Tema 3. Seguridad en red

Administración de Sistemas y Redes II

Julio López Albín

xulio@dec.usc.es

Índice

1. Tipología de los ataques de red

Los **ataques de red** son un problema común hoy en día en Internet. Estos son realizados por diversos motivos, tales como *desafíos*, intereses económicos o fraude. Los ataques son realizados para comprometer la confidencialidad, integridad y disponibilidad de los recursos en red. Podemos distinguir las siguientes categorías:

- Alteración o eliminación de información privada
- Repudio de un evento ya ocurrido
- Ataque de Denegación de Servicio (**DoS**), mediante los que la disponibilidad de recursos o servicios de red se ve comprometida.
- Acceso no autorizado a un sistema.

Las pérdidas en empresas causadas por cualquier tipo de ataque, pueden ser muy cuantiosas: pérdida y robo de datos, tiempo de caída y recuperación del sistema, secuestro de ordenadores y pérdida de reputación

1.1. Prácticas: Entorno Simulado

Para nuestro entorno de pruebas vamos a suponer al menos 4 máquinas:

Nombre	FQDN	IP	MAC
cliente1	cliente1.qemunet	192.168.0.101	`00:20:21:00:00:B1`
cliente2	cliente2.qemunet	192.168.0.102	`00:20:21:00:00:B2`
pasarela	pasarela.qemunet	192.168.0.1	`08:00:0B:01:01:01`
	ppublica.qemunet	dhcp	`00:00:7E:0E:0E:0E`
atacante	atacante.qemunet	192.168.0.200	`02:AA:3C:00:AA:3C`

- Todas ellas se econtrarán conectadas mediante una red local simulada en qemu a través de una conexión tcp.
- La pasarela ofrecerá conexión a internet a todos los demás, estando conectada la interfaz *ppublica.qemunet* en espacio de usuario.
- Recordar que para renombrar las imágenes, se deberían modificar los archivos `/etc/hostname` (con el nombre de la máquina) y `/etc/hosts` (con los datos sobre todas los hosts de la red).
- Para obtener conexión a internet a través de la pasarela, descomentaremos en la pasarela en el archivo `/etc/sysctl.conf` la siguiente línea:

  ```
  net.ipv4.conf.default.forwarding=1
  ```

- Para cargar el cambio de la configuración sin reiniciar, ejecutar:

  ```
  sysctl -p
  ```

1.1.1. netcat

`netcat` es una utilidad simple y versátil que permite escribir datos por red tanto por tcp como udp. Es una herramienta muy útil para testear conexiónes y generar tráfico para análisis de redes.

La sintaxis básica para establecer una conexión es:

`nc [-u]` *`host_remoto puerto_remoto`* `[-p` *`puerto_local`*`]`

La opción `-u` permite cambiar su protocolo por defecto , tcp, por el protocolo udp.

`netcat` también puede actuar como servidor, en cuyo caso su sintaxis básica es:

`nc [-u] -l -p` *`puerto_local`* `[`*`host_remoto puerto_remoto`*`]`

donde el *puerto local* indica en que puerto permanecerá escuchando, y tanto el *host como puerto remotos* limitan la dirección de origen del cliente.

Ejemplo de conexión simple:

cliente1 $ `nc -l -p 1052`

cliente2 $ `nc cliente1 1052`

Otras opciones que resultan especialmente interesante de `netcat` son:

-e *ejecutable* Permite especificar un ejecutable al que será rederigida toda la salida recibida por netcat. Por ejemplo, una sencilla implementación de telnet:

cliente1 $ `nc -l -p 1052 -e /bin/bash`

-L sustituye a `-l` para permitir que al acabar una conexión, netcat vuelva a permanecer en escucha.

Se puede usar tuberías del sistema para redirigir los datos hacia la conexión en red. Por ejemplo, para enviar un paquete udp con un string podríamos hacer:

Prácticas :

1. Simular un chat entre los puertos 2000 del cliente1 y cliente2.
2. Levantar una instancia de netcat que sólo escuche las conexiónes tcp al puerto 2020 en la pasarela desde el cliente1.
3. Desafío: realizar una conexión "a un telnet" de netcat en el cliente1, que permita desde fuera del qemu enviar comando a dentro de él, sin cambiar la configuración del qemu.

2. Escuchas de Red: Sniffers

Las escuchas de red son ataques realmente peligrosos, puesto que pueden obtener información compromentida y son difícilmente detectables.

- Las herramientas que permiten obtener estos datos son los sniffers y los analizadores de protocolos.

- Un sniffer o rastreador de red es un proceso que olfatea el tráfico que se genera en la red a nivel de enlace; de este modo puede leer toda la información que circule por el tramo (segmento) de red en el que se encuentre.
- Un analizador de protocolos es un sniffer que ha extendido su funcionalidad para comprender ciertos protocolos y permiten analizar la información contenida en los paquetes enviados por la red.
- Un ordenador conectado a una red mediante un hub puede ver el tráfico de toda la red poniendo su tarjeta en modo promiscuo y analizarlo con programas como `tcpdump`, `dsniff`, `wireshark`, `ettercap`, ...

2.1. tcpdump

`tcpdump` es una programa para el análisis de tráfico en la red, usando la librería *libpcap*.

> `tcpdump [-np] [-c` *`paquetes`*`] [-i` *`interfaz`*`] [-s` *`bytes`*`] [-w` *`archivo`*`] [-F` *`archivoFiltros`*`] [`*`expresion`*`]`

- Por defecto, captura todos los paquetes hasta que pulsemos CTRL+C o si añadimos la opción `-c` *`paquetes`*, hasta que ese número de paquetes sea capturado.
- Se puede definir filtros para seleccionar que tráfico capturar, o bien con una *expresión* en línea de comandos, o desde un fichero con la opción `-F`.
- Necesita ejecutarse como root o con el setuid de root.
- Por defecto, tcpdump escucha la primera interfaz configurada. La opción `-i` permite especificar una distinta.
- `-n` no convierte direcciones (ip, puertos, ..) a nombres.
- `-p` no cambia la interfaz a modo promiscuo.
- `-s` permite variar la cantidad de bytes capturados (por defecto 68). Con el valor 0 captura todo el paquete.
- `-w` permite guardar la salida en un fichero.

