es aceptado en cualquier momento, incluso cuando no se ha realizado la consulta de registro. De esta forma, el cliente acepta el mensaje y marca en su lista de nombres su recurso como "en conflicto" y deja de utilizarlo.

La solución para evitar el fallo de este mecanismo es utilizar algún tipo de autenticación. De esta forma, el equipo que envía el mensaje maligno *DENY*, tendría que probar que en efecto es él el que ha registrado previamente el nombre. Desgraciadamente, *NetBIOS* no ha sido diseñado con esta característica y es necesario aplicar otro tipo de soluciones. La forma más efectiva de evitar que *NetBIOS* juegue una mala pasada es desactivarlo; no obstante, muchas redes basan su infraestructura en este sistema. Otra solución sería el uso de IPsec en el servicio asociado UDP.

7.4 Ataque *DNS cache*

Las versiones de BIND 4.9.5 y 8.1.X sufrían una vulnerabilidad asociada a la recursividad que permitía cachear información falseada. Cuando un cliente realiza una petición DNS a su servidor para un dominio sobre el que éste no tiene autoridad, si la recursividad está activa, el servidor local preguntará al servidor DNS con autoridad sobre el dominio, obtendrá la respuesta y se la reenviará al cliente solicitante. Esta información se almacenaba en caché para agilizar peticiones posteriores.

Mediante el ataque conocido como *PTR record spoofing*, un atacante podía insertar registros PTR en la caché del servidor a través de la recursividad, modificando la relación entre direcciones IP y

nombres. Asimismo, este ataque puede implicar DoS cuando un nombre de dominio concreto se asocia a una dirección IP inexistente, negando el servicio proporcionado por ese dominio.

El servicio de DNS ha sufrido numerosas vulnerabilidades en los últimos años [BN-1], algunas de ellas contempladas en el aviso del CERT con identificador “CERT *Advisory* CA-2001-02”. Tanto las vulnerabilidades como las recomendaciones asociadas para evitarlas, principalmente mediante la configuración del servicio, pueden obtenerse del ISC [BN-2].

7.5 Vulnerabilidades DoS en las pilas TCP/IP

El equipo de seguridad de RAZOR descubrió y documentó numerosas vulnerabilidades de las implementaciones TCP/IP [VU-5]. Todas ellas permiten a los atacantes consumir recursos del sistema (tiempo de CPU, ancho de banda de la red, memoria, así como recursos del *kernel* del S.O.: entradas de procesos, identificadores de ficheros abiertos...), y por tanto, reproducir un DoS.

Las implementaciones de TCP disponen de un número limitado de recursos al simular la máquina de estados del protocolo. Cualquier sistema que permite que ciertos recursos críticos sean consumidos, es potencialmente un candidato de un DoS.

Existe un ataque denominado *Naptha* [NA-1] [NA-2], basado en un DDoS, que permite como se ha comentado, consumir la totalidad de los recursos de un sistema, empleando para ello un gasto de recursos reducido (*ICMP broadcast*), mantiene el anonimato del atacante (*IP Spoofing*), y puede llevarse a cabo de forma distribuida, DDoS. El ataque se basa en el número de conexiones que puede mantener activas la pila TCP en los diferentes estados posibles: SN RECVD, ESTABLISHED, FIN WAIT 1, FIN WAIT 2...

La defensa frente a estos ataques pasa por:

- Aplicar los parches específicos del fabricante de la implementación de la pila TCP/IP.
- Parametrizar de forma apropiada, adecuándolo a su uso, el sistema.
- Preparar previamente las medidas a realizar en el caso de producirse un DoS.

Existen protecciones frente a *Naptha* de fabricantes como Compaq en su Unix Tru64, FreeBSD, Microsoft, Sun...

8 FUTURO

El futuro de la seguridad pasa por la generación de herramientas de control inteligentes, como los IDS basados en la IA, Inteligencia Artificial, así como por el desarrollo de sistemas de comunicaciones diseñados con la seguridad como característica fundamental, como es el caso de IPv6.

La incorporación y el desarrollo de las denominadas "redes inteligentes" podría dificultar considerablemente las actividades de los *hackers*. El Instituto Tecnológico de Georgia, EEUU, trabaja en un proyecto de desarrollo de redes neuronales, que probablemente aumentará la seguridad del tráfico digital. El término red neuronal refleja una de las técnicas empleadas en el área de la inteligencia artificial dentro de la computación. Ésta pretende reproducir el funcionamiento de aprendizaje y reconocimiento de las neuronas del cerebro humano, que aprenden de la experiencia, creando conexiones entre las distintas áreas de conocimiento. Con todo, cabe precisar que no se trata de redes que estén en condiciones de pensar, sino de sistemas capaces de identificar patrones en el flujo digital y aprender de los intentos de intrusión.

Hoy en día, los administradores de sistemas deben actualizar manualmente los sistemas de protección de las redes contra los ataques informáticos, ya que aparecen nuevas vulnerabilidades y técnicas de ataque con demasiada frecuencia. Con la incorporación de redes inteligentes se hará más previsible y fácil la contención de los intrusos [FT-1]. Tales redes estarán incluso en condiciones de detectar máquinas que monitorizan ilegalmente el tráfico de la red para captar y apoderarse de información, es decir, *sniffers*. La novedad es que las redes neuronales detectarán ese tipo de máquinas sin que sus operadores se percaten [FT-2].

Las implicaciones de la nueva versión de IP, IPv6 o *IP Next Generation*, en el área de las VPNs es realmente importante [NG-2] y facilitará su expansión. Debe partirse de la base de que el protocolo IPSec fue diseñado inicialmente para esta versión de IP, portándose posteriormente a la versión 4, ofreciendo así la posibilidad de trabajar con tráfico IP encriptado [NG-3]. Actualmente, los fabricantes de productos para VPNs aceptan que los sistemas no disponen de una implementación de la pila TCP/IP basada en la versión 6, por lo que se emplean modificaciones sobre las implementaciones actuales de la versión 4.

El IP Forum [NG-1] encargado de generar las especificaciones del protocolo y de su evolución futura tiene entre sus filas a los grandes fabricantes en el mundo de las redes: Cisco, 3Com, Microsoft, MCI Worldcom, Nokia, British Telecom, Siemens y otros.

9 CONCLUSIONES

La seguridad de las redes de comunicaciones, y concretamente de Internet, evoluciona a pasos agigantados cada minuto que transcurre. Nuevas vulnerabilidades y utilidades, tanto para explotarlas como para combatirlas, son distribuidas públicamente en la red. La información disponible al respecto es inmanejable, por lo que el diseño de un sistema de seguridad debe basarse en la fortaleza de las tecnologías empleadas, y no en la ocultación de las mismas: “*security through obscurity*”.

El protocolo TCP/IP sufre algunos problemas de seguridad por las características intrínsecas de su diseño, los cuales han sido ampliamente analizados a lo largo de los últimos años. La nueva versión de IP, versión 6, se diseñó con la seguridad en mente, de ahí la aparición del estándar IPSec, que junto a otras tecnologías, como las infraestructuras de clave pública, PKIs, permiten controlar y disolver muchas de las vulnerabilidades presentadas a lo largo del presente trabajo.

Asimismo, se puede concluir que la seguridad de una red no se basa en una única técnica exclusivamente, como promulga la idea errónea de securizar una red mediante un *firewall*, sino que viene reforzada por la utilización de multitud de tecnologías que permiten monitorizar y gestionar cada uno de los aspectos críticos de la red: encriptación, IDSs, *firewalls*, software específico de seguridad, protocolos seguros (SSL, SSH, IPSec), PKIs...

Lo que es más, mediante el uso exclusivo de las tecnologías mencionadas no es posible asegurar la seguridad de la red. Para ello es necesario a su vez disponer de procedimientos y políticas adecuadas que permitan concienciar a los usuarios y administradores de los sistemas informáticos, así como facilitar la aplicación de análisis y controles exhaustivos en la propia red y los elementos que la componen. Es por tanto necesario dedicar tiempo y esfuerzo a evolucionar la red hacia un entorno seguro, siendo necesario mantenerse actualizado (preferiblemente de forma automática, por ejemplo, mediante listas de distribución) de los avances que se realizan en este campo, así como de los nuevos avisos, vulnerabilidades y tecnologías que salen a la luz.

El objetivo final es asegurar ciertas características en las comunicaciones, como son, la autenticidad, la integridad de la información, la privacidad o confidencialidad, el no repudio, el control de acceso a la información...

