Ismael Briones Vilar

ARP-SPOOFING

Espiando en redes segmentadas

La segmentación de redes mediante el uso de Switches parecía la solución perfecta para evitar los temibles sniffers. Pero no es oro todo lo que reluce y es posible aprovechar una inseguridad en el protocolo ARP para espiar en la red. En este articulo vamos a explicar una de las técnicas utilizadas para poder sniffear en una red segmentada mediante Switches : ARP-SPOOFING.

Empecemos metiéndonos un poco en ambiente, aclarando ideas y definiendo términos. No quiero empezar a soltar siglas y palabras técnicas y que os quedéis todos con la boca abierta y buscando otro artículo para leer.

Redes Ethernet

La ethernet fue concebida en torno a una idea principal: todas las máquinas de una misma red local comparten el mismo medio (el cable). Todas las máquinas son capaces de "ver" todo el tráfico de la red. Debido a esto, las tarjetas ethernet incorporan un filtro que ignora todo el tráfico que no está destinado a él. Esto se consigue ignorando aquellos paquetes cuya dirección MAC (Media Access Control) no coincide con la suya. Un sniffer elimina este filtro de la tarjeta de red y la coloca en modo promiscuo. De esta forma la tarjeta es capaz de "ver" todo el tráfico que pasa por la red. Solo es cuestión de colocar los filtros adecuados y comenzar a capturar los paquetes que más nos interesen(login/passwd de conexiones de telnet, POP3, vnc,...)
El empleo de switches soluciona este problema. Mediante la segmentación de la red el único tráfico que seremos capaces de ver será el nuestro, ya que el Switch se encarga de enrutar hacia nuestro segmento solo aquellos paquetes destinado a nuestra dirección MAC.

¿Qué es una dirección MAC?

Todos los ordenadores de una misma red comparten el mismo medio, por lo que debe de existir un identificador único para cada equipo, o mejor dicho para cada tarjeta de red. Esto no sucede en una conexión telefónica mediante modem, ya que se supone que cualquier dato que se envía está destinado al equipo que se encuentra al otro lado de la línea. Pero cuando se envían datos en una red local, hay que especificar claramente a quien van dirigidos. Esto se consigue mediante la dirección MAC, un número compuesto por 12 dígitos hexadecimales que identifica de forma única a cada dispositivo ethernet. La dirección MAC se compone de 48 bits. Los 24 primeros bits identifican al fabricante del hardware, y los 24 bits restantes corresponden al número de serie asignado por el fabricante, lo que garantiza que dos tarjetas no puedan tener la misma dirección MAC. Direcciones MAC duplicadas causarían problemas en la red.

Ahora que más o menos a quedado claro que para comunicarnos en una red ethernet con otro equipo de la misma es necesario conocer su dirección MAC, la cual es única, vamos a ver como podemos conocer las direcciones MAC de los equipos de nuestra red. La configuración de la tarjeta de nuestro equipo la obtendremos con el comando ifconfig -a. La salida de este comando se asemejará al siguiente:

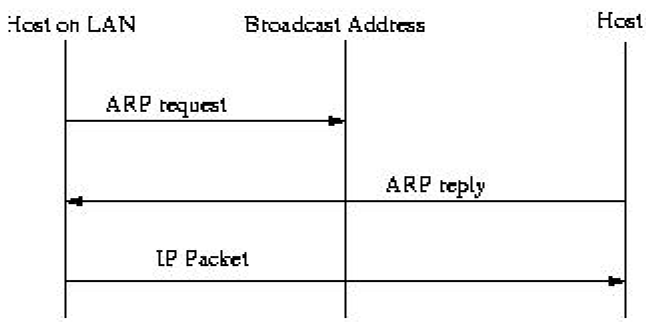
```
eth0   Link encap:Ethernet  HWaddr 00:C0:4F:68:BA:50
       inet addr:192.168.0.1  Bcast:192.168.0.255  Mask:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:31658 errors:0 dropped:0 overruns:0 frame:0
       TX packets:20940 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:100
       Interrupt:19 Base address:0xdc00
```