La *expresión* usada para definir un filtro se compone de un conjunto de primitivas, cada una de las cuales puede tener alguno de los tres tipos de modificadores siguientes:

tipo Las opciones más comunes son `host` (una máquina: host 192.168.0.1), `net` (una red net 192.168.0.0/34), `port` (un puerto concreto: port 22) y `portrange` (un rango de puertos: portrange 8000-8100). El modificador por defecto es `host`.

dir Especifica desde o hacia donde se va a dirigir el flujo de datos, siendo los más comunes: `src`, `dst`, `src and dst` y `src or dst`, siendo la última el valor por omisión.

proto Se puede especificar el protocolo a capturar o por defecto en los que tengan sentido la restricción. Los más comunes son: `ether`, `ip`, `tcp`, `udp` y `arp`.

Se pueden combinar primitivas con los operadores `and`, `or` y `not`, además de poder agrupar y cambiar precedencias con paréntesis..

Ejemplos:

tcpdump dst host 192.168.0.101 Captura el tráfico cuyo destino tenga la IP 192.168.0.101

tcpdump host 192.168.0.101 Captura el tráfico cuyo origen o destino tenga la IP 192.168.0.101

tcpdump ether src 00:20:21:00:00:B2 Captura el tráfico cuyo origen tiene la mac 00:20:21:00:00:B2

tcpdump tcp port 80 Captura todo el tráfico tcp por el puerto 80.

tcpdump portrange 21-23 Captura todo el tráfico con origen o destino de los puertos 21, 22 o 23.

tcpdump tcp and not port 80 Captura todo el tráfico tcp que no viaje a través del puerto 80.

Otras opciones interesantes para realizar filtros son:

gateway *máquina* Captura aquellos paquetes que van a ser redirigidos por la *máquina*. Internamente, son aquellos que tienen la dirección ethernet destino la de la*máquina*, pero su dirección IP destino no lo es.

less *longitud* establece un tamaño máximo para los paquetes a caturar.

greater *longitud* análogamente define un tamaño mínimo para los paquetes.

ip proto *protocolo* selecciona como paquetes a analizar aquellos cuyo protocolo está definido por *protocolo*, donde puede tomar valores como `icmp`, `tcp` o `udp`, de los que hay definidos alias que no requieren esta sintaxis compleja (realmente el alias encapsula ip e ip6).

[ether] broadcast válido si la trama esta dirigida a la dirección de difusión ethernet.

ip broadcast que es cierto cuando el paquete va dirigido a la dirección de difusión IP.

icmp[icmptype]=*tipo* permite seleccionar el tráfico icmp según su tipo, donde los valores válidos son su código numérico o los siguientes valores: icmp-echoreply, icmp-unreach, icmp-sourcequench, icmp-redirect, icmp-echo, icmp-routeradvert, icmp-routersolicit, icmp-timxceed, icmp-paramprob, icmp-tstamp, icmp-tstampreply, icmp-ireq, icmp-ireqreply, icmp-maskreq e icmp-maskreply.

tcp[tcpflags]=*flags* permite seleccionar paquetes tcp según las flags activas: tcp-fin, tcp-syn, tcp-rst, tcp-push, tcp-ack, tcp-urg.

Ejemplo: Paquetes que van a ser redirigidos por la pasarela.

```
tcpdump -n gateway pasarela
```

o

```
tcpdump -n ether host 08:00:0B:01:01:01 and not host 192.168.0.1
```

Prácticas de filtros tcpdump

1. Obtener todas las peticiones http.
2. Filtrar sólo el tráfico interno.
3. Establecer un chat entre dos máquinas a traves de los puertos 2000 y 3000 respectivamente con netcat, y definir un filtro de forma que sólo se obtenga todo el tráfico generado por esta comunicación.
4. Obtener los paquetes de broadcast de la red local, que no provenga de la propia red.
5. Obtener un listado del tráfico de inicio de conexión.

6. Observar con un filtro lo más reducido posible el tráfico generado por un conjunto de pings a distintos ordenadores y el tráfico arp que aparece en la red.

7. Obtener el tráfico generado por los pings mayores de 500 bytes.

2.2. dsniff

dsniff es un conjunto de utilidades que permiten realizar escuchas y auditorias dentro de una red.

- `dsniff` es un password sniffer que puede manejar los protocolos FTP, Telnet, SMTP, HTTP, POP, IMAP, NFS, X11, CVS, IRC, AIM, ICQ, PostgreSQL, Oracle SQL*Net, Sybase y Microsoft SQL.

  `dsniff [-n] [-i` *`interface`*`] [`*`filtro`*`]`

- `urlsnarf` es una herramienta que permite examinar las peticiones de paginas web que existen en la red.

  `urlsnarf [-n] [-i` *`interface`*`] [[-v]` *`patrón`* `[`*`filtro`*`]]`

- `mailsnarf` monitorea el tráfico de correo a través de los protocolos POP y SMTP.

  `mailsnarf [-i` *`interface`*`] [[-v]` *`patrón`* `[`*`filtro`*`]]`

- `filesnarf` permite obtener un copia de un archivo seleccionado de un sistema NFS.

  `filesnarf [-i` *`interface`*`] [[-v]` *`patrón`* `[`*`filtro`*`]]`

Parámetros de las herramientas dsniff:

- `-n` No utiliza resolución inversa de DNS.
- `-i` Permite seleccionar la interfaz de red.
- *patron* Sólo se guardaran las direcciones, correos o archivos que cumplan ese patrón.
- `-v` El patrón de búsqueda se aplicará de forma inversa.
- *filtro* Se especifica una filtro de tcpdump para seleccionar los paquetes a analizar.