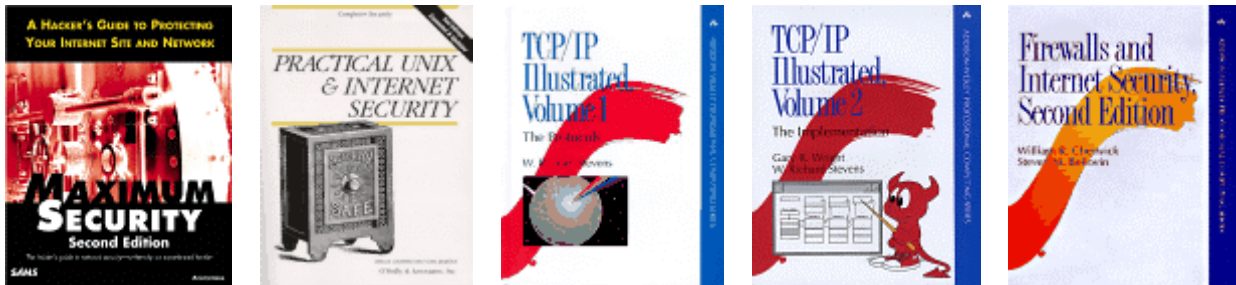
Finalmente, cabe concluir con una reflexión al respecto de la seguridad: el esfuerzo dedicado a la protección de un entorno debe ser directamente proporcional al valor de su contenido. Debido a que toda vulnerabilidad posee, más tarde o más temprano, su correspondiente protección, la vertiginosa carrera en la que la seguridad de las redes se debate actualmente, se centra en el mantenimiento constante y actualizado de las protecciones necesarias para evitar las vulnerabilidades existentes y ya conocidas.

10 BIBLIOGRAFÍA Y URLS

NOTAS:

Todas las URLs referenciadas en el presente trabajo han sido probadas durante el periodo de realización, hasta su publicación (mayo 2001 – junio 2002), lo que no asegura su existencia en el futuro, debido al dinamismo actual de Internet, que se ve incrementado en las páginas Web dedicadas a seguridad y al mundo *underground*.

Las referencias han sido clasificadas por orden alfabético del símbolo de referencia, es decir, [BO-1]. El significado del símbolo tiene que ver con el tema tratado, por ejemplo, BO-x, *Buffer Overflows*.



[BH-1] “Building a Windows NT bastion host in practice v1.3” y “Securing Windows NT/2000 Servers for the Internet (O'Reilly, 2000)”: <http://people.hp.se/stnor/>

[BH-2] “Building a Bastion Host Using HP-UX 11”. <http://people.hp.se/stevesk/bastion11.html>

[BH-3] “Building a Bastion Host Using HP-UX 10”. <http://people.hp.se/stevesk/bastion10.html>

[BH-4] ”Hardening Solaris Secure installation of Bastion hosts”:
http://www.boran.com/security/sp/Solaris_hardening2.html
http://www.securityportal.com/topnews/solaris_hardening20000523.html

[BH-5] “What is a bastion host?”: <http://www.sans.org/newlook/resources/IDFAQ/bastion.htm>

[BH-6] “Hardening Red Hat Linux with Bastille. Securely Installing a Bastion Host”
<http://www.securityportal.com/cover/coverstory20000501.html>

[BH-7] Windows bastion host: <http://secinf.net/info/nt/ntbastion/>

[BH-8] Linux Bastille: <http://www.bastille-linux.org/>

[BN-1] “CERT Advisory CA-2001-02 Multiple Vulnerabilities in BIND”:
<http://www.pgp.com/research/covert/advisories/047.asp>

[BN-2] BIND Security recommendations:
<http://www.isc.org/products/BIND/>
<http://www.isc.org/products/BIND/bind-security.html>

[BO-1] “The security implications of open source software”:
<http://www-106.ibm.com/developerworks/library/l-oss.html>

- [BO-2] “Smashing The Stack For Fun And Profit”. (Phrack 49. Volume Seven, Issue Forty-Nine. File 14 of 16). by Aleph One (aleph1@underground.org)
- [BO-3] “The Internet Worm”. Robert Morris, Jr. 1988. <http://www.zyvex.com/nanotech/worm.html>
- [BO-4] Securing applications: <http://www.boran.com/security/it13-applications.html>
- [BO-5] “How to write secure code”: <http://www.shmoo.com/securecode/>
- [BR-1] “Building Bastion Routers Using Cisco IOS”. Phrack Magazine - Vol. 9 - Issue 55.
- [CE-1] CERT organizations:
<http://securityportal.com/research/> - Research Center
<http://www.mycert.mimos.my/others.html>
- [CO-1] COPS: <ftp://info.cert.org/pub/tools/cops>
<http://dan.drydog.com/cops/>
- [DB-1] Smurf amplifiers: <http://www.powertech.no/smurf/>
- [DD-1] “DDoS Web attacks in the media”:
<http://www.zdnet.com/zdnn/stories/news/0,4586,2437202,00.html>
- [DD-2] “DDoS FAQ”: <http://securityportal.com/research/ddosfaq.html>
- [DD-3] <http://abcnews.go.com/sections/scitech/>
- [DD-4] “Protect systems from DoS attacks: general security principles”.
<http://all.net/journal/netsec/0004.html>
<http://www.cert.org/security-improvement/>
- [DD-5] “Protecting the border’s and gateways”: RFC2267 y RFC2644.
<http://www.rfceditor.com>
- [DD-6] ¿Qué hacer en caso de ser el objetivo de un DDoS?: <http://www.sans.org/y2k/DDoS.htm>
- [DD-7] Preparing DDoS evidence: <http://www.nanog.org/mtg-9910/>
- [DD-8] IDS tools to detect DDoS: <http://www.sans.org/giac.htm>
- [DD-9] DDoS News Flash: <http://www.cisco.com/warp/public/707/newsflash.html>
- [DD-10] DDoS Tools: <http://packetstorm.securify.com/> - Search: “distributed”
- [DD-11] DoS: <http://www.microsoft.com/TechNet/security/denialof.asp>
- [DD-12] “Trends in DoS Attack Technology”. CERT.
http://www.cert.org/archive/pdf/DoS_trends.pdf

- [DG-1] DIG: <http://www.inetdaemon.com/compendium/internet/dns/dig.html>
- [DG-2] DIG: <http://www.dns.net/dnsrd/tools.html>
- [DG-3] DIG Web services:
http://www.stcnet.ch/service/tools/dig_dns_check.htm
<http://www.netliner.com/dig.html>
http://www.ip-plus.net/tools/dig_dns_set-en.html
- [FA-1] Security Considerations for IP Fragment Filtering:
<http://www.alternic.org/rfcs/rfc1800/rfc1858.pdf>
<http://www.ipa.go.jp/security/rfc/RFC1858EN.html>
- [FI-1] Finger Bomb: (Network ICE Firewall)
http://www.networkice.com/Advice/Exploits/Services/finger/finger_bomb/default.htm
- [FI-2] Finger Bomb: <http://xforce.iss.net/static/47.php>
- [FI-3] Kaput: <http://www.sli.net.nz/cdroms/hacktoolkit/> - DoS – Multi Programs
- [FN-1] IP personality: <http://ippersonality.sourceforge.net/>
- [FN-2] Buscadores:
<http://www.google.com>
<http://www.yahoo.com>
<http://www.altavista.com>
<http://www.lycos.com>
<http://www.ferretsoft.com>
<http://www.dogpile.com>
<http://www.buscopio.com>
<http://www.ozu.es>
- [FN-3] RFC 2196: “Site Security Handbook”: <http://www.faqs.org/rfcs/rfc2196.html>
- [FN-4] Traceroute: <ftp://ftp.ee.lbl.gov/traceroute.tar.Z>
<ftp://ftp.ee.lbl.gov/traceroute-1.4a5.tar.Z>
- [FN-5] VisualRoute: <http://www.visualroute.com>
- [FN-6] NeoTrace: <http://www.neotrace.com>
- [FN-7] Snort: <http://www.snort.org>
- [FN-8] RotoRouter: <http://packetstorm.securify.com/UNIX/loggers/rr-1.0.tgz>
- [FN-9] fping: http://packetstorm.securify.com/Exploit_Code_Archive/fping.tar.gz
- [FN-10] gping: <http://www.hackingexposed.com/tools/tools.html>