donde la dirección MAC es 00:C0:4F:68:BA:50.
Si queremos conocer las direcciones de otros equipos de la red, haremos uso de la cache arp de nuestro equipo, mediante el comando arp -a. Este comando nos mostrará la relación IP/MAC de los equipos que la cache tiene almacenados en ese momento (Si queremos obtener la dirección ethernet, de una maquina, primero le haremos un ping. De esta forma almacenaremos la dirección MAC en nuestra cache y mediante arp -a podremos obtener su

dirección MAC). Cada vez que deseamos comunicarnos con un equipo de la red, debemos de conocer su dirección MAC. Para ello enviaremos un arp-request a la dirección de Broadcast, solicitando la MAC de la ip del equipo con el que queremos contactar. Este responderá con un arp-reply informándonos de su MAC. Esta quedará almacenada el la cache arp, durante algunos minutos, para futuras comunicaciones. De esta forma no tendremos que volver a solicitar la dirección MAC. Aquí es donde comienza el problema.

Funcionamiento de ARP

El protocolo arp (address resolution protocol) es el encargado de "traducir" las direcciones ip de 32 bits a las correspondientes direcciones de hardware. En ethernet & Token ring estas direcciones suelen tener 48 bits. La traducción inversa la hace el protocolo RARP o Reverse ARP.
Cuando un ordenador necesita resolver una dirección IP a una MAC, lo que hace es efectuar una petición arp (Arp request) a la broadcast de dicho segmento de red, FF:FF:FF:FF:FF:FF, solicitando que el equipo con dicha IP responda con su dirección ethernet (MAC).
Esquemáticamente el proceso es:



Con el fin de reducir el tráfico en la red, cada arp-reply que llega a la tarjeta de red es almacenado en la cache, incluso si la petición no la realizamos nosotros. Es decir, todo arp-reply que nos llega es almacenado en la cache. Este factor es el que utilizaremos para realizar arp-spoofing.

Arp-Spoofing

Este método no pone la interfaz de red en modo promiscuo. Esto no es necesario porque los paquetes son para nosotros y el switch enrutará los paquetes hacia nosotros. Vamos a ver como es esto posible.

El método consiste en "envenenar" la cache arp de las dos máquinas que queremos sniffear. Una vez que las caches estén envenenadas, los dos hosts comenzarán la comunicación, pero los paquetes serán para nosotros, los sniffearemos y los enrutaremos de nuevo al host apropiado. De esta forma la comunicación es transparente para los dos hosts. L a única forma de descubrir que existe "a man in the middle" en nuestra conexión sería ver la cache arp de nuestra máquina y comprobar si existen dos maquinas con la misma dirección MAC. El esquema de la comunicación es sencillo:

Desde nuestra máquina enviaremos paquetes de tipo arp-reply falsos a las dos host que queremos sniffear. En estos reply's debemos de decirle al host 1 que la dirección ethernet del segundo host es la nuestra, quedando esta información almacenada en su cache arp. Este equipo enviará ahora los paquetes al host 2 pero con nuestra dirección MAC. Los paquetes ya son nuestros. El switch se encargará de hacernos llegar los datos. Imaginemos la siguiente situación:

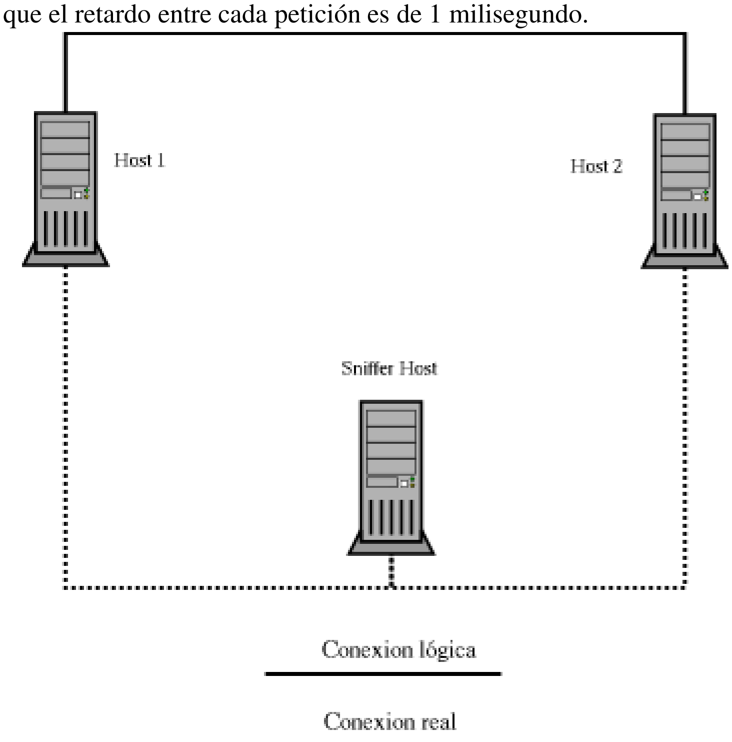
Enviamos un flujo constante de arp-reply (para evitar que la cache arp de las maquinas se refresque con la información verdadera) al host 1 y host 2 con los siguientes datos:

HOST 1 : arp-reply informando que 192.168.0.2 tiene dirección MAC 03:03:03:03:03:03
HOST 2 : arp-reply informando que 192.168.0.1 tiene dirección MAC 03:03:03:03:03:03

De esta forma estamos "envenenando" las cache arp. A partir de ahora lo paquetes que se envíen entre ambas nos llegarán a nosotros, pero para que ambos hosts no noten nada extraño, deberemos de hacer llegar los paquetes a su destino final. Para ello deberemos de tratar los paquetes que recibamos en función del host de origen:

Paquetes procedentes de HOST 1 -----------------> reenviar a 02:02:02:02:02:02
Paquetes procedentes de HOST 2 -----------------> reenviar a 01:01:01:01:01:01

De esta forma la comunicación entre ambos no se ve interrumpida, y podemos "ver" todo el tráfico entre ellos. Solo tendremos que utilizar un sniffer para poder capturar y filtrar el tráfico entre amos, ya sea login/passwd de telnet, ftp, POP3,..., o incluso la sesión completa. Eso ya depende de la habilidad y el interés de cada cual.
Como podéis comprobar el proceso no es complicado, pero ¿que utilidades tenemos disponibles para poder enviar los paquetes arp falsificados?
Existen varios programas para "juguetear" con el arp-spoofing: Arptool, Arp-Fun, ettercap. Este último esta muy completo, ya que permite varios tipos de sniffeo: Por IP, MAC y Arp-Spoofing. Pudiendo ejecutarse bien en modo comando, o mediante un entorno de ventanas. En este entorno se nos mostrará al inicio un listado de los hosts encontrados en la LAN. Para realizar esta búsqueda, el programa envía un ARP-REQUEST de las IP teniendo en cuenta la IP del host donde se está ejecutando y la máscara de red. Obteniendo a continuación los ARP-REPLYs podremos componer la lista de los hosts presentes en la red. Hay que tener mucho cuidado con la máscara de red que usemos, porque si es de clase B (255.255.0.0) el programa realizará 255*255=65025 ARP-REQUEST, lo cual le llevará su tiempo ya que el retardo entre cada petición es de 1 milisegundo.