Prácticas

- Obtener password de una conexión ftp al exterior y una telnet en el interior.
- Observar las páginas visitadas por un usuario en la red. Hacer un filtro para restringir a las paginas de la usc y accedidas sólo desde uno de los clientes.

2.3. wireshark

`wireshark` es un programa similar a tcpdump y basado también en la librería pcap, pero pensado para entorno gráfico.

- Puede trabajar tanto con captura de paquetes como con un fichero previamente capturado, incluso en otro sistema.
- Permite inspeccionar las tramas así como analizar las diferentes conexiónes.
- Permite restringir los paquetes a capturar con filtros similares a tcpdump.
- Una vez finalizada la captura, también se pueden aplicar filtros para el análisis y visualización.
- En el entorno gráfico muestra cada uno de los paquetes, y se pueden observar las distintas cabeceras de las distintas capas de red.
- Permite el seguimiento de conexiónes TCP.
- Permite visualizar los datos contenidos en los paquetes capturados con distintos filtros (ASCII, binario, hexadecimal, ...).

Prácticas

- Obtener pasword de una conexión http no encriptada al exterior, y comprobar que no es visible si está encriptada.
- Tomar una observación prolongada, generando tráfico ftp, telnet, http, icmp, etc... Observar que se muestran los passwords en claro y observar las capacidades para obtener datos estadisticos por parte de `wireshark`.

2.4. Detección de sniffers

La detección de sniffers en redes conmutadas es difícil. Existen distintos tests de red que permiten detectar tarjetas en modo promiscuo, pero non son totalmente fiables: .

Test DNS Generar paquetes falsos, y analizar si se realizan peticiones DNS inversas a las direcciones IP de esos paquetes.

Test Ping Generar paquetes echo-request con la dirección IP correcta y la mac falsa esperando respuestas.

Test Ping de latencia Incrementar el tráfico de nuestra red con conexiónes TCP falsas, y analizar el tiempo de latencia a un ping de las distintas máquinas de la red con y sin tráfico.

Test ARP Realizar una petición arp con la IP real, pero una mac falsa y esperar respuestas.

En el caso de que tengamos acceso a la máquina, un análisis directo de la interfaz con `ifconfig -a` si la interfaz nos puede mostrar si está en modo promiscuo.

2.4.1. sniffdet

`sniffdet` es un programa que realiza test de detección remota de interfaces en modo promiscuo.

`sniffdet [-i` *`interface`*`] -t` *`test target`*

- Los parámetros de test de detección validos son: dns, icmp, latency y arp.
- Otras opciones interesantes son:

  -c Nos permite especificar otro archivo de configuración para los test.

  -l Para especificar el archivo donde guardar los resultados del test.

  -f Para proporcionar un archivo con host a analizar.

  -p Plugin para formatear la salida.

Para instalar este programa nos debemos descargar la última versión del svn. Para poder compilarlo, necesitais primero instalar `autoconf`, `automake`, `libtool`, `libnet0-dev`, `libltdl3-dev` y `libpcap-dev`. Luego, para compilar el programa:

```
autoreconf -ifs
./configure
make
make install
```

Prácticas de detección de sniffers

- Comprobar que al activar el `tcpdump`, la tarjeta se pone en modo promiscuo.
- Hacer detección de interfaces en modo promiscuo, hacia una tarjeta en este modo, y otra no en ello.
- Comprobar la diferencia de usar `tcpdump -n` para el test DNS.

2.5. Protección contra escuchas

- Una solución para evitar esta técnia consiste en la segmentación de la red y de los equipos mediante switch. Con ello, el único tráfico que sería visible por las máquinas sería el destinado a ellas, puesto que el conmutador sólo envía los paquetes destinados a su dirección MAC. Aún así se debe ser precavido, porque existen técnicas más agresivas que permiten continuar realizando el sniffing, como el ARP spoofing.
- La mejor defensa contra los sniffers es el uso de sesiones cifradas; aunque si no se vigilan de forma adecuada contraseñas y certificados, también existen técincas que quebrantan en gran medida la seguridad de los protocolos encriptados más comunes.

3. Ataques de suplantación (Spoofing)

- Es Spoofing se refiere al uso de técnicas de suplantación de identidad para obtener acceso a recursos en red mediante la falsificación de paquetes.
- Podemos dividir las diversas técnicas de spoofing según la tecnología en la que se usan: IP Spoofing, ARP spoofing y DNS spoofing entre otros.

3.1. IP Spoofing

- Consiste básicamente en sustituir la dirección IP origen de un paquete TCP/IP por otra dirección IP a la cual se desea suplantar.
- Los programas que basan la autentificación en la dirección del cliente, son sensibles a este ataque. Por ejemplo, rsh.
- Distinguimos dos tipos de spoofing: *no blind* y *blind*.

3.1.1. No blind

Tenemos acceso al segmento local de la víctima suplantada, y podemos observar las repuestas del servidor.

- Para evitar interferencias, se suele realizar un ataque de denegación de servicio sobre la máquina a suplantar.
- Realizamos la petición de conexión con la dirección del host a suplantar.
- Analizando los paquetes enviados al host a suplantar, respondemos imitándolo.

Se trata de un ataque bastante sencillo de realizar tanto conceptual como técnicamente.

3.1.2. Blind

Es un ataque más complicado, ya que no podemos observar la respuesta del servidor hacia el host suplantado.

- En caso de usar la pila TCP, debemos mantener además el orden en los número de secuencia. Recordemos que en una conexión TCP, se usan números de secuencia para mantener el orden en los datos.
- Como no podemos observar las respuestas del servidor, debemos descubrir el número usado por éste.
- En ciertas implementaciones de la pila TCP, la generación de los números de secuencia es predecible, lo que nos permite realizar este ataque.

3.1.3. Protección ante IP Spoofing

- Para protegerse del IP spoofing la solución más eficaz es no utilizar autentificación basada en IP, puesto que hoy en día no ofrece ventajas sobre otras disponibles.
- Los router se deben configurar para que no se admitan paquetes desde redes externas con direcciones ips internas. Con ello se reduce el riesgo de ataques ciegos, aunque no se elimina.