- [FN-11] Pinger: <http://www.nmrc.org/files/snt>
- [FN-12] Scanlogd: <http://www.openwall.com/scanlogd/>
- [FN-13] IPPL: <http://pltplp.net/ippl/>
- [FN-14] ICMP information:
<http://packetstorm.securify.com/UNIX/scanners/icmpquery.c>
<http://packetstorm.securify.com/UNIX/scanners/icmpush22.tgz>
- [FN-15] Fingerprinting traces: <http://www.enteract.com/~lspitz/traces.txt>
- [FN-16] QUESO: <http://www.apostols.org/projectz/queso/>
- [FN-17] Fingerprinting (Fyodor): <http://www.insecure.org/nmap/nmap-fingerprinting-article.html>
- [FN-18] “Insertion, Evasion, and DoS: Eluding Network Intrusion Detection”:
<http://www.clark.net/~roesch/idspaper.html>
- [FN-19] Find network O.S.: <http://project.honeynet.org/papers/>
- [FN-20] Siphon: <http://www.subterrain.net/> - “Projects - Siphon”
- [FN-21] Cheops: <http://www.marko.net/cheops>
Tkined: <http://wwwhome.cs.utwente.nl/~schoenw/scotty/>
- [FN-22] Netcat o nc: <http://www.l0pht.com/~weld/netcat/index.html>
- [FN-23] Fingerprinting defense: http://www.pgci.ca/p_fingerprint.html
<http://www.pgci.ca/fingerprint.html>
- [FN-24] Fingerprinitnting:
<http://www.net-security.org/text/articles/spitzner/fingerprinting.shtml>
- [FN-25] Fingerprinitnting:
http://www.sans.org/newlook/resources/IDFAQ/TCP_fingerprinting.htm
- [FS-1] “Format String Attacks”. Tim Newsham, Guardent, Inc. September 2000.
- [FT-1] “New methods for Intrusion Detection”: http://www.infowar.com/survey/ids_newm.html
- [FT-2] IDS basados en redes neuronales: <http://www.gtri.gatech.edu/res-news/rchnews.html>
- [FW-1] IP Chains How To: <http://www.linuxdoc.org/HOWTO/IPCHAINS-HOWTO.html>
- [FW-2] <http://www.securityportal.com/lasg/firewall/index.html>
- [FW-3] “IP Chains configuration Tool”: <http://www.linux-firewall-tools.com/linux/firewall/>
- [FW-4] Firewall 1 – Checkpoint: <http://www.checkpoint.com/products/firewall-1/index.html>

- [FW-5] Raptor – Axent: <http://www.raptor.com>
VelociRaptor – Synmatec: <http://www.axent.com>
- [FW-6] StoneGate: <http://www.stonesoft.com>
- [FW-7] IPF: <http://www.securityportal.com/closet/closet20000216.html>
- [FW-8] Cisco PIX: <http://www.cisco.com/go/pix>
- [FW-9] PGP Gauntlet: <http://www.pgp.com/products/gauntlet/default.asp>
- [HI-1] Juggernaut: <http://www.packetfactory.net>
- [HI-2] Hunt: <http://lin.fsid.cvut.cz/~kra/index.html>
- [HI-3] Demonstration: session hijacking:
<http://staff.washington.edu/dittrich/talks/qsm-sec/hijack.html>
<http://staff.washington.edu/dittrich/talks/qsm-sec/script.html>
- [HI-4] “Man in the middle attacks”: <http://www.sans.org/infosecFAQ/threats/middle.htm>
- [HI-5] Hijacking SSH/SSL demo: <http://www.itcon-ltd.com/Simulationsm.htm>
- [HI-6] Hijacking tool: dsniff. <http://www.monkey.org/%7edugsong/dsniff/>
- [HI-7] Hijacking tool: ettercap. <http://ettercap.sourceforge.net/>
- [HV-1] “Secrets and Lies : Digital Security in a Networked World”. Bruce Schneier, 2000.
John Wiley & Sons. ISBN: 0471253111. <http://www.counterpane.com>
- [HV-2] “Ataques semánticos: la tercera ola de ataques a redes”. Bruce Schneier.
http://www.kriptopolis.com/criptograma/0030_1.html
- [HV-3] “Un ataque semántico sobre URLs” Bruce Schneier.
http://www.kriptopolis.com/criptograma/0034_7.html
- [I-1] “Estadísticas de nodos en Internet”: <http://www.isc.org/ds/>
- [I-2] “Estadísticas de nodos Internet europeos”:
<http://www.ripe.net/ripenncc/mem-services/registration/statistics/index.html>
- [I-3] “Estadísticas de nº. de usuarios en Internet”: <http://www.nua.ie/surveys/>
- [IC-1] “ICMP usage in scanning”: <http://www.plus.or.kr/seminar/icmp.ppt>
- [IC-2] “ICMP scanning”: http://qb0x.net/papers/Scans/ICMP_Scanning_v2.5/
- [IC-3] ISIC tool: <http://expert.cc.purdue.edu/~frantzen/>

- [ID-1] Database of Intrusions detected by Network ICE:
http://www.toyo.co.jp/security/ice/advice_old/Intrusions/default.htm
- [ID-2] “State of the Practice of Intrusion Detection Technologies”
<http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028chap01.html>
- [IP-1] “IPSec Working Group's charter on the IETF”:
<http://www.ietf.org/html.charters/ipsec-charter.html>
- [IP-2] “IPSec Developers' Forum”: <http://www.ip-sec.com/>
- [IP-3] IPSec RFCs: -- RFC 2401, RFC 2402 , and RFC 2406 -- RFC 2409.
<http://www.rfc-editor.org/>
- [IP-4] IPSec WhitePaper:
http://www.cisco.com/warp/public/cc/techno/protocol/ipsecur/ipsec/tech/ipsec_wp.htm
- [IP-5] An introduction to IPSec: <http://www.cisco.com/warp/public/105/IPSECpart11.html>
- [IP-6] Cisco IPSec:
http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113t/113t_3/ipsec.htm
- [IP-7] IPSec security: <http://www.counterpane.com/ipsec.html>
- [IP-8] RSA challenges: <http://www.rsasecurity.com/rsalabs/challenges/>
- [IP-9] RC5 cracking: <http://www.distributed.net/rc5/index.html.en>
- [IP-10] “Probable PlainText Cryptoanalysis of the IPSec security protocols”:
<http://www.computer.org/proceedings/sndss/7767/7767toc.htm>
- [IS-1] CERT Advisory CA-1995-01 “IP Spoofing Attacks and Hijacked Terminal Connections”
<http://www.cert.org/advisories/CA-1995-01.html>
- [IS-2] CERT Advisory CA-2001-09 “Statistical Weaknesses in TCP/IP Initial Sequence Numbers”
<http://www.cert.org/advisories/CA-2001-09.html>
- [IS-3] Linux Exploit: <http://teso.scene.at/releases/adv1.tar.gz>
- [IS-4] Linux:
http://www.securiteam.com/unixfocus/Linux_Kernel_2_2_x_ISN_vulnerability_makes_the_kernel_vulnerable_to_blind_TCP_spoofing.html
- [IS-5] Cisco ISN randomness:
<http://www.cisco.com/warp/public/707/ios-tcp-isn-random-pub.shtml>
http://kernelnotes.org/lxnlsts/linux-kernel/lk_9909_04/msg00664.html
- [IS-6] ISN random implementation:
<http://lxr.linux.no/source/drivers/char/ChangeLog#L258>
<http://lxr.linux.no/source/drivers/char/random.c#L1855>

[IS-7] CERT Advisory CA-1996-21: "TCP SYN Flooding and IP Spoofing Attacks"
<http://www.cert.org/advisories/CA-1996-21.html>

LIBROS: [L-#]

[L-1] "Hacking Exposed: Network security secrets & solutions" (Second Edition, 2001). J. Scambray, S. McClure, G.Kurtz. Osborne - Mc Graw Hill. ISBN: 0072127481.
<http://www.hackingexposed.com>

[L-2] "Practical Unix and Internet Security". Simson Garfinkel, Gene Spafford. (2nd edition, 1996). O'Reilly & Associates. ISBN: 1565921488.

[L-3] "Maximum Security: A Hacker's Guide to Protecting Your Internet Site and Network". (2nd Bk&cdr edition, 1998). Sams. ISBN: 0672313413.