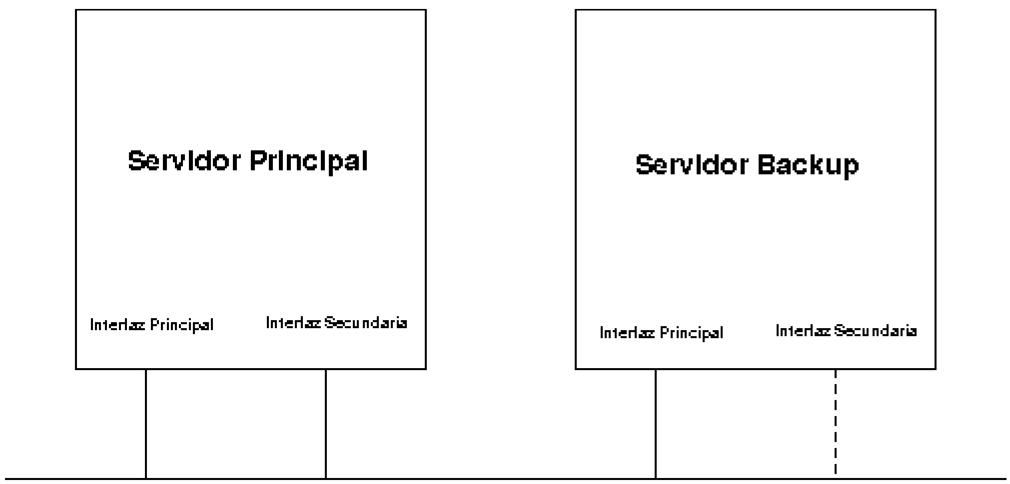


Hasta aquí hemos visto la forma en la que se pueden utilizar las vulnerabilidades del protocolo ARP para poder espiar en nuestra red. Pero las posibilidades son múltiples: ataques DoS (Denegación de servicio), si "envenenamos" la cache arp de una máquina haciéndonos pasar por el gateway de la red, toda comunicación con el exterior pasará por nosotros. Si desechamos los paquetes procedentes de este host y no los reenviamos al gateway, el host no podrá comunicarse con el exterior. Algunos switches pueden ser manipulados mediante paquetes ARP para que en vez de actuar en modo “bridging” lo hagan en modo “repetición”. Es decir, que en vez de enviar los paquetes por la “boca o puerto” adecuado del switch, los enviará por todos, a todas las máquinas les llegarán todo los paquetes de la red. Esto se consigue inundando la tabla de direcciones con gran cantidad de direcciones MAC falsas. El switch al recibir un paquete cuya dirección MAC de destino no tenga en su cache, lo enviará a todos los equipos, esperando la respuesta del equipo para poder almacenar su MAC en la cache. Pero como estamos “bombardeándola” con direcciones MAC falsas, esto no ocurrirá.