3.2. ARP Spoofing

- El protocolo ARP es el encargado de traducir direcciones IP de 32 bits, a las correspondientes direcciones hardware, generalmente de 48 bits en dispositivos ethernet.
- Cuando un ordenador necesita resolver una dirección IP en una dirección MAC, se efectúa un arp-request a la dirección de broadcast de dicho segmento de red (FF:FF:FF:FF:FF:FF), solicitando que el equipo que tiene esta IP responda con su dirección MAC.
- Con objetivo de reducir el tráfico en la red, cada respuesta ARP(arp-reply) que llega a la tarjeta de red es almacenada en una tabla caché aunque la máquina no haya realizado la correspondiente petición. Este es el factor que permite realizar el ataque de suplantación ARP.
- El objetivo es envenenar la tabla ARP de una víctima y forzarla a que envíe los paquetes a un host atacante en lugar de hacerlo a su destino legítimo.
- Para ello, el atacante enviará arp-reply falsos, donde proclamará que su MAC es la asociada a una IP especifica, por ejemplo un router, con lo que la información dirigida al router pasaría por el atacante. Este podría sniffar y redirigir o modificar esta información.
- Esta técnica permite realizar escuchas en redes conmutadas.

3.2.1. arp

`arp` permite examinar y modificar la tabla de caché de arp.

`arp [-vn] [-i` *`interfaz`*`] -a [`*`host`*`]`

Esta orden permite examinar la tabla arp, o con el parámetro *host*, sólo una entrada.

`arp [-v] [-i` *`interfaz`*`] -s` *`host direccMac`*

Podemos asociar a una dirección IP una dirección de forma estática. Esta es la solución más eficiente para bloquear el arpsoofing.

`arp [-v] [-i` *`interfaz`*`] -d` *`host`*

Con este comando se puede elimiar una entrada de la caché de arp.

3.2.2. arpspoof

es una herramienta del paquete dsniff que permite realizar un arp spoofing.

`arpspoof [-i` *`interfaz`*`] [-t target] host`

`arpspoof arpspoof` hace creer a toda la red, o a un target concreto si lo especificamos, que nuestra mac corresponde al host especificado mediante la creación de respuestas ARP falsas.

En un ataque arp-spoofing con esta herramiente, los pasos fundamentales serían:

- Para no realizar una denegación de servicio al equipo auditado, debemos recoger su tráfico, analizarlo y devolverlo para que siga su camino. Para ello activaremos el ip forwording:

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

- Para poder recibir el tráfico entre el cliente1 y la pasarela, debemos hacer que ambas máquinas crean que somos la otra, es decir, debemos ejecutar ambos comandos:

  ```
  arpspoof -t 192.168.0.1 192.168.0.101
  arpspoof -t 192.168.0.101 192.168.0.1
  ```

- Podemos comprobar si el tráfico esta redirigido con `tcpdump -p` o mirar las tablas de cache arp para ver si se han realizado los cambios necesarios.

3.2.3. Contramedidas contra el arp-spoofing

Una de las aproximaciones más eficaces para evitar ataques ARP spoofing, es establecer la dirección MAC del router de forma estática.

Otra opción es usar la herramienta `arpwatch` que monitoriza la actividad de la red ethernet y mantiene una base de datos de pares de direcciones ip y mac. En caso de que se produzca un cambio, esta aplicación lo reporta por email, distinguiendo 4 avisos:

new activity El par MAC/IP ha sido usado por primera vez en los últimos seis meses.

new station La dirección MAC nunca ha sido usada.

flip flop La dirección MAC ha cambiado desde el último valor registrado al penúltimo.

reused old ethernet address La dirección MAC ha cambiado desde el último valor registrado hasta un valor anterior al penúltimo.

changed ethernet address La dirección IP ha cambiado a una nueva dirección MAC.

Para probar este servicio sin configurar el servidor de correo:

```
arpwatch -d
```

3.3. DNS Spoofing

- En el spoffing de DNS, el atacante pone en peligro el servidor de DNS modificando las tablas que relacionan direcciones IP y nombres de hosts.
- La modificación de la tabla se suele conseguir mediante alguna vulnerabilidad del software del servidor o por su confianza hacia servidores poco fiables. Las entradas falseadas de un servidor DNS son susceptibles de infectar (envenenar) el caché DNS de otros servidores.
- La forma más sencilla de detectar este ataque consiste en comparar los resultados proporcionados por el ordenador del que sospechamos con otros servidores.
- En general, la forma más eficaz para dificultar los ataques de spoofing consiste en el uso de protocolos cifrados para la comprobación de identidad entre servidores DNS (DNSSEC).

3.4. Ataques Man In The Middle

En criptografía, un ataque man-in-the-middle (MitM o intermediario, en castellano) es un ataque en el que el enemigo adquiere la capacidad de leer, insertar y modificar a voluntad, los mensajes entre dos partes sin que ninguna de ellas conozca que el enlace entre ellos ha sido violado. El atacante debe ser capaz de observar e interceptar mensajes entre las dos víctimas. El ataque

MitM es particularmente significativo en el protocolo original de intercambio de claves de Diffie-Hellman, cuando éste se emplea sin autenticación.

3.4.1. ettercap

`ettercap` es una herramienta diseñada para actuar como sniffer en una red communtada y que permite realizar una gran cantidad de ataques man in the midlle.

- Escuchas usando una o dos tarjetas (actuando como bridge).
- Injección y modificación de paquetes.
- Obtención de passwords.
- Soporta múltiples interfaces de usuario: texto(-T), ncurses(-C), demonio (-D) y gtk (-G).
- Soporte a filtros y plugins.
- Ataques man in the middle.
- Soporte para la disección de un gran número de protocolos, incluso cifrados.