[L-4] "TCP/IP Illustrated, Volume 1: The Protocols (Addison-Wesley Professional Computing Series). W. Richard Stevens. (Vol. 1, 1994). Addison-Wesley Pub Co. ISBN: 0201633469.

[L-5] "TCP/IP Illustrated, Volume 2: The Implementation (Addison-Wesley Professional Computing Series). Gary R. Wright (Contributor), Wright Gary R., W. Richard Stevens. (Vol. 2, 1995). Addison-Wesley Pub Co. ISBN: 020163354X.

[L-6] "TCP/IP Illustrated, Volume 3: TCP for transactions, HTTP, NNTP, and the Unix Domain Protocols. (Addison-Wesley Professional Computing Series). W. Richard Stevens. (Vol. 3, 1995). Addison-Wesley Pub Co. ISBN: 0201634953.

[L-7] "Advanced Programming in the Unix Environment (Addison-Wesley Professional Computing Series)". W. Richard Stevens. 1992. Addison-Wesley Pub Co. ISBN: 0201563177.

[L-8] "Maximum Linux Security: A Hacker's Guide to Protecting Your Linux Server and Workstation". 1999. Sams. ISBN: 0672316706.

[L-9] "Hacker Proof: The Ultimate Guide to Network Security". Lars Klander, Edward J., Jr. Renehan. 1997. Jamsa Press; ISBN: 188413355X.

[L-10] "Cisco IOS 12.0 Network Security". Cisco Systems. 1999. Cisco Pr. ISBN: 1578701600.

[L-11] "Designing Network Security". Cisco Press Fundamentals Series. 1999. ISBN: 1578700434.

[L-12] "Hacking Linux Exposed: Network Security Secrets & Solutions". Lee, James B. / Carasik, Anne / Hatch, Brian / Kurtz, George / Shah, Saamil. McGraw-Hill. 2001. ISBN: 0072127732.

[L-13] "Network Intrusion Detection: An Analyst's Handbook". Northcutt, Stephen / McLachlan, Donald / Novak, Judy. (2nd Edition). New Riders Publishing. 2000. ISBN: 0735710082.

[L-14] "Construya Firewalls para Internet". D. Brent Chapman y Elizabeth D. Zwicky. O'Reilly. 1997. ISBN: 1565921240.

[L-15] “Firewalls and Internet Security: Repelling the Wily Hacker”. William R. Cheswick, Steven M. Bellovin (Contributor). 2nd edition, 2000. Addison-Wesley Pub Co; ISBN: 020163466X

[L-16] “Techniques Adopted By 'System Crackers' When Attempting To Break Into Corporate or Sensitive Private Networks”. By the consultants of the Network Security Solutions Ltd. Front-line Information Security Team (FIST), December 1998. - fist@ns2.co.uk - <http://www.ns2.co.uk>

[L-17] Cisco Packet Magazine Q1 2001: “Security in the 21st Century”

[L-18] Information Systems Security: <http://www.auerbach-publications.com/iss/>

[LI-1] “Linux Administrator's Security Guide”: <http://www.seifried.org/lasg/>

[LK-1] Loki: ”Phrack Magazine: Volume Seven, Issue Forty-Nine, File 06 of 16, Project Loki.”

[LN-1] Land attack (DoS): <http://www.insecure.org/sploits/land.ip.DOS.html>

[LN-2] TCP Loopback DoS Attack (land.c) and Cisco Devices:
<http://www.cisco.com/warp/public/770/land-pub.shtml>

[LN-3] CA-1997-28: IP DoS Attacks
<http://www.cert.org/advisories/CA-1997-28.html>

[MP-1] MPLS e IPsec: <http://www.cisco.com/go/packet/mplsipsec>

[NA-1] Naptha: <http://www.sans.org/infosecFAQ/threats/naptha.htm>
<http://www.sans.org/infosecFAQ/malicious/naptha.htm>

[NA-2] RAZOR Naptha: http://razor.bindview.com/publish/advisories/adv_NAPTHA.html

[NB-1] NetBIOS deny: <http://cultdeadcow.com>

[NB-2] NetBIOS exploit: <http://pr0n.newhackcity.net/~sd/nbname.html>

[NF-1] <http://www.cisco.com/public/cons/isp/documents/IOSEssentialsPDF.zip>

[NG-1] Ipv6 Forum: <http://www.ipv6forum.com>

[NG-2] “IPv6 and VPN security”. Tim Greene. Network World Fusion Focus on VPNs, 7/99.

[NG-3] “The Effects of the Transition to IPv6 on Internet Security”:
<http://www.tml.hut.fi/Opinnot/Tik-110.501/1999/papers/ipv6/ip6sec.html>

[NG-4] Cisco IPv6: <http://www.cisco.com/go/packet/ipv6>

[NG-5] IPv6 IETF: <http://www.ietf.org/html.charters/ipngwg-charter.html>

[NT-1] NTP software: <http://www.eecis.udel.edu/~ntp/software.html>

[PA-1] Listado de passwords por defecto de los diferentes fabricantes:
<http://www.phenoelit.de/dpl/dpl.html>

- [PG-1] PGP: <http://www.pgp.com>
- [PG-2] The International PGP home page: <http://www.pgpi.org/>
- [PI-1] “Ping of death” page:
<http://www.insecure.org/sploits/ping-o-death.html>
<http://www.dfm.dtu.dk/netware/pingod/ping.html>
- [PK-1] “Understanding the Public-Key Infrastructure”. Carlisle Adams, Steve Lloyd. ISBN: 157870166X.
- [PK-2] “Ten Risks of PKI”. Whitepaper by Ellison and Stinger.
<http://www.counterpane.com/pki-risks.html>
- [PS-1] Penetration Testing in Windows: <http://www.sunbelt-software.com/product.cfm?id=810>
- [PS-2] Programación segura:
http://umeet.uninet.edu/umeet2001/talk/10-12-2001/umeet/umeet_files/v3_document.htm
- [PT-1] NMAP: <http://www.insecure.org/nmap> - previous <http://www.dhp.com/~fyodor/>
- [PT-2] RFC 793: <http://www.rfceditor.com> : <ftp://ftp.isi.edu/in-notes/rfc793.txt>
- [PT-3] Port scan analysis: <http://www.openwall.com/scanlogd/P53-13.gz>
- [PT-4] PortSentry – Abacus Sentry: <http://www.psionic.com>
- [PT-5] Logcheck: <http://www.psionic.com/abacus/logcheck/>
- [PT-6] Firewall-1 port scan detection: <http://www.enteract.com/~lspitz/intrusion.html>
- [PT-7] BlackICE: <http://www.networkice.com>
- [PT-8] Port numbers: <http://www.iana.org/assignments/port-numbers>
- [RP-1] RIP spoofing: <http://www.technotronic.com/horizon/ripar.txt>
- [SF-1] IP Filter software: <http://coombs.anu.edu.au/ipfilter/>
- [SI-1] OptOut -- Packet Sniffing Details and Resources: <http://grc.com/oo/packetsniff.htm>
- [SI-2] Sniffing FAQ: <http://www.robertgraham.com/pubs/sniffing-faq.html>
- [SL-1] Netscape security: <http://home.netscape.com/security/index.html>
- [SL-2] OpenSSL: <http://www.openssl.org/>
- [SL-3] SSL v3.0 specification: <http://home.netscape.com/eng/ssl3/>

