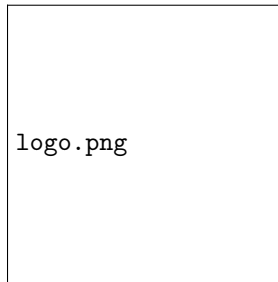


DEPARTMENT OF INFORMATION
ENGINEERING AND COMPUTER SCIENCE
(DISI)
UNIVERSITY OF TRENTO, ITALY



NETWORK SECURITY - AY 2017/2018

Lab Activity Report

Man in the Middle attacks

Author:
Gabriele Gemmi [198042]
Lorenzo Brugnera [197054]

1 Introduction

What are Man in the Middle (MitM) attacks A man in the Middle attack is an attack where the attacker secretly relays and possibly alters the communication between two parties who believe they are directly communicating with each other. The aim of the attacker is to take the control of the communication and to intercept all relevant messages passing between the two victims.

Once the attacker puts himself “*in the middle*”, he can do eavesdropping, which means secretly listening to a private conversation of other parties without their consent. This type of attack is very effective when the connection between the two victims is not secure (HTTP). In this way, all the traffic is exchanged in clear and the attacker can easily manage to intercept sensitive data, like usernames, passwords, cookies and so on.

A MitM attack can be performed on secure connections as well, with the only difference that the attacker has to establish two independent SSL sessions, one for each TCP connection. Generally, browsers handle this kind of things, notifying the user that the certificate is not valid.

A MitM is well represented by the following image:

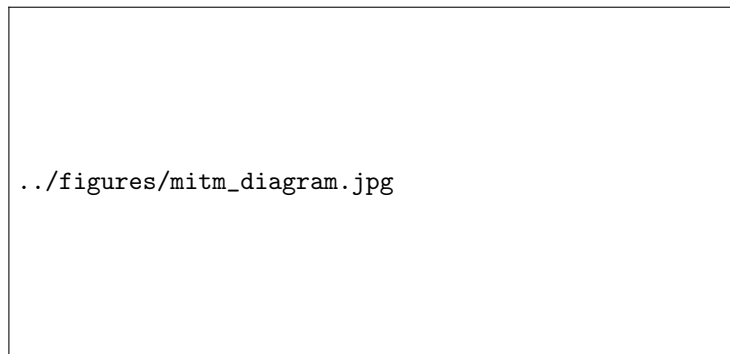


Figure 1: Diagram of a MitM attack

There are several ways to mount a MitM, but one important requisite is that the attacker must be “close” to the victims, more precisely in the same local network.

1.1 Network attacks

- ARP poisoning
- DHCP (DHCPv6) poisoning
- Evil Twin

1.2 Service attacks


1.2.1 ARP Poisoning

The Address Resolution Protocol (ARP) is a communication protocol used to discover link layer addresses, such as a MAC addresses, associated with a given network layer address, which is typically IPv4. This mapping between MAC and IP addresses is a critical function in the Internet protocol suite.

ARP poisoning is a technique by which an attacker sends spoofed ARP messages onto a local area network with the purpose to alter the mapping between IP and MAC addresses. ARP spoofing may allow an attacker to intercept data frames on a network, modify the traffic, or stop all traffic. Often the attack is used as an opening for other attacks, such as denial of service, man in the middle, or session hijacking attacks. The basic principle behind ARP poisoning is the exploitation of the the ARP protocol by sending spoofed messages onto the LAN. ARP poisoning attacks can be run from a compromised host on the LAN, or from an attacker's machine that is connected directly to the target LAN.

Generally, the goal of the attack is to associate the attacker's host MAC address with the IP address of a target host, so that any traffic meant for the target host will be sent to the attacker's one. The attacker may choose to manipulate the data before forwarding them or simply forwarding the packets to the destination in order to sniff private informations. The following diagram shows the behaviour of the attack:

Fixme Note: Too many repetition of ARP



../figures/arp_spoofing.jpeg

Figure 2: ARP Spoofing attack diagram

Countermeasures Several countermeasures can be adopted to prevent an ARP poisoning attack:

- ARP poisoning proof switches
- Port-Based Network Access Control (802.1x), based on the control of the access ports to the LAN. The protocol provides a mechanism to authenticate devices which want to establish a connection to LAN or WLAN network through a switch or a Wi-Fi access point
- VPN, based on encryption. A VPN uses not only an encrypted tunnel, but data are encrypted as well

FiXme Note: We should either do a list of "things" without providing further information or organize it and provide these information for all of them

1.2.2 DHCP Poisoning

Dynamic Host Configuration Protocol (DHCP) poisoning is a technique which consists in setting up a rogue DHCP server controlled by the attacker. Each time a client sends a requests to the legit server, the rogue's response is supposed to arrive earlier than the legit one. When the legit response arrives, it is simply discarded. The rogue server can spoof the gateway and all the traffic going away from the local subnet will start to flow into the attacker machine. Then, the attacker forwards the received traffic accordingly, so the host will not notice any disruption in connectivity. The following diagram explains how the attack works:

FiXme Note: Should we write a brief introduction on what is each of these things? ARP, DHCP, SSL, HSTS?

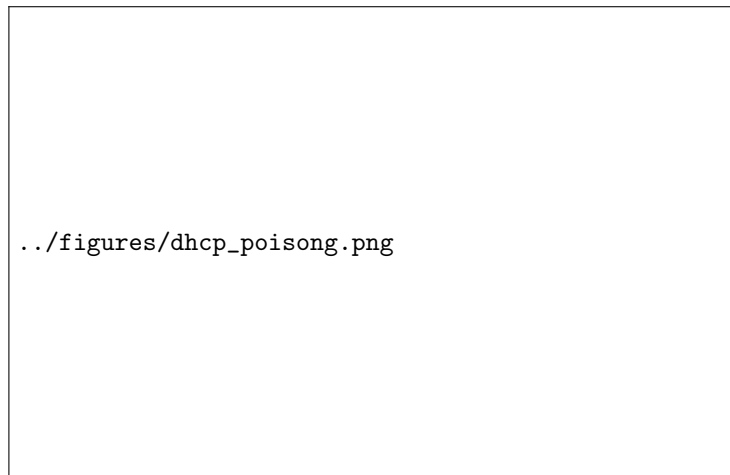


Figure 3: DHCP poisoning attack diagram

Countermeasures Smart switches are one of the best solutions to avoid this kind of attack. They are based on a mechanism which sets a port as trusted or untrusted. As you can see by the following diagram, the router expects to

FiXme Note