Parte 1 (70%)

Usted debe implementar una herramienta, llamada *arpdespoof*, la cual detecta ataques ARP *spoofing*. La herramienta hace *sniffing* de la red (o lee desde un archivo *pcap* que contiene un tráfico de red) buscando el tráfico ARP. Detecta un ataque cuando identifica que una solicitud ARP, en un intervalo de tiempo dado (configurable por el usuario), recibe múltiples respuestas que son diferentes entre sí.

Cuando un ataque es detectado, la herramienta imprime la siguiente alerta en la salida estándar (todo en una línea):

```
DETECT: who-has <target-ip>, R1: <ethernet1>, R2: <ethernet2>, TS: <secs.usecs>
```

Por ejemplo, si una solicitud ARP para 172.16.48.130 recibe dos respuestas, una aseverando que 172.16.48.130 está en 00:0c:01:01:01:01 y la otra que 172.16.48.130 está en 00:0c:02:02:02:02, la herramienta debe mostrar (asumiendo que la segunda respuesta fue recibida en el timestamp 1296003035.190345):

```
DETECT: who-has 172.16.48.130, R1: 00:0c:01:01:01:01, R2: 00:0c:02:02:02:02, TS: 1296003035.190345
```

La herramienta puede imprimir mensajes de *debug* que empiecen con la cadena "DEBUG:". No debe producir ninguna otra salida en la salida estándar.

La herramienta debe aceptar las siguientes opciones desde línea de comando:

- -i DEV, especifica la interface de red con la cual hacer sniff.
- -r FILE, lee el tráfico de red desde un archivo pcap, en vez de hacerlo desde una tarjeta de red.
- -t SECONDS, especifica la ventana de tiempo que puede ser usada en la detección (si no se especifica, la ventana de tiempo debe ser 5 segundos)

La herramienta puede ser implementada en C (recomendado) o Java:

Instrucciones para C:

La herramienta lee el tráfico en la red usando la librería *libpcap*. La herramienta debe ser implementada como un único archivo, llamado arpdespoof.c. Este archivo debe compilar usando el siguiente comando:

```
$ gcc arpdespoof.c -o arpdespoof -lpcap -lnet
```

Subir el archivo arpdespoof.c al SidWeb.

Instrucciones para Java:

La herramienta lee el tráfico de red usando *jNetPcap* o la librería *Jpcap*. La herramienta debe estar implementada como un único archivo, llamado ArpDespoof.java. Usted debe crear un archivo jar que contenga el código y se llame ArpDespoof.jar. Se debe poder correr la herramienta con (asumiendo que libs contiene las librerías jNetPcap y Jpcap):

```
$ java -classpath ArpDespoof.jar:libs/* netsec.hw2.ArpDespoof
```

Envíe los archivos ArpDespoof.java y ArpDespoof.jar en el SidWeb.

Pruebas: para efectos de pruebas, puede usar los siguientes archivos:

- hw2-trace1.pcap: hw2-trace1-exp.txt (salida esperada)
- hw2-trace2.pcap: hw2-trace2-exp.txt (salida esperada)

Parte 2 (30%)

Usted debe implementar una herramienta llamada antiscan, que impide un ICMP-based scanning. En particular, la herramienta debe simular un host no existente, llamado *target*, tanto a nivel Ethernet como a nivel IP. Cuando el intruso envía un mensaje ICMP echo request al target (para verificar si está arriba), la herramienta genera el respectivo ICMP echo reply, de esta manera confunde al intruso en su intento de ataque.

Note que usted deberá hacer spoofing tanto a mensajes ICMP, como a paquetes ARP que son necesarios para simular la presencia del host target.

Suponga que antiscan es configurada para simular la presencia de 172.16.48.1. Entonces, al hacer un ping al host target deberíamos obtener una salida similar a:

```
$ ping -c 1 172.16.48.1
  PING 172.16.48.1 (172.16.48.1) 56(84) bytes of data.
  64 bytes from 172.16.48.1: icmp_seq=1 ttl=64 time=2.48 ms

  --- 172.16.48.1 ping statistics ---
  1 packets transmitted, 1 received, 0% packet loss, time 0ms
  rtt min/avg/max/mdev = 2.483/2.483/2.483/0.000 ms
```

La herramienta debe aceptar las siguientes opciones por línea de comando:

- -i DEV, especifica la tarjeta de red con la que se va a hacer sniff
- -t TARGET, especifica el target a simular

La herramienta puede ser implementada en C (recomendado) o Java:

Instrucciones para C:

La herramienta lee el tráfico de la red usando la librería libpcap. La herramienta debe ser implementada como un único archivo, llamado antiscan.c. Este archivo debe compilar por medio del siguiente comando:

```
$ gcc antiscan.c -o antiscan -lpcap -lnet
```

Envíe el archivo antiscan.c por SidWeb.

Instrucciones para Java:

La herramienta lee el tráfico usando la librería jNetPcap o Jpcap. La herramienta debe ser implementada como un único archivo llamado AntiScan.java. Usted debe crear un archivo jar AntiScan.jar que contenga su código fuente. Se debe poder correr la herramienta por medio de (asumiendo que libs contiene las librerías jNetPcap y Jpcap):

```
$ java -classpath AntiScan.jar:libs/* netsec.hw2.AntiScan
```

Envíe el archivo AntiScan.java y AntiScan.jar usando SidWeb.

Pruebas: se debe realizar en una red aislada.

Autor: Marco Cova