Relazione di Network Security

Nome: Marco

Cognome: De Stefano

N° Matricola: 7025991

Data: Novembre 2024

1 Introduzione

In questa relazione analizzeremo le vulnerabilità di sicurezza, concentrandoci su una in particolare, che testeremo e simuleremo, illustrando i metodi di configurazione dell'ambiente di test e i risultati ottenuti.

Ci focalizzeremo in particolare sull'ARP Spoofing, una tecnica che sfrutta una vulnerabilità strutturale del protocollo ARP.

2 Protocollo ARP

Prima di descrivere nel dettaglio l'attacco e il suo funzionamento, è utile comprendere meglio il protocollo ARP. L'Address Resolution Protocol (ARP) è utilizzato nelle reti LAN per ottenere l'indirizzo MAC (Media Access Control) dei dispositivi in rete a partire dal loro indirizzo IP. In particolare, l'ARP funziona in questo modo:

1. Quando un dispositivo deve ottenere l'indirizzo MAC di un altro dispositivo nella stessa sottorete, conoscendo solo il suo indirizzo IP, invia una richiesta di tipo **ARP request** in modalità broadcast, chiedendo "Qual è l'indirizzo MAC associato a questo indirizzo IP?".

2. Ogni dispositivo della rete locale riceve la richiesta e confronta l'indirizzo IP richiesto con il proprio. Solo il dispositivo con l'indirizzo IP corrispondente risponde inviando una **ARP reply** al mittente, contenente il proprio indirizzo MAC. Questo messaggio viene inviato in modalità unicast, cioè solo al dispositivo che ha originato la richiesta ARP.

3. Ricevuta la risposta ARP, il dispositivo aggiorna la propria tabella di associazione tra indirizzi IP e indirizzi MAC (nota come **ARP cache**), memorizzando il risultato per velocizzare le comunicazioni future all'interno della sottorete.

Nell'immagine sottostante è illustrato il funzionamento del protocollo ARP in una rete con topologia a bus; lo stesso principio si applica a qualsiasi topologia. In una rete a topologia a stella, ad esempio, il messaggio broadcast viene inoltrato dallo switch, il quale invia la richiesta a tutti i dispositivi connessi.

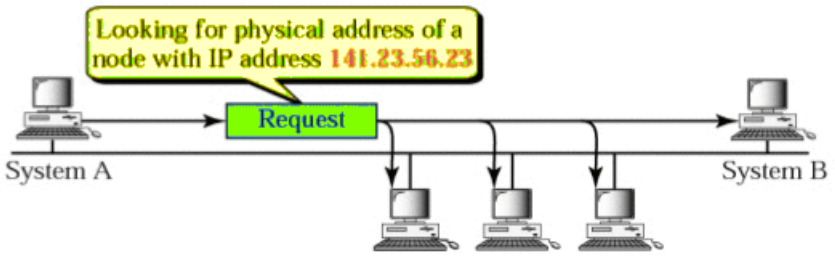

Fig: ARP request is broadcast

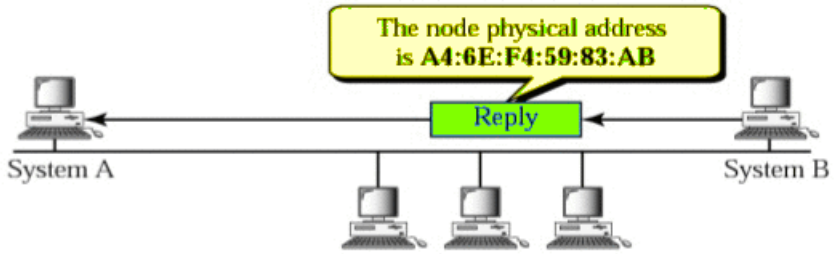

Fig: ARP reply is unicast

Figure 1: Funzionamento Protocollo ARP

3 Tipologia di Vulnerabilità Analizzata

Nel protocollo ARP, come descritto precedentemente, si evidenzia una vulnerabilità significativa: l'assenza di un meccanismo di autenticazione. Questa vulnerabilità è di design, poiché è nota e accettata sin dalla progettazione del protocollo.