`ettercap` esta diseñado para interceptar el tráfico entre dos objetivos y su sintaxis es:

```
ettercap [OPCIONES] [OBJETIVO1] [OBJETIVO2]
```

Para la especificación de cada objetivo se usa una expresión de la forma MAC/IPs/PUERTOs, donde se puede omitir cualquiera de las partes de la expresión. Tanto las IPs como los PUERTOs puede ser uno o varios, concatenándose valores con el operador `,` y se definiéndose rangos con el operador `-`.

Ejemplos de expresiones:

/192.168.0.100/

/192.168.0.100,192.168.0.105-7/

/192.168.0.100/21-23

Al iniciarse `ettercap` realiza un escaneo arp de la parte de la red que entre dentro de los objetivos (excepto que especifiquemos la opción `-z`). Con el objetivo // denotaremos toda nuestra subred.

ettercap -T /192.168.0.1/ realizaría un arp-request a la ip 192.168.0.1

ettercap -T /192.168.0.1/ // realizaría arp-requests a todas las ips de nuestra subred.

El tráfico analizado en una red no conmutada en ambos casos sería el mismo. Pero en una red conmutada si tuviesemos que combinarlo con Mitm solo sería posible en el segundo caso, donde especificarmos ambos objetivos.

Algunas de las opciones más comunes son:

- La opción `-q` en el modo texto (-T), permite un modo silencioso en el que no muestra el contenido de los paquetes, solo la información que obtiene sobre contraseñas.
- `-i` *`interfaz`* permite especificar al interfaz de red.
- `-p` no activa la tarjeta en modo promiscuo.
- `-u` situa a ettercap en modo no ofensivo. En este modo `ettercap` no redirige los paquetes que analiza, lo que permite ejecutar múltiples instancias sobre una maquina sin duplicar paquetes.
- `-P` *`plugin`* carga un plugin de ettercap. Se puede obtener los plugins disponibles con `-P list`.
- `-F` *`filtro`* carga un filtro compilado con `etterfilter` para modificar los paquetes redirigidos por ettercap.
- `-L` *`logfile`* guardar en formato binario todos los paquetes, asi como información sobre contraesñas y host.
- `-l` *`logfile`* guarda un fichero de log sin paquetes binarios.
- Los ataques man in the middle se especifican con la opción `-M` *`ataque`*.
  - `-M arp [remote]` nos permite redirigir el tráfico usando arp-spoofing. Debemos usar la opción `remote` si uno de los host implicados es un gateway y queremos obtener el tráfico con la red exterior.
  - `-M port` permite realizar Port Stealing sobre switch ethernet.

Ejemplos:

ettercap -Tzq Escucha passwords en una red no conmutada.

ettercap -T -M arp:remote /192.168.0.1/ /192.168.1.101-2/ Redirige usando arp-spoofing todo el tráfico de los cliente1 y cliente2 con la pasarela e internet.

Configuración segura para diseccionar protocolos encriptados `ettercap` una vez iniciada su conexión con el driver de la tarjeta de red, cambia el dueño de su proceso al usuario nobody para no comprometer la seguridad.

Para realizar la disección de los protocolos cifrados, necesita establecer una serie de reglas de redirección en el firewall. En debian deberíamos editar el archivo `/etc/etter.conf`, desconmentado las siguientes lineas:

```
#redir_command_on = "iptables -t nat -A PREROUTING -i %iface
   -p tcp --dport %port -j REDIRECT --to-port %rport"
#redir_command_off = "iptables -t nat -D PREROUTING -i %iface
   -p tcp --dport %port -j REDIRECT --to-port %rport"
```

además, en el comando que elimina las reglas de redirección, debemos ejecutarlo mediante sudo, siendo el resultado:

```
  redir_command_on = "iptables -t nat -A PREROUTING -i %iface
-p tcp --dport %port -j REDIRECT --to-port %rport"
  redir_command_off = "sudo iptables -t nat -D PREROUTING -i %iface
-p tcp --dport %port -j REDIRECT --to-port %rport"
```

Cuando se inicia ettercap, al ejecutarse como root, puede introducir las reglas; pero en el apagado se ejecuta bajo otro uid sin los permisos necesarios. Para poder realizar este proceso de forma segura, crearemos un usuario llamado *ettercap* que no podrá iniciar sesión en nuestra máquina, pero podrá eliminar reglas de esta tabla.

```
adduser -system -home /nonexistent -no-create-home ettercap
```

Damos permisos para eliminar reglas de redirección de nuestro firewall con la siguiente línea en el archivo `/etc/sudoers`

```
ettercap ALL=NOPASSWD:/sbin/iptables -t nat -D *
```

Como último paso asignaremos a la variable `ec_uid = 65534` el uid del usuario ettercap en el fichero `/etc/etter.conf`, con lo cual el proceso del `ettercap` se ejecutará bajo el usuario que hemos creado.

Prácticas

1. Realizar un ataque Mitm para obtener la contraseña de una cuenta de correo electrónico accediendo a ella mediante https.

2. Modificar código html en una conexión a un servidor web usando un filtro de ettercap, de forma que los enlaces a la UDC apunten a la USC. Para ello se deberá compilar el sigiente filtro con el programa `etterfilter`:

```
if (tcp.src == 80 && ip.proto == TCP ) {
  replace("www.udc.es","www.usc.es");
}
```

3. Realizar un dns-spoofing con el plugin `dns_spoof` de ettercap, de forma que las peticiones a la página web www.udc.es seas redirigidas a www.usc.es. La configuración de este plugin se encuentra en el archivo `/usr/share/ettercap/etter.dns`.

4. Ataque de denegación de servicio (DoS)

Un ataque de denegación de servicio (DoS) es cualquier acción, iniciada por una persona o por cualquiera otra causa, que incapacite el hardware, software o ambos, ya sea de un host o de hardware de red y que lleve a que no se pueda llegar a un sistema y se deniege un servicio de red.

packit

`packit` es una herramienta para la captura e inyección de paquetes, tanto a nivel ethernet como en la capas IP, así como en los protocolos TCP, UDP y ICMP.