- [SM-1] “Stopping Smurf and Spam”: <http://www.securityportal.com>
- [SM-2] Sendmail spamming: <http://www.sendmail.org>
- [SM-3] S-MIME Working Group: <http://www.imc.org/ietf-smime/>
- [SM-4] S-MIME Central: <http://www.rsasecurity.com/standards/smime/>
- [SM-5] Qmail: <http://www.qmail.org>
- [SM-6] CA-1998-01: Smurf DoS attacks: <http://www.cert.org/advisories/CA-1998-01.html>
- [SN-1] Tripwire:
<ftp://coast.cs.purdue.edu/pub/tools/unix/ids/tripwire>
<http://www.tripwiresecurity.com>
- Derivados de Tripwire e IDSs:
- ViperDB: <http://www.resentment.org/projects/viperdb/>
- Triplight: <http://packetstormsecurity.org/UNIX/IDS/indexdate.shtml>
- AIDE: <http://www.cs.tut.fi/~rammer/aide.html>
- Sentinel: <http://www.securitysoft.com/>
- [SN-2] CPM: <ftp://coast.cs.purdue.edu/pub/tools/unix/sysutils/cpm>
- [SP-1] SNMP v3: <http://www.ietf.org/html.charters/snmpv3-charter.html>
- [SS-1] “Guide to securing Intranet and Extranet servers”. Verisign, 2000.
<http://www.verisign.com/rsc/gd/ent/secure-ext/intro.html>
- [SS-2] “Web Commerce Technology Handbook”. D. and E. Minoli. McGraw-Hill, 1998. ISBN: 0070429782.
- [SS-3] Autoridades de Certificación (CAs):
- Verisign: <http://www.verisign.com>
- Thawte: <http://www.thawte.com>
- Baltimore: <http://www.baltimore.com>
- Entrust: <http://www.entrust.com>
- FNMT: <http://www.fnmt.es> (España)
- [SS-4] SSH.org: <http://www.ssh.org/>
- [SS-5] SSH.com: <http://www.ssh.com/>
- [SS-6] OpenSSH: <http://www.openssh.com/>
- [ST-1] Stacheldraht: <http://staff.washington.edu/dittrich/misc/stacheldraht.analysis>
- [ST-2] <http://staff.washington.edu/dittrich/misc/sickenscan.tar>
- [SY-1] <http://www.rootshell.com> - “synk”: SYN-Flood generator (Linux)

- [SY-2] <http://www.niksula.cs.hut.fi/~dforsber/synflood/result.html> - TCP SynFlood study
- [TC-1] TCP/IP introduction: <ftp://ftp.infonexus.com/pub/Philes/NetTech/TCP-IP/tcipIp.intro.txt.gz>
- [TC-2] TCP/IP Secrets: <http://www.windowsitlibrary.com/Content/329/04/toc.html>
- [TC-3] TCP/IP Illustrated Vol.1: http://lib.nevalink.ru/tcp_stevens/
- [TC-4] “UNIX IP Stack Tuning Guide v2.7”:
<http://www.cymru.com/~robt/Docs/Articles/ip-stack-tuning.html>
- [TC-5] SYN cookies: <http://cr.yp.to/syncookies.html>
- [TD-1] Teardrop: <http://www.attrition.org/security/denial/w/teardrop.dos.html>
- [TD-2] Syndrop: <http://www.csclub.stthomas.edu/~bugtraq/1998/msg00704.html>
- [TD-3] Teardrop and land attacks: <http://www.hut.fi/~lhuovine/hacker/dos.html>
- [TD-4] Teardrop: <http://www.fortunecity.com/bennyhills/straightman/13/dos/teardrop.htm>
- [TD-5] Teardrop attack: <http://www.vircio.org/doc/teardrop1.htm>
- [TD-6] Teardrop fix:
<ftp://ftp.microsoft.com/bussys/winnt/winnt-public/fixes/usa/nt40/hotfixes-postSP3/simptcp-fix>
- [TD-7] Teardrop2: <http://www.nta-monitor.com/news/tdrop2.htm>
- [TD-8] Teardrop2 fix: <http://www.utexas.edu/cc/utconn/answers/windows/teardrop.html>
- [TD-9] Teardrop: <http://www.wired.com/news/technology/0,1282,9581,00.html>
- [TF-1] TFN: <http://staff.washington.edu/dittrich/misc/tfn.analysis>
- [TF-2] http://staff.washington.edu/dittrich/misc/ddos_scan.tar
- [TF-3] TFN defense tools:
<http://www.keir.net> DDoSPing
<http://www.nipc.gov> find_ddos
- [TF-4] TFN2K: http://packetstorm.securify.com/distributed/TFN2k_Analysis-1.3.txt
- [TI-1] TIGER tool: <http://www.net.tamu.edu/network/tools/tiger.html>
- [TI-2] TARA: <http://www-arc.com/tara/index.shtml>
- [TJ-1] TLSecurity information: <http://www.tlsecurity.net/trojanh.htm>
- [TJ-2] TLSecurity DB: <http://www.tlsecurity.net/tlfaq.htm>

- [TJ-3] TDS: <http://www.multimania.com/ilikeit/tds2.htm>
- [TJ-4] TheCleaner: <http://www.moosoft.com/cleaner.html>
- [TJ-5] <http://www.commodon.com/menu.htm>
<http://www.chebucto.ns.ca/~rakerman/port-table.html>
- [TP-1] TCP Wrappers (TCPD) Under FreeBSD: <http://flag.blackened.net/freebsd/tcpd.html>
- [TP-2] TCPD: Unix - Logging and Restriction: <http://vancouver-webpages.com/security/tcpd.html>
- [TP-3] Installing and configuring TCP Wrappers on Solaris 7 and Solaris 8:
<http://www.kempston.net/solaris/tcpwrappers.html>
- [TJ-6] BackOfficer Friendly: <http://www.nfr.net/products/bof/>
- [TR-1] <http://staff.washington.edu/dittrich/misc/trinoo.analysis>
- [TR-2] <http://www.fbi.gov/> - Buscar por “trinoo”
- [U-1] Phrack: <http://www.phrack.com>
<http://phrack.infonexus.com>
- [UX-1] “Enhancing Security on Unix systems”
<http://www.vtcif.telstra.com.au/pub/docs/security/ser-doc/unix-security.html>
- [VP-1] IETF: <http://www.ietf.org>
<http://www.ietf.org/html.charters/pppext-charter.html>
- [VP-2] PPTP vulnerability: <http://www.counterpane.com/pptp.html>
<http://www.counterpane.com/pptp-faq.html>
- [VP-3] PPTP vulnerability. ”Phrack Magazine: Volume 12, Issue 53”. Aleph One.
- [VP-4] Microsoft PPTP solution: <http://www.counterpane.com/pptpv2-paper.html>
- [VP-5] PPTP fixes: <http://www.microsoft.com/technet/security/bulletin/ms98-012.asp>
- [VP-6] Securing PPTP and VPN:
http://www.microsoft.com/ISN/whitepapers/microsoft_virtual_pr_952.asp
<http://www.microsoft.com/ntserver/zipdocs/vpnsecur.exe>
http://www.microsoft.com/ntserver/commserv/deployment/moreinfo/VPNSec_FAQ.asp
- [VU-1] “Statistics about the number of vulnerabilities in BugTraq database”:
<http://www.securityfocus.com/vdb/stats.html>
- [VU-2] “Common Vulnerabilities and Exposures (CVE) Editorial Board”: <http://cve.mitre.org>
- [VU-3] “Remote Cisco Identification”:

<ftp://ftp.technotronic.com/routers-switches/cisco-identification.txt>

[VU-4] “Router Access Port DoS”:

<ftp://ftp.technotronic.com/routers-switches/access-port-DoS.txt>

[VU-5] CERT Advisory CA-2000-21 “Dos Vulnerabilities in TCP/IP Stacks”

<http://www.cert.org/advisories/CA-2000-21.html>

[W-1] Ver listado (*más abajo*) de las webs de seguridad de los distintos fabricantes.

[WI-1]: ICANN: <http://www.icann.org/>

IANA (Internet Assigned Numbers Authority): <http://www.iana.org>

ARIN: <http://www.arin.net>

RIPE: <http://www.ripe.net>

APNIC: <http://www.apnic.net>

España: <http://www.nic.es> (ES-NIC)

Mundial: <http://www.allwhois.com>

[WN-1] Winnuke test page: <http://www.jtan.com/resources/winnuke.html>

[WN-2] The Winnuke Relief page: <http://www.users.nac.net/splat/winnuke/>

[WN-3] NukeNabber: <http://www.dynamicsol.com/puppet/>

[WN-4] http://www.wwdsi.com/demo/saint_tutorials/winnuke.html

[WN-5] <http://www.h0wt0.com/dosattacks/winnuke.html>

[WN-6] Out-of-Band attacks: <http://support.microsoft.com/support/kb/articles/Q168/7/47.asp>

[WN-7] Anti Winnuke: <http://members.nbc.com/sanx/spindex.htm>

[WN-8] <http://www.giga.com/~thunder/winnuke/>

[WN-9] Supernuke: <http://www.giga.com/~thunder/winnuke/>

[WN-10] Winnuke: <http://www.attrition.org/security/newbie/security/winnuke.html>

[WR-1] Wireless Wardriving: <http://www.wardriving.com>

[WR-2] Net Sumbler: <http://netsumbler.com>

11 ANEXO I: WEBS DE SEGURIDAD

El objetivo de este ANEXO es englobar en un apartado común un conjunto de enlaces y referencias interesantes existentes en Internet relacionadas con la seguridad. Dentro de éstos existen tanto empresas privadas que distribuyen productos de seguridad, como empresas de servicios de seguridad, páginas personales, webs *underground*, organismos oficiales, webs de comunidades y grupos relacionados con la seguridad...