I principali fattori che rendono il protocollo ARP vulnerabile sono:

- **Mancanza di Autenticazione:** ARP accetta automaticamente tutte le risposte ricevute senza verificare se provengano effettivamente dal mittente legittimo. Questo significa che chiunque nella rete locale può inviare risposte ARP contraffatte, associando il proprio indirizzo MAC all'indirizzo IP di un altro dispositivo.
- **Aggiornamento Non Autenticato della Cache ARP:** I dispositivi in una rete locale mantengono una cache ARP che associa indirizzi IP e MAC. Poiché le nuove risposte ARP aggiornano automaticamente questa cache, un utente malintenzionato può sovrascrivere tali associazioni inviando pacchetti ARP falsificati.
- **Fiducia Implicita nella Rete Locale:** ARP è stato progettato per reti locali considerate "affidabili", senza prevedere misure per prevenire azioni dannose. Questa fiducia implicita rende ARP vulnerabile in ambienti condivisi o non sicuri, in cui qualsiasi dispositivo può inviare pacchetti ARP senza restrizioni.

4 Attacco ARP Spoofing

L'**ARP Spoofing** è un attacco che sfrutta le vulnerabilità del protocollo ARP per realizzare un attacco di tipo Man-In-The-Middle.

L'uso di questo tipo di attacco è motivato dal fatto che, nelle moderne reti Ethernet, gli hub sono stati in gran parte sostituiti dagli switch. A differenza degli hub, gli switch utilizzano una tabella per inoltrare il traffico esclusivamente al dispositivo di destinazione, rendendo inefficaci i tentativi di sniffing tradizionale.

In un attacco ARP Spoofing, l'attaccante invia risposte ARP (ARP reply) appositamente modificate e costruite in modo da "avvelenare" le cache ARP dei dispositivi presenti nella rete. In questo modo, i dispositivi vengono ingannati: credono di avere associazioni IP-MAC corrette per i dispositivi della rete, ma alcune di queste associazioni puntano invece al MAC dell'attaccante. Ciò consente all'attaccante di intercettare il traffico delle vittime, visualizzando così le comunicazioni all'interno della rete locale e ritrasmettendole al vero destinatario.

Questo attacco è noto anche come **ARP Poisoning**. Si tratta di un attacco di tipo **link-local**, poiché avviene interamente all'interno della stessa rete locale.

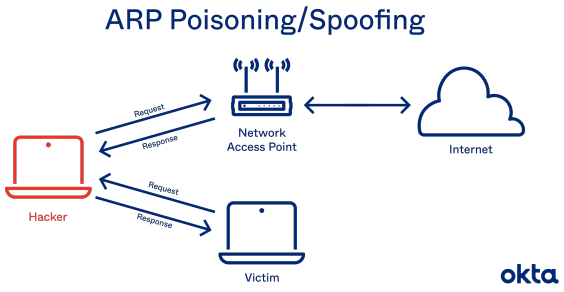

Figure 2: Attacco ARP Poisoning o Spoofing

5 Esperimento

5.1 Creazione del Testbed

Per simulare la vulnerabilità di ARP Spoofing, creiamo un testbed utilizzando Docker su un sistema host con macOS. Di seguito i passaggi per configurare l'ambiente di test:

- **Creazione della rete:** per prima cosa, si crea una rete Docker di tipo bridge con il comando:

  ```
  docker network create -d bridge test_network
  ```

  Dove:

  - `docker network`: è il comando per gestire le reti in Docker.
  - `-d bridge`: specifica il tipo di rete da creare; le reti bridge sono ideali per isolare gruppi di container, permettendo loro di comunicare tra loro come se fossero su una stessa LAN.
  - `test_network`: è il nome assegnato alla rete.

  Questo comando crea una rete personalizzata chiamata `test_network`, che isola i container dal resto del sistema e dalla rete esterna. I container collegati a questa rete possono comunicare tra loro come se fossero sulla stessa rete locale.

- **Creazione dei container client e server:** si aggiungono alla rete due container che fungeranno rispettivamente da client e server, rappresentando due dispositivi della rete. I comandi sono i seguenti:

  - `docker run -d --name server --network test_network nginx`: questo comando avvia un container chiamato `server` utilizzando l'immagine 'nginx', un server web open-source.
    - L'opzione `-d` (detached) esegue il container in background, mantenendolo attivo senza bloccare il terminale.
    - Il container è connesso alla rete `test_network`.
  - `docker run -it --name client --network test_network ubuntu /bin/sh`: questo comando avvia un container chiamato `client` utilizzando l'immagine 'ubuntu'.
    - L'opzione `-it` lo avvia in modalità interattiva, aprendo una shell ('/bin/sh') per interagire con il container.
    - Anche questo container è connesso alla rete `test_network`.