La síntaxis resumida para la inyección de tráfico sería:

`packit -m inject [-t` *`protocolo`*`]` *`opciones`*

Las opciones más básicas son:

-t *protocol* Especifica el tipo de paquete a inyectar, permitiendo: TCP (por defecto), UDP, ICMP y ARP.

-c *número* Especifica el número de paquetes a inyectar y con el valor 0 no se detiene.

-w *intervalo* especifica el tiempo en segundo entre cada conjunto de paquetes (por defecto 1).

-b *número* determina el número de paquetes a inyectar en cada instante (de forma continua con el valor 0).

-s origen especifica la dirección IP de origen.

-sR usa una dirección IP de origen aleatoria.

-d destino especifica la dirección IP de destino.

-dR usa una dirección IP de destino aleatoria.

-S puerto especifica el puerto origen en los protocolos TCP y UDP.

-D puerto especifica el puerto destino en los protocolos TCP y UDP. Se puede usar un rango: 100-200.

-F flags permite especificar las flags TCP: S(SYN), F(FIN), A(ACK), P(PSH), U(URG), R(RST).

4.1. Buffer Overflow

Un servicio o aplicación sensible a un desbordamientos de memoria responde ante la recepción de más datos de los esperados con un error de ejecución que puede provocar la caída del servicio, o incluso una modificación en su flujo de ejecución normal.

- Los errores de ejecución son debidos a que el conjunto de datos recibidos es mayor que el espacio en memoria reservado, y si la aplicación no hace comprobación del tamaño de los datos se produce un desbordamiento.
- Los datos insertados pueden llegar a sobreescribir zonas de memoria relacionadas con el flujo de ejecución del programa, pudiendo esto provocar la ejecución de instrucciones incrustadas en los datos de entrada. Estas instrucciones pueden dar acceso a información comprometida o incluso porporcionar una shell.

4.2. Ataques de fragmentación

Los ataques de fragmentación atacan debilidades en el reensamblado de paquetes IP fragmentados con tamaños incorrectos.

4.2.1. Ping of death

El ping de la muerte es un ataque basado en un overflow muy conocido por tratarse de un ataque sencillo al que fueron sensibles un gran número de sistemas operativos.

- Un paquete IP esta limitado a $2^{16} - 1(65535)$ bytes (RFC-791) incluyendo la cabecera (generalmente 20 bytes).
- Los paquetes que superan el MTU de la red (en ethernet generalmente 1500) son fragmentados en paquetes más pequeños.

- Un ICMP echo-request es enviado dentro de un paquete IP, y tiene una cabecera de 8 bytes (RFC-792) seguida por un payload de datos especificado por el usuario (por defecto 64 bytes).
- El tamaño máximo permitido para los datos serían $65535 - 20 - 8 = 65507$.
- En este ataque se enviaban un conjunto de fragmentos que al reensamblarse superaban el tamaño máximo permitido produciendo un overflow de las variables internas de 16 bits, que según el sistema producía caídas o reinicios.
- Este ataque se hizo muy popular porque tanto Windows 95 como Windows NT permitían construir pings ilegales con la implementacion de ping integrada: `ping -s 65510` *`IP_Victima`*.
- Eran vulnerables a este ataque windows95, NT, Irix, Solaris, HP-UX, AIX, MacOS, Linux 2.0.23, ...

4.2.2. Teardrop

Este es otro ataque famoso de fragmentación IP al que eran sensibles kernel de linux previos al 2.0.32 y Windows 95/NT.

- El ataque consistía en enviar dos tramas IP fragmentadas a ensamblar en el destino, en el cual en el segundo fragmento enviado se establece un valor en el campo de desplazamiento del fragmento que cae dentro del bloque anterior.
- Al reensamblar los paquetes, el sistema consideraba el tamaño del paquete negativo, lo que provocaba un desbordamiento de memoria en el ensamblado, que producia la caida del sistema.
- Este ataque usaba un paquete udp, y generalmente se dirigia la puerto 53 (DNS) para poder atravesar más facilmente los cortafuegos.
- Existen diversas variaciones de este código, como NewTear, Bonk, Boink, SynDrop; que atacaban vulnerabilidades presentes en los primeros parches de seguridad.

4.3. IP Flooding

El ataque de *IP Flooding* se basan en la inundación másiva de la red mediante datagramas IP.

- Estos ataques se pueden utilizar para degradar el rendimiento de la red a la cual está conectado el atacante generando paquetes con origen y destino **aleatorio**.
- Además del degradado de la red, también pueden colapsar un equipo, con un ataque **dirigido** contra una víctima.
- Se puede magnificar usando la dirección de broadcast, llamado **Broadcast IP Flooding**. La forma más sencilla de este ataque reside en enviar datagramas IP a la dirección de broadcast de la red.

4.3.1. Smurf y Fraggle

Ambos ataques son tipos concreto de Broadcast IP Flooding.

- El ataque **smurf** usa paquetes ICMP de tipo echo-request con IP origen la de la máquina atacada, y con IP destino una dirección broadcast de la red local o las redes que se utilizaran para atacar a la víctima. Con ello, todos los intermediarios enviarán paquetes ICMP echo-reply magnificando el ancho de banda usado y ralentizando la red e incluso llegando a colapsar a la víctima.
- El ataque **fraggle** es muy similar al **smurf**, pero usando el protocolo UDP. En este caso el atacante enviará un mensaje UDP al puerto 7 (*echo*) con destino el puerto 19 (*chargen*) de la víctima. Con este ataque los host que tengan activo el servicio *echo* reenviarán el paquete a la víctima, y los que no mandarán un ICMP de error. Además, si la víctima tiene levantado el servicio chargen, entrará en una bucle infinito con los servidores con echo activo.

4.4. Ataques contra implementaciones la pila TCP/IP

4.4.1. SYN Flooding

Este ataque se basa en el control de sesiones del protocolo TCP. Para mantener las conexiones tcp se crean estructuras en memoria que almacena datos sobre la conexión.