11.1 Portales y repositorios de seguridad (*en orden alfabético*)

http://ccc.osb.net/Network_Security/home.htm - Network Security
http://ciberia.ya.com/n_b_k/ - NBK (underground)
<http://csrc.nist.gov/> - CSRC: Computer Security Resource Center
<http://cultdeadcow.com> - Cult of the Dead Cow group
<http://cve.mitre.org> - Common Vulnerabilities and Exposures (CVE)
<http://defaced.alldas.de/> - Defaced Web Sites (sitios atacados en Internet)
<http://farm9.com/> - Farm9 Tools (security managed)
<http://grc.com/> - Gibson Research Corporation (security tools and tests)
<http://hackersclub.com/km> - HackersClub (underground)
<http://hackerwatch.org/> - Sitio web para enviar información de ataques sufridos
<http://hispahack.ccc.de/> - HispaHack
<http://linux.oreillynet.com/topics/linux/security> - Linux Security
<http://ntsecurity.nu> - Windows security
<http://packetstorm.decepticons.org> - Packet Storm
<http://packetstorm.securify.com> - Packet Storm
<http://project.honeynet.org> - The Honeynet project (Honey pots)
<http://qb0x.net/> - QBOX
<http://qb0x.net/papers/index.php> - Qbox papers
<http://rootshell.connectnet.com> - RootShell
<http://safer.siamrelay.com> - Security Alert for Enterprise Resources
<http://searchsecurity.com/> - SearchSecurity Portal
<http://search.securepoint.com/index.php> - SecurePoint messages
<http://secinf.net/> - Network Security Library
<http://seclab.cs.ucdavis.edu/> - Computer Security Laboratory
<http://securityportal.com> - Security Portal
<http://securityportal.com/research/research.commonvuln.html> - Common Vulnerabilities
<http://staff.washington.edu/dittrich/talks/agora/index.html> - Some TCP/IP vulnerabilities
<http://tlsecurity.net/> - Seguridad (underground)
<http://warez.com> - Warez (underground)
<http://www.2600.com/> - 2600 The Hacker Quaterly (*e-zine*)
<http://www.alw.nih.gov/Security/prog-full.html> - Security Software
<http://www.antioffline.com/> - Antioffline (removing de dot in .com)
<http://www.antionline.com/archives/documents/advanced/> - Antionline's Text Archives
<http://www.antionline.com/directory/> - Antionline directory (books, links...)
<http://www.astalavista.com> - Buscador Underground
<http://www.attrition.org/> - Attrition
<http://www.atstake.com/> - @tstake company
<http://www.blackcode.com/archive/> - BlackCode *exploits* archive

<http://www.blackhat.com/> - Black Hat Hackers
<http://www.boran.com/security/> - IT Security cookbook
<http://www.bvdh.com.ar/hcpv.htm> - HCPV
<http://www.ccc.de/> - Chaos Computer Club (Alemania)
<http://www.cert.org/summaries/> - “CS-año-# – Special Edition” documentos sobre temas concretos.
<http://www.chez.com/doggzland/hpcv.htm> - HPCV Francés (underground)
<http://www.ciac.org/ciac/> - CIAC: Computer Incident Advisory Center
<http://www.cisecurity.org/> - CIS: The Center for Internet Security
<http://www.counterpane.com> - Counterpane company
<http://www.criptored.upm.es> - Red de criptografía hispana
<http://www.cs.purdue.edu/coast/coast.html> - COAST project
<http://www.cybertech.com> - CyberTech site (UHF: uhf_united@hotmail.com)
<http://www.cyervo.com.ar> - LaCovacha (underground)
<http://www.deepzone.org/> - DeepZone
<http://www.deepzone.org/quick.asp?link=s1ms> - S1M's Quick DB (Base de datos de vulnerabilidades)
<http://www.defcon.org> - DEF CON
<http://www.denialinfo.com/> - DoS resources
<http://www.epm.ornl.gov/~dunigan/security.html> - Lista *URLs* de seguridad.
<http://www.esecurityonline.com> - eSecurityOnline company
<http://www.etext.org/Zines/ASCII/CoTNo/> - Cotno e-zine
<http://www.first.org/> - Forum of Incidents Response and Security Teams
<http://www.foundstone.com> - Foundstone company
<http://www.galeon.com/lacovacha/> - LaCovacha (underground)
<http://www.galeon.com/lacovacha/hk/buscadores.htm> - Buscadores de seguridad y HCPV
<http://www.gnusec.com> - GNUSEC
<http://www.google.com> - Busca por HCPV: Hacking, Cracking, Phreaking, Viring.
<http://www.grc.com/> - GRC
<http://www.grcsucks.com/> - GRC Sucks <http://www.groovyweb.f2s.com> - GroovyWeb
<http://www.groovyweb.uklinux.net/> - GroovyWeb computer security
<http://www.guerrilla.net> - Wireless (underground)
<http://www.hack3rs.com/> - New hackers
<http://www.hackbusters.net/> - HackBusters (LaBrea tool)
<http://www.hackemate.com.ar/> - Hackemate Argentina (underground)
<http://www.hacker.com.br> - Hackers (underground)
<http://www.hackersclub.com/km/> - Hackers Club (underground)
<http://www.hackerslab.com> - HackerLabs company
<http://www.hackerslab.org/eorg/hackingzone/hackingzone.htm> - Hacker Labs
<http://www.hackerz.org> - Hackerz (underground)
<http://www.hackindex.org> - Hackindex (underground)
<http://www.hackingexposed.com> - HackingExposed book web pages
<http://www.hackingexposed.com/tools/tools.html> - Tools
<http://www.hackinthebox.org> - Hack in the Box
<http://www.heinekenteam.com/> - Heineken (underground)
<http://www.hispasec.com/> - HispaSec
<http://www.hispasec.com/sana/> - SANA: Servicio de Análisis, Notificación y Alertas (**de pago**)
<http://www.icsalabs.com> - International Computer Security Association
<http://www.ideahamster.org/> - IdeaHamster Organization (Open Standards Security)
<http://www.incidents.org> - Incidents.org
<http://www.incident-response.org> - Incident-response.org
<http://www.infohackers.org/> - AIH: Asociación para información de hackers

<http://www.infonexus.com/> - InfoNexus
<http://www.informativos.info/> - AIH (Boletín)
<http://www.itprc.com/security.htm> - Network security
<http://www.kb.cert.org/vuls/> - CERT vulnerabilities notes database.
<http://www.kriptopolis.com/> - Kriptopolis e-zine
<http://www.lanzadera.com/haygentepato> - Hay gente pa to ...
<http://www.l0pht.com> - @tstake company
<http://www.l3h.8k.com> - Los 3 hermanos (underground)
<http://www.lorsomer.com/> - Lorsomer (underground)
<http://www.megasecurity.org/> - MegaSecurity
<http://www.mp3glowe.com/defson/hacking.htm> - DeFSon
<http://www.neohapsis.com> - Neohapsis: defending the digital horizon
<http://www.netsec.net/> - NetSec company
<http://www.networkcommand.com/> - Network Command
<http://www.networkintrusion.co.uk/> - Talisker's Network Security Tools
<http://www.newhackcity.net> - New Hack City
<http://www.nmrc.org/> - Nomad mobile research center
<http://www.nmrc.org/files/unix/> - NMRC: Unix security
<http://www.nsa.gov/> - National Security Agency
<http://www.ntbugtraq.com/> - NT BugTraq
<http://www.ntsecurity.net> - Windows IT security
<http://www.openwall.com> - Openwall project
<http://www.packetstormsecurity.org> - Packet Storm
<http://www.phrack.org/> - Phrack (underground)
<http://www.phreak.org/> - Phreaking y Hacking (underground)
<http://www.r00t.org/> - ROOT
<http://www.reefteam.com.ar/> - Reefteam (underground)
<http://www.robertgraham.com/> - Robert Graham Web site
<http://www.rootaccount.net/> - RootAccount
<http://www.rootshell.com> - RootShell
<http://www.rootshell.com/beta/documentation.html> - Documentación
<http://www.rootshell.com/beta/exploits.html> - Exploits
<http://www.root-core.com> - Root Core Network
<http://www.sans.org/> - SANS Institute online
<http://www.sans.org/tools.htm> - Roadmap to Security Tools and Services Online
<http://www.sans.org/topten.html> - How to Eliminate The Ten Most Critical Internet Security Threads
<http://www.sec33.com> - Sec33
<http://www.securecomputing.com/> - Secure Computing company
<http://www.securiteam.com> - SecuriTeam
<http://www.securityfocus.com> - SecurityFocus – BugTraq
<http://www.securitynewsportal.com/index.shtml> - Portal for Security News
<http://www.securitysearch.net> (developed by <http://www.shake.net>)
<http://www.securitytracker.com/> - SecurityTracker
<http://www.sli.net.nz/> - Southern Lights Inc. (Download security programs: exploits ...)
<http://www.sys-security.com> - Sys-security group
<http://www.technotronic.com/rhino9/> - Rhino9 Security Group (antiguo <http://rhino9.ml.org/>)
<http://www.tlsecurity.net> - GRC sucks
<http://www.totse.com/en/hack/> - Underground
<http://www.tracking-hackers.com> - Honeypots dedicated web site (L.Spitzner)
<http://www.tribal-2002.cjb.net/> - Tribal Team (underground)