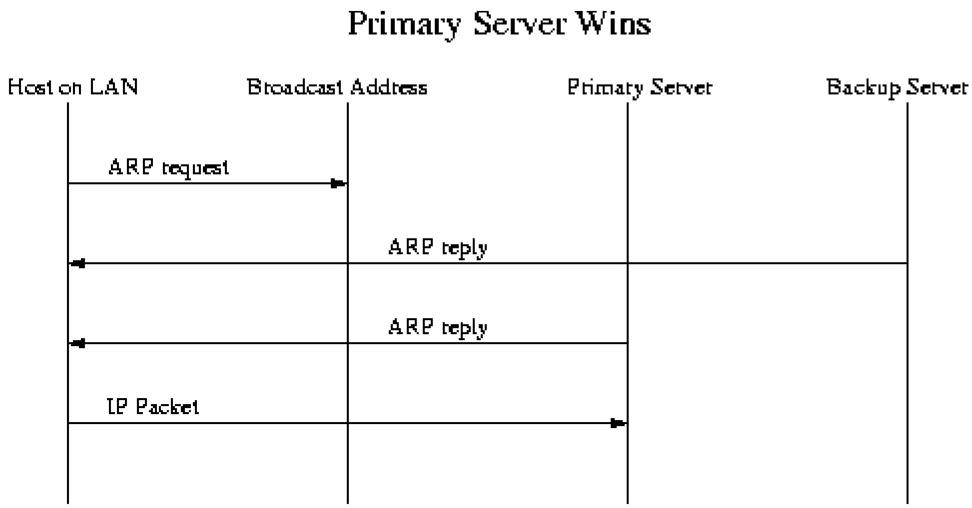
ARP-SPOOFING y servidores redundantes

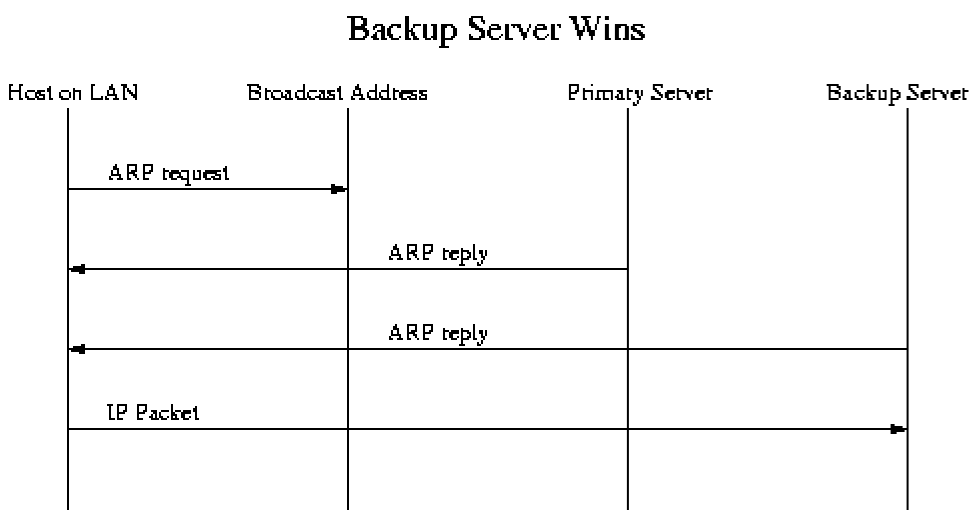
El ARP-SPOOFING no solo sirve como herramienta para espiar en una red o realizar ataques DoS. También se puede utilizar para crear servidores redundantes, servidores con tolerancia fallos. La idea consiste en crear un servidor auxiliar que tome la identidad del servidor que ha dejado de funcionar. Para ello haremos uso de IP alias y de ARP-SPOOFING. Es una solución rápida y no muy ortodoxa, pero nos puede ser útil si nuestro presupuesto no es muy elevado.

En muchas ocasiones es vital la redundancia en un determinado servicio: HTTP, FTP, SMTP, POP... . Los equipos que nos permiten la redundancia en un servicio suelen ser bastante caros, por lo que este método nos puede solucionar el problema. La situación es la siguiente:



Tenemos un servidor principal con dos interfaces de red (la segunda nos permitirá acceder al servidor cuando el backup esté funcionando) y un servidor de backup también con dos tarjetas de red (o bien puede utilizarse una sola en el redundante y emplear IP Alias para poder disponer de dos direcciones IP).
El servidor de backup debe de configurarse de forma que cuando se active (porque el principal ha sufrido algún problema y deja de funcionar) configure su tarjeta de red con la IP del servidor principal que debe de sustituir (bien configurando la única tarjeta con IP Alias o configurando una segunda tarjeta). A continuación el servidor de backup utilizará ARP SPOOFING para asegurarse de que recibe todos los paquetes destinados al servidor que el está sustituyendo. Los paquetes ARP que se enviarán informarán de que la dirección MAC correspondiente a la IP del servidor principal es ahora la MAC del servidor de backup. Este envío debe de producirse de manera continua, mientras dure la caída del principal, evitando así que las cache arp se refresquen con la dirección MAC verdadera del servidor principal. Si la cache arp expirase y se actualizase con la MAC verdadera del servidor principal, se produciría un “race condition”, como podemos ver a continuación:





Como podemos ver en la figura, si el servidor de backup responde al ARP REQUEST con su ARP REPLY antes que el servidor principal, el ARP REPLY del backup se “machacará” con el del principal. Mientras que si es al contrario, será el ARP REPLY del principal el que será sustituido por el del servidor de backup. Esto no debe de llegar a producirse, ya que sino el proceso de redundancia sería inútil.
La activación del servidor de backup deberá de producirse de forma automática, ya que estos problemas suelen producirse de madrugada (no siempre claro) y no serviría de nada que fuésemos nosotros los que tuviésemos que activarlo manualmente.
Utilizar un servidor de NFS para ambos equipos sería lo más conveniente, ya que de esta forma podremos acceder a los mismos contenidos desde ambos servidores, una solución interesante para servicios como HTTP, POP3, FTP...

Lo principal ya esta explicado, ahora solo tienes que hacer uso de tu imaginación. En la bibliografía encontrarás algunos artículos en los que me he basado para escribir este artículo y que seguro también te pueden servir de ayuda. Ahora solo tienes que leer y practicar. Pero recuerda una cosa: “NO ROMPAS NADA”.

Ismael Briones Vilar
ismak@inkatel.com
ismael@el-mundo.net

[BIBLIOGRAFÍA]

[1] Sniffing (network wiretap,sniffer) FAQ
http://www.robertgraham.com
[2] Creating Redundant Linux Servers
http://linuxexpo.zip.com.au