- **Creazione del container attaccante:** per simulare l'attacco ARP Spoofing, si aggiunge un container 'attacker':

  ```
  docker run -it --name attacker --network test_network kalilinux/kali-rolling
                                        /bin/bash
  ```

  Questo comando avvia un container basato sull'immagine 'kali-rolling' di Kali Linux, un sistema operativo progettato per test di sicurezza.

  - Dopo aver avviato il container, si esegue il comando `apt update && apt install -y dsniff` per installare 'dsniff', una suite di strumenti per l'intercettazione del traffico di rete. La suite include 'arpspoof', uno strumento che consente di eseguire l'ARP Spoofing, reindirizzando il traffico tra host nella stessa rete locale.

- **Creazione del container monitor:** su macOS, Docker esegue i container in una macchina virtuale (VM) e non fornisce un'interfaccia di rete 'docker0' per monitorare il traffico di rete direttamente dall'host. Per ovviare a questa limitazione, si crea un container 'monitor':

  ```
  docker run -it --name monitor --network test_network ubuntu /bin/sh
  ```

  Una volta avviato, si installa 'tcpdump' con il comando:

  ```
  apt update && apt install -y tcpdump
  ```

  'tcpdump' è uno strumento per la cattura del traffico di rete. È simile a Wireshark, ma opera da linea di comando e consente di salvare i dati catturati in formato '.pcap' per un'analisi successiva.

5.2 Svolgimento Esperimento

Una volta creato il testbed, verifichiamo innanzitutto che il client e il server siano correttamente connessi. Per fare ciò, accediamo al terminale del client con il comando:

```
docker exec -it client /bin/sh
```

All'interno del terminale del client, utilizziamo il comando curl per inviare una richiesta al server:

```
curl http://server
```

Se la connessione è avvenuta con successo, visualizzeremo la pagina di benvenuto del server, confermando che il client e il server sono connessi correttamente.

```
(base) marcodestefano@MacBook-Pro-di-Marco-4 ~ > docker exec -it client /bin/sh
# curl http://server
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
#
```

Figure 3: Tentativo connessione con server

Successivamente, accediamo alla bash del container monitor ed eseguiamo il seguente comando:

```
tcpdump -i any -w /tmp/container_traffic.pcap
```

Questo comando cattura tutto il traffico di rete su tutte le interfacce disponibili e salva i pacchetti catturati nel file /tmp/container_traffic.pcap.

Figure 4: Ispezione della rete

Dalla figura sopra, si può vedere i container nella rete **test_network**:

- attacker:
  - **MAC Address:** 02:42:AC:12:00:05
  - **IP Address:** 172.18.0.5/16
- server:
  - **MAC Address:** 02:42:AC:12:00:02
  - **IP Address:**172.18.0.2/16
- client:
  - **MAC Address:** 02:42:AC:12:00:04
  - **IP Address:** 172.18.0.4/16

Attraverso il container **attacker** si avvia lo strumento **arpspoof** attraverso il comando:

```
arpspoof -i eth0 -t 172.18.0.2 172.18.0.4
```

Ecco una descrizione dei vari parametri:

- **arpspoof**: è lo strumento che esegue l'attacco di ARP spoofing.
- **-i eth0**: specifica l'interfaccia di rete su cui l'attacco deve essere eseguito.
- **-t 172.18.0.2**: indica il target dell'attacco, ovvero l'indirizzo IP del dispositivo che si desidera ingannare (in questo caso il server).
- **172.18.0.4**: è l'indirizzo IP del dispositivo che si intende impersonare. Il comando fa credere al dispositivo con l'IP 172.18.0.2 che l'indirizzo MAC associato a 172.18.0.4 appartenga al dispositivo che esegue l'attacco.

Nel mentre che l'attacker esegue l'attacco, attraverso il container monitor catturo tutto il traffico con tcpdump che viene salvato tutto nel file container_traffic.pcap .

```
root@8f63e6652dbe:/# tcpdump -i any -w /tmp/container_traffic.pcap
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
^C14 packets captured
14 packets received by filter
0 packets dropped by kernel
```

Figure 5: Dimostrazione che tcpdump ha catturato dei pacchetti

A questo punto copiamo `container_traffic.pcap` sul file `traffic.pcap` nel nostro dispositivo eseguendo il seguente comando docker:

```
docker cp monitor:/tmp/container_traffic.pcap ./traffic.pcap
```

```
(base) marcodestefano@MacBook-Pro-di-Marco-4 ~ > docker cp monitor:/tmp/container_traffic.pcap ./traffic.pcap
                                                Successfully copied 2.56kB to /Users/marcodestefano/traffic.pcap
```