<http://www.tryc.on.ca/security.html> - BoS: Best Of Security Archives
<http://www.ubizen.com/> - Ubizen company
<http://www.undersec.com/> - UnderSec Security Team
<http://www.ussrback.com/> - USSR
<http://www.vanhackez.com/> - VanHackez
<http://www.whitehats.com/> - WhiteHats
<http://xforce.iss.net> - Internet Security Systems (Computer Thrats & Vulnerabilities)
<http://xforce.iss.net/alerts> - Alertas de seguridad

11.2 Portales de seguridad de los principales fabricantes

Cisco:

- SAFE:
<http://www.cisco.com/go/safe/>
- PSIRT: Cisco Product Security Incident Response:
http://www.cisco.com/warp/public/707/sec_incident_response.shtml
- Cisco Secure Encyclopedia:
<http://www.cisco.com/warp/public/784/packet/jan01/index.html>
<http://www.cisco.com/cgi-bin/front.x/csec/csecHome.pl>
- Cisco Internet Security Advisories:
<http://www.cisco.com/warp/public/707/advisory.html>
- Cisco Feature Navigator:
<http://www.cisco.com/go/fn/>
- Security Notices:
<http://www.cisco.com/warp/public/707/notices.html>
- Cisco Secure Consulting Services:
<http://www.cisco.com/go/securityconsulting>
- Cisco Vulnerability Statistics Report:
http://www.cisco.com/warp/public/778/security/vuln_stats_02-03-00.html
- Networking Professional Connection:
<http://www.cisco.com/go/netpro>

<http://www.cisco.com/go/security>
<http://www.cisco.com/warp/public/779/largeent/issues/security>
<http://www.cisco.com/cgi-bin/front.x/csec/csecHome.pl>

Microsoft:

<http://www.microsoft.com/technet/security>
<http://www.microsoft.com/technet/security/current.asp> - Microsoft Security bulletin page
<http://www.microsoft.com/technet/security/current.asp> - Security bulletins
<http://www.microsoft.com/security>
<http://www.microsoft.com/technet/mpsa/start.asp> - Microsoft Personal Security Advisor (MPSA)
<http://support.microsoft.com/support/kb/articles/q303/2/15.asp?id=303215&sd=tech> - [HFNetChk](#): web

Linux:

<http://www.linuxsecurity.com>

<http://security.debian.org> - Debian

<http://suse.de/de/support/security/> - S.U.S.E

<http://www.redhat.com/support/errata/> - Security Advisories - RedHat

<http://www.freebsd.org/security/> - FreeBSD

HP:

<http://itrc.hp.com> :

“Maintenance and Support Menu” - “more...” - “Notifications” - “Support Information Digests”

- The security patch matrix:

ftp://ftp.itrc.hp.com/export/patches/hp-ux_patch_matrix

11.3 Revistas de seguridad o *e-zines*

- Information Security Magazine:
<http://www.infosecuritymag.com>
- Hackemate: (underground)
<http://hackemate.com.ar/ezines/>
<http://hackemate.com.ar/ezines/zines/>
- EKO magazine:
<http://www.ezkracho.com.ar/>
- NetSearch:
<http://www.netsearch-ezine.com/>
- Phrack:
<http://phrack.org/>

11.4 Grupos de noticias

comp.security
comp.security.*
comp..security.announce
comp.security.unix
alt.security
alt.security.*
alt.2600
linux.security.alert

11.5 Listas de correo

E-Listas: <http://www.elistas.net/>

- Crackindex: http://www.hackindex.org/lista_cra.htm

- Hackindex: http://www.hackindex.org/lista_hi.htm

SecurityFocus: <http://www.securityfocus.com>

- Pen-test: Penetration tests

- Vuln-dev: Vulnerabilities

11.6 Virus

<http://www.alertaantivirus.es/>
<http://www.antivirus.com/>
<http://www.icsalabs.com>
<http://www.wildlist.org>
<http://wtc.trendmicro.com/wtc/>

- Certificaciones de Antivirus
- Lista de virus de mayor difusión (ITW – “In The Wild”)
- Trend World Virus Tracking Center

Hoax (bulos en la red):

<http://www.antivirus.com/vinfo/hoaxes/hoax.asp>

Fabricantes antivirus:

<http://www.kaspersky.com>
<http://www.cai.com>
<http://www.f-secure.com>
<http://www.mcafee.com>
<http://www.norman.no>
<http://www.pandasoftware.es>
<http://www.sophos.com>
<http://www.symantec.com>
<http://www.trendmicro.com>

- AVP (Kaspersky)
- Inoculate IT (Computer Associates)
- F-Secure
- VirusScan (McAfee)
- Norman
- Panda
- Sophos
- Norton antivirus (Symantec)
- PC-cillin (Trend Micro)

12 ANEXO II: HERRAMIENTAS DE SEGURIDAD

Este ANEXO engloba referencias en Internet de software de seguridad:

- Tablas de clasificación de herramientas de seguridad relacionadas con los diferentes tipos de ataques mencionados.
Existe una primera tabla compuesta por todas aquellas herramientas de pequeño tamaño (pero no por ello poco importantes) existentes en Internet que permiten poner en práctica las tecnologías mencionadas.
Asimismo se ha incluido otra tabla de herramientas de seguridad más complejas y extendidas, así como fáciles de encontrar en Internet, con referencias presentadas en este trabajo y también otras no estudiadas en este trabajo por estar asociadas a la seguridad de los sistemas y no de la red.
- Repositorios en Internet de herramientas y utilidades de seguridad.
- Enlaces a repositorios de *exploits*.
- Se dispone de otros enlaces, como motores de búsqueda y repositorios de código fuente...

12.1 Tablas de herramientas de seguridad

Esta tabla contiene programas y herramientas sencillas y gratuitas (ninguna es comercial) existentes en Internet, tanto para la realización de ataques contra la seguridad como para defenderse de éstos. Mediante las mismas es posible obtener un conocimiento práctico avanzado de todos los conceptos y técnicas mencionados.