- El atacante inicia múltiples conexiones enviando paquetes SYN, pero sin llegar a completar el establecimiento de la conexión con los paquetes ACK respuesta al desafío ACK/SYN de la víctima. Esto produce un llenado de la pila conexiones del host atacado, con lo cual no se pueden establecer nuevas conexiones.

- El servidor desestima las conexiones semi-abiertas al cabo de un tiempo, pero sí se mantiene el flujo de paquetes SYN sin respuesta, esto provocará que dicho puerto no pueda establecer nuevas conexiones.
- Para dificultar el rastreo del atacante, se suele combinar con un IP Spoofing, enviando los paquetes SYN con dirección origen falsa.

Práctica

1. Realizar un ataque SYN flooding a una maquina que tenga instalado telnet. Intentar conectarnos mientras se realiza el ataque. Comprobar que nos podemos conectar a otro servicio distinto. Podemos ver el estado de las conexiones con `netstat -n -p tcp`.

4.4.2. LAND

- El ataque LAND consiste en enviar un paquete TCP con el indicador SYN activo contra un puerto activo de la víctima, y con puerto y dirección origen los mismos que el destino.Este ataque puede conseguir que el ordenador se responda a sí mismo indefinidamente. En algunos sistemas puede llegar a colgarlo o apagarlos, y en otros subir el consumo de la CPU durante un tiempo y manteniendo un conjunto constante de SYN se puede realizar un ataque de denegacion de servicio.
- El ataque LaTierra es una modificación del anterior especifica para Windows NT, que es sensible a un conjunto de paquetes SYN, pero cambiando sucesivamente de puerto.

4.4.3. WinNuke

Es un antiguo ataque que simplemente al enviar un paquete TCP a un sistema Windows NT con el indicador URG puesto en la cabecera TCP al puerto 139 (NetBios), el sistema se quedará colgado y será necesario reiniciar.

4.5. Ataques de Denegación de servicio distribuidos: DDoS

Un ataque de denegación de servicio distribuido es aquel en que un conjunto de sistemas previamente comprometidos realizan un ataque de denegación de servicio sincronizado a un mismo objetivo. Al aunar los recursos de todos los sistemas comprometidos, se consigue un flujo que es capaz de saturar objetivos de gran potencia y gran ancho de banda.

Podemos distinguir dos tipos de ataques por la estructura usada:

4.5.1. Ataques preprogramados

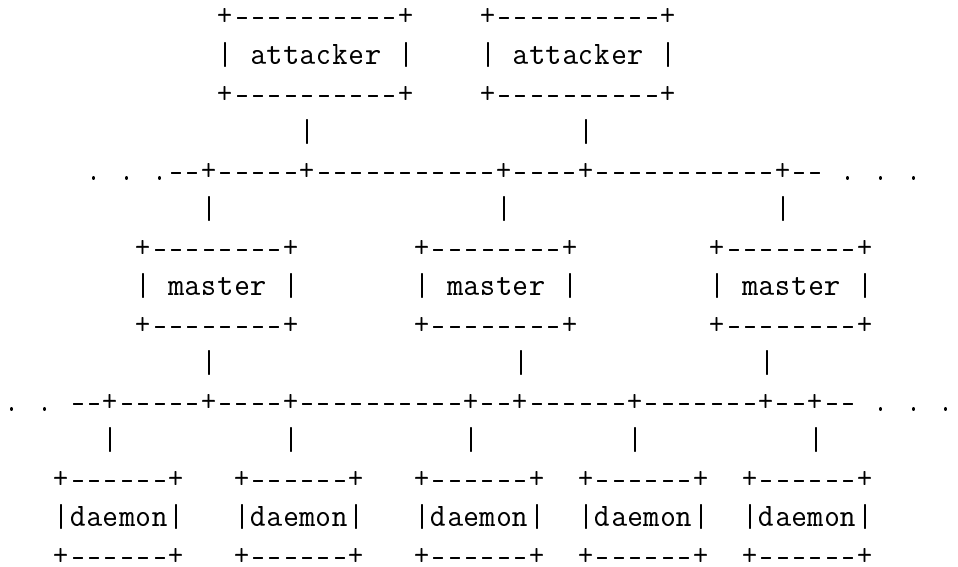
Se usa un virus que incluye un ataque de denegación de servicio preprogramado. Al cumplirse una condición de tiempo, todos los ordenadores infectados realizan un ataque conjunto contra un objetivo preseleccionado.

Un ejemplo es el virus Mydoom, que se propagaba a través de correo electrónico y estaba diseñado para que los ordenadores infectados atacaran a SCO el 1 de febrero del 2004.

4.5.2. Ataques por control remoto

Este tipo de ataques utilizan máquinas comprometidas con un troyano para realizar ataques de denegación de servicio contra cualquier ip y en cualquier momento.

- Generalmente utilizan una estructura por capas donde el atacante se conecta a maestros, que son sistemas comprometidos por el atacante, y cada uno de ellos controla a un conjunto de host esclavos que se usarán para realizar los ataques de DoS.
- Usualmente, una vez el master ha sido infectado por el atacante, el master tratará de infectar a ordenadores de su misma red como esclavos usando rutinas automatizadas para explotar vulnerabilidades en programas que acepten conexiones remotas.



Existen múltiples herramientas para relizar este tipo de ataques tales como: trinoo, Tribe Flood Network, Tribe Flood Network 2000, Shaft y Stacheldraht.

4.5.2.1. TRIN00 TRIN00 (alias Trinoo) es un conjunto de programas master-slave que implementa una herramienta de ataque distribuido de denegación de servicio.

- Los slaves de TRIN00 realizan el ataque mediante un UDP Flood.
- La comunicación entre el atacante y el master se realiza por el puerto tcp 27665.
- La comunicación entre los masters y los slaves se realiza mediante paquetes upd (puerto 27444 del master al slave y 31335 al inverso).
- Los master encriptan la lista de slaves usando blowfish.
- Los password de master y slaves se almacenan encriptados con crypt.
- Tanto master como slave no requieren permisos de root para ejecutarse, ya que usan puertos no privelegiados.
- Se puede detectar porque tanto el master como el slave tiene un conjunto de cadenas fácilmente reconocibles en el binario y se pueden ver los puertos abiertos con `lsof`.