Figure 6: Il file container_traffic.pcap viene copiato nell'host

Analizzo dal mio dispositivo effettivo il file traffic.pcap con Wireshark ottenendo questo:

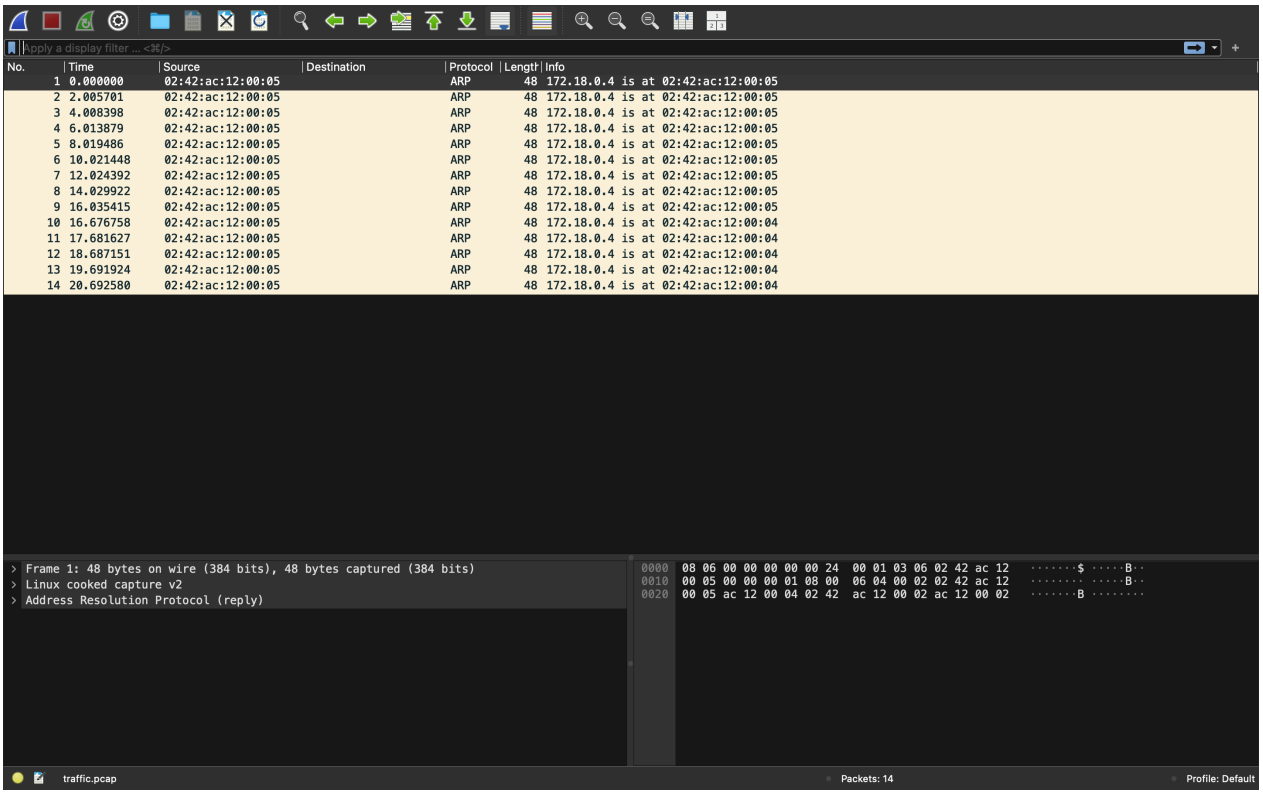

Figure 7: Scansione visualizzata su Wireshark

Nella figura 7 si può osservare che, nei primi nove pacchetti ARP, viene inviata al server l'informazione falsa che l'indirizzo IP 172.18.0.4 debba essere associato all'indirizzo MAC dell'attaccante.

Per verificarlo, accediamo al server e, eseguendo il comando `arp -a` per visualizzare la **ARP cache**, notiamo che, durante l'attacco di ARP Spoofing, l'indirizzo IP del dispositivo impersonificato risulta associato al MAC dell'attaccante.