<i>Herramienta</i>	<i>Ataques</i>	<i>Lenguaje Prog.</i>	<i>SSOO</i>	<i>Notas</i>
ACKCmd	Trojano: covert channel (ACKs)	C	Win	S
blast	TCP stress tool	C	Win	S
chroot	Salir entorno chroot	C	Unix	S
clog	TCP SYN logger	C	Unix	S
dds	Detector de agentes DDoS	C	Unix	S
dsniff	Man-in-the-Middle attack	C	Linux	S: monkey.org
firehole	Ataque a firewalls personales	C	Win	S: keir.net
fport	Puertos-procesos abiertos	C	Win	S, ~lsof
frag	ICMP fragments generator	C	Linux	S
hunt	TCP hijacking	C	Linux	S
icmpush	ICMP packets generator	C	Linux	S
juggernaut	TCP hijacking	C	Linux	S
lanportscan	Escáner de puertos	C	Win	S
leaktest	Ataque a firewalls personales	C	Win	S: grc.com
lps	Local Port Scanner	C	Win	S
lsof	List open files (puertos-procesos)	C	Unix	S
nc	NetCat: network listener	C	Unix, Win	S: freshmeat.net
neptune	SYN Flood	C	Linux	S, 2.2 y 2.4
netbrute	Netbrute Scanner Security Suite	C	Win	S
mod_icmp	Proxy for ICMP echo request/reply	C	Linux	S
out	Man-in-the-Middle attack	C, Perl	Linux	S
queso	OS fingerprinting	C	Linux	S

raccess	Escáner de exploits	C	Linux	S: freshmeat.net
rat	Router Audit Tool (Cisco)	Perl	Unix	S
SMBScanner	Escaner ping y SMB (netBIOS)	C	Win	S
SNMPPing	Escáner SNMP	C	Win	S
snmpwalk	Utilidad acceso a SNMP (conjunto)	C	Win, Unix	S
snsn	Escáner SNMP	C	Win	S
spoofit	Generador paquetes TCP,UDP	C	Linux	S
superscan	Escáner de puertos	C	Win	S
synk4	SYN Flood	C	Linux	S, 2.2 y 2.4
synlog	TCP SYN logger	C	Unix	S
synsniff	TCP SYN logger	Perl	Unix	S
SYNWatch	TCP SYN logger	C	Unix	S
tocsin	TCP SYN logger	C	Unix	S
tunnelshell	Covert channel varios paquetes	C	Linux	S
udpflood	Generador paquetes UDP	C	Win	S
udpsniff	UDP bind shell channel	C	Linux	S
urk	Universal Root Kit	C	Unix	S

Nomenclatura:

- Una “S” en “**Notas**” significa que se dispone de la herramienta en el repositorio asociado al presente documento [*Si no dispones del repositorio, busca la herramienta en Internet ;-)*].

Esta segunda tabla contiene herramientas más complejas, que tienen asociados desarrollos más elaborados, disponibles ampliamente en Internet y de conocida relevancia.

Herramienta	Ataques	Lenguaje Prog.	SSOO	Notas
aide	Integridad sistema de ficheros	C	Unix	L, ~Tripwire
cops	Paquete de herram. de auditoría	C,Perl	Unix	fish.com/cops/
crack	Obtención passwords	C	Unix	L
ethereal	Sniffer gráfico	C	Unix	ethereal.com
ettercap	sniffer/interceptor/logger	RPM	Linux	sourceforge.net
ipchains	Módulo de firewall	C	Linux	2.2, L
iptables	Módulo de firewall	C	Linux	2.4, iptables.org
john the ripper	Obtención passwords	C	Unix, Win	L
LanGuard netscan	Escáner de vulnerabilidades	C	Win	C: gfi.com
nessus	Escáner de vulnerabilidades	C	Linux	nessus.org
Network Monitor	Sniffer	C	Win	C: Microsoft
nmap	Escáner de puertos...	C	Unix	insecure.org/nmap
putty	Cliente SSH	C	Win	openssh.org
saint	Escáner de vulnerabilidades	C	Unix	L
sara	Escáner de vulnerabilidades	C	Unix	L
satan	Escáner de vulnerabilidades	C	Unix	L
Sniffer Pro	Sniffer	C	Win	C: NAT
Snort	IDS	C	Unix	snort.org
SocksCapV2	Cliente SOCKS	C	Win	C: socks.nec.com
SSH	Protocolo SSH: cliente-servidor	C	Unix	C: openssh.org
SSS	Shadow Security Scanner	C	Win	L
sudo	Ejecución tareas como root	C	Unix	L

tara	Escáner de vulnerabilidades	C	Unix	L
tiger	Escáner de vulnerabilidades	C	Unix	L
tcpdump	Sniffer	C	Unix	tcpdump.org
tripwire	Integridad sistema de ficheros	C	Varios	C: tripwire.org,.com
vision	Info sistema: lsof, app, procesos...	C	Win	foundstone.com
winfingerprint	OS fingerprinting	C	Win	L

Nomenclatura:

- Una “C” en “Notas” indica que existe una versión comercial de la herramienta, además de la gratuita u *open-source*.
- Una “L” en “Notas” indica que en la sección posterior se dispone del enlace a la herramienta. En los enlaces de la tabla se ha omitido el prefijo “www”.

Direcciones Web:

aide:	www.cs.tut.fi/~rammer/aide-0.7.tar.gz
crack:	ftp://coast.cs.purdue.edu/pub/tools/unix/pwdutils/crack/
ipchains:	netfilter.samba.org/ipchains/
iptables:	www.netfilter.org
john the ripper:	www.openwall.com/john/
saint:	www.wwdsi.com/saint/
sara:	www-arc.com/sara/
satan:	www.fish.com/satan/
SSS:	www.networkingfiles.com/SecurityApps/Shadow.htm
sudo:	www.courtesan.com/sudo/
tara:	www-arc.com/tara/
tiger:	www.net.tamu.edu/network/tools/tiger.html
winfingerprint:	sourceforge.net/projects/winfingerprint/

12.2 Repositorios con herramientas y utilidades de seguridad

- Network Security Tools:
<http://ciac.llnl.gov/ciac/ToolsUnixNetSec.html>
- **Top 50** Security Tools:
<http://www.insecure.org/tools.html>
- Identifying tools that aid in detecting signs of intrusion:
<http://www.cert.org/security-improvement/implementations/i042.07.html>
- **SANS** Tools list:
http://www.sans.org/infosecFAQ/tools/tools_list.htm
- Navegar de forma anónima por la red:
<http://www.megaproxy.com>
- **CiSecurity** audit. Tools: Cisco, Unix, Windows...
<http://www.cisecurity.org/>
- **Razor** Tools:
<http://razor.bindview.com/tools/index.shtml>
- **IdeaHamster** Tools:
<http://www.ideahamster.org/tools.htm>
- **COAST** repository: CERIAS Security FTP Archive
<ftp://coast.cs.purdue.edu>
<ftp://coast.cs.purdue.edu/pub/tools/unix/>

- **Technotronic:** Security Information FTP Archive
<ftp://ftp.technotronic.com>
- **@stake:**
<http://www.atstake.com/research/tools/index.html>
- **YoLinux:**
<http://www.yolinux.com/TUTORIALS/LinuxSecurityTools.html>
- **COTSE:**
<http://wetelephant.cotse.com/tools/>
- **GNUSEC:**
<http://www.gnusec.com/resource/security-stuff/>

Incident Response Tools:

- General:
<http://www.incident-response.org>
<http://www.incident-response.org/tools.html>
- Windows:
<http://www.incident-response.org/windows.htm>
- Unix:
<http://www.incident-response.org/unix.html>
<http://www.weihenstephan.de/~syring/win32/UnxUtils.html>
- Recovering from an incident (CERT):
<http://www.cert.org/nav/recovering.html>

12.3 Repositorios de *exploits* (*exploits archives*)

- SecuriTeam:
<http://www.securiteam.com/exploits/archive.html>
- Hackemate:
<http://www.hackemate.com.ar/exploits/>
- Fyodor's Exploit World:
<http://www.insecure.org/sploits.html>
- Opennet:
<http://security.opennet.ru/base/exploits/>
- Antionline:
<http://www.antionline.com/directory/Computers/Hacking/Exploits/index.html>
- Razor:
<http://www.nmrc.org/files/index.html>
- Xhack:
<http://www.xhack.ch/index.php3?sub=exploits&link=Archive>
- Sec33:
<http://www.sec33.com/modules.php?name=Downloads>

12.4 Otros enlaces

Buscadores genéricos:

<http://www.teoma.com/>

<http://www.google.com>

<http://www.copernic.com>

Security Search Engine: (Searchsecurity y Google)

<http://searchsecurity.techtarget.com/integratedSearchAdvanced/>

Buscadores de seguridad y HCPV

<http://www.galeon.com/lacovacha/hk/buscadores.htm>

Buscador *cracks underground*:

<http://www.astalavista.com>

Códigos fuentes (y binarios):

<http://www.planet-source-code.com>

<http://sourceforge.net/>

<http://freshmeat.net/>

<http://www.codeguru.com/>

Defacements in the web:

<http://www.attrition.org/security/commentary/>

13 GNU FREE DOCUMENTATION LICENSE

GNU Free Documentation License
Version 1.1, March 2000

Copyright (C) 2000 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit

reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. In any section entitled "Acknowledgements" or "Dedications",

preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of

following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (c)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.1
or any later version published by the Free Software Foundation;
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.