4.5.2.2. Tribe Flood Network 2000 (TFN2K) TFN2K es una versión modificada del TFN. La terminología usada en la redes TFN es denominar cliente al programa ejecutado en el master y demonio al ejecutado en el slave.

- El cliente no escucha conexiones remotas, y se debe utilizar algún tipo de shell remoto para controlarlo.
- Las comunicaciones entre el cliente y el demonio son encriptadas y se mezclan con comunicaciones señuelo para ocultarlas. Éstas son realizados de forma aleatoria a través de paquetes TCP, UDP y ICMP.
- Las comunicaicones entre el cliente y el demonio son en unico sentido, no envinado el demonio confirmacion de recepción.

- Todos los comandos van cifrados. La clave se define en tiempo de compilación y los comando en los paquetes no se basan en cadenas, sino en id de comando seguido de sus parámetros.
- Las cabeceras de los paquetes varían de forma aleatoria, excepto el protocolo ICMP, que siempre son mensajes de tipo echo reply.
- El cliente envía los paquetes con su ip falsificada.
- El demonio puede atacar a través de floods TCP/SYN, UDP, ICMP/PING, y BROADCAST PING (SMURF); o mezclar los 4 aleatoriamente.

Todo ello difuculta de sobremanera la detección de patrones de comportamiento en la red, haciéndolo mucho más difícil de detectar.

Práctica de ataques DOS

1. Crear un ataque que combine la base del SYN Flooding y con el **Broadcast IP Flooding**, con objetivo de saturar las conexiones de una red a todos los telnet así como bloquear una máquina concreta.

4.6. Configuración segura de red: sysctl

- Sysctl es una interfaz para visualizar y cambiar dinámicamente parámentros en el kernel.
- /proc también proporciona una interfaz para examinar y visualizar esta información.
- Muchos parámetros que afectan al comportamiento de red se encuentran en: `/proc/sys/net/`.
- Nos centraremos en los parámetros de red para IPv4, que se encuentran en: `/proc/sys/net/ipv4`.
- Podemos modificar estos parámetros:
  - Variando los valores en `/proc` mediante una redirección:

    ```
    echo 0 >/proc/sys/net/ipv4/ip_forward
    ```

  - con el comando `sysctl`: `sysctl -w net.ipv4.ip_forward=1`
  - o para que se conserven al reiniciar el sistema, editando `/etc/sysctl.conf`.

Opciones para asegurar nuestra red

- Si no actuamos como router, es interesante deshabilitar el forwarding:

```
sysctl -w net.ipv4.ip_forward = 0
```

- Si nuestra máquina no tiene varias IPs , es interesante activar el *rp_filter*, que rechaza paquetes cuyo origen no se corresponde con una dirección alcanzable por la interfaz. Esta opción nos puede ayudar a evitar IP spoofing:

```
sysctl -w net.ipv4.conf.all.rp_filter = 1
sysctl -w net.ipv4.conf.default.rp_filter = 1
```

- Para evitar ser detectado por algunos escaneadores de puertos que buscan máquinas activas, se pueden ignorar todos los pings (téngase en cuenta que otros servicios legítimos también serán ignorados):

```
sysctl -w net.ipv4.icmp_echo_ignore_all = 1
```

- También por el mismo motivo y para evitar ataques de tipo smurf, se pueden ignorar los dirigidos a la interfaz broadcast:

```
sysctl -w net.ipv4.icmp_echo_ignore_broadcasts = 1
```

- En caso de recibir paquetes icmp mal formados serán registrados por el kernel. Ciertos router son propensos a generar este tipo de tráfico y para evitar generar archivos de log demasiado grandes, se puede activar en ese caso:

```
sysctl -w net.ipv4.icmp_ignore_bogus_error_responses=1
```

- El control de rutas es usado para especificar en un paquete la ruta exacta a ser usada a un destino. Esta característica puede ser usada por un atacante para hacer circular el tráfico a través de una máquina en su control, por lo que puede ser interesante desactivar el soporte a estes paquetes en el kernel:

```
sysctl -w net.ipv4.conf.all.accept_source_route = 0
sysctl -w net.ipv4.conf.default.accept_source_route=0
```

- En caso de que el router al que estemos enviando nuestro tráfico para un destino particular no sea el más adecuado, éste puede enviar un mensajde de redirección ICMP para informar de la ruta correcta a utilizar en el futuro. Si un atacante envía estos mensajes podría conseguir

dirigir nuestro tráfico a través de él. Para evitar esto se puede tomar dos opciones:

- Solo admitir paquetes de redirección con origen un gateway:

```
sysctl -w net.ipv4.conf.all.secure_redirects = 1
sysctl -w net.ipv4.conf.default.secure_redirects=1
```

- o rechazar completamente los paquetes de redirección:

```
sysctl -w net.ipv4.conf.all.accept_redirects = 0
sysctl -w net.ipv4.conf.default.accept_redirects=0
```

- También resulta interesante detectar activades sopechosa: como envío de paquetes con dirección no valida, paquetes con origen y destino la misma máquina, o paquetes con origen 127.0.0.1 a traves de un interfaz ethernet.

```
sysctl -w net.ipv4.conf.all.log_martians = 0
sysctl -w net.ipv4.conf.default.log_martians = 0
```

- Para reducir el riesgo de ataque SYN flooding se puede hacer descartar conexiones antiguas con:

```
sysctl -w net.ipv4.tcp_syncookies = 1
```

- No obstante esta opción rompe en parte con el estandar tcp, y elimina el uso de algunas extensiones. En ciertos casos se debe ajustar tambien el número máximo de conexiónes en cola admitidas (sobre todo en servidores con alta demanda), aunque ello consuma mayor memoria con:

```
sysctl -w net.ipv4.tcp_max_syn_backlog = 128
```