Una volta che l'ARP Spoofing viene interrotto, la ARP cache viene ripristinata, riportando gli accoppiamenti corretti tra IP e MAC, come erano prima dell'attacco.

```
root@96fd6cb37959:/# arp -a
client.test_network (172.18.0.4) at 02:42:ac:12:00:05 [ether] on eth0
? (172.18.0.1) at 02:42:c1:1f:27:10 [ether] on eth0
attacker.test_network (172.18.0.5) at 02:42:ac:12:00:05 [ether] on eth0
monitor.test_network (172.18.0.3) at 02:42:ac:12:00:03 [ether] on eth0
root@96fd6cb37959:/# ^C
root@96fd6cb37959:/# arp -a
client.test_network (172.18.0.4) at 02:42:ac:12:00:04 [ether] on eth0
? (172.18.0.1) at 02:42:c1:1f:27:10 [ether] on eth0
attacker.test_network (172.18.0.5) at 02:42:ac:12:00:05 [ether] on eth0
monitor.test_network (172.18.0.3) at 02:42:ac:12:00:03 [ether] on eth0
```

Figure 8: Due visualizzazioni dell' ARP cache, il primo durante attacco e successivamente quello dopo l'attacco

Sempre osservando la figura 7, notiamo che gli ultimi pacchetti ARP presentano una differenza sostanziale rispetto ai primi. Questi ultimi contengono la corretta associazione tra indirizzo IP e MAC. Questo accade quando l'attacco viene interrotto, poiché l'attaccante, per non lasciare tracce, deve evitare di mantenere il proprio indirizzo MAC nel server.
Per risolvere questo, si effettua un'operazione di Re-Arping, durante la quale vengono inviati nuovi pacchetti ARP con l'associazione IP-MAC corretta, ripristinando la cache come se l'attacco non fosse mai avvenuto.
Si può vedere come, durante l'attacco con lo strumento `arpspoof`, vengano inviati numerosi pacchetti "velenosi" che consentono all'attaccante di acquisire il traffico destinato al dispositivo impersonificato. Finché

l'attacco è attivo, l'attaccante agisce come un vero e proprio Man In The Middle. Quando l'attacco viene interrotto, l'attaccante invia pacchetti ARP per ripristinare correttamente la cache del server. Da quel momento, l'attaccante non può più intercettare il traffico destinato al dispositivo compromesso.

6 Conclusioni

L'ARP Spoofing rappresenta un attacco di tipo Man In The Middle (MITM), in cui un malintenzionato intercetta e manipola il traffico di rete destinato a un dispositivo, inserendosi tra quest'ultimo e il destinatario legittimo. L'attaccante agisce in modo da intercettare i pacchetti inviati alla vittima e passarli al destinatario originale, visualizzando o alterando i dati prima che giungano al loro destinatario legittimo. In questo contesto, l'ARP Spoofing sfrutta la vulnerabilità del protocollo ARP, che non include meccanismi di autenticazione per verificare la validità delle risposte ARP, permettendo così a un attaccante di inviare risposte false e di associare il proprio indirizzo MAC a un indirizzo IP legittimo.

Tuttavia, esistono diverse contromisure per proteggere le reti da questo tipo di attacco. Tra le principali soluzioni vi sono:

- **Static ARP entries:** Una delle soluzioni più efficaci per prevenire l'ARP Spoofing consiste nella configurazione di voci ARP statiche sui dispositivi critici della rete, come router, server e computer sensibili. Le voci ARP statiche associano permanentemente un indirizzo IP a un indirizzo MAC specifico, impedendo che tali associazioni possano essere modificate da pacchetti ARP falsificati. In questo modo, anche se un attaccante invia pacchetti ARP falsi, questi non avranno alcun effetto poiché il dispositivo ignorerà le risposte ARP che non corrispondono alle voci statiche configurate.
- **ARP Spoofing Detection:** Esistono diversi software di monitoraggio della rete che sono in grado di rilevare attività sospette relative all'ARP Spoofing. Questi strumenti analizzano continuamente la rete alla ricerca di anomalie, come risposte ARP incoerenti o pacchetti ARP ripetuti provenienti dallo stesso indirizzo IP. Quando viene rilevato un attacco di ARP Spoofing, il software può allertare l'amministratore della rete o bloccare automaticamente il traffico sospetto.
- **Protocollo di autenticazione per ARP:** Alcuni protocolli di sicurezza avanzati, come il Dynamic ARP Inspection (DAI), consentono di autenticare le risposte ARP e verificare la validità delle associazioni IP-MAC prima di aggiornarle nella cache ARP dei dispositivi di rete. Questi protocolli richiedono che le risposte ARP siano verificate rispetto a una base di dati conosciuta o a una chiave di autenticazione, riducendo notevolmente la possibilità che pacchetti ARP falsi vengano accettati dalla rete.

In conclusione, sebbene l'ARP Spoofing sia un attacco relativamente semplice da eseguire, esistono numerose tecniche per prevenire o rilevare questo tipo di vulnerabilità nelle reti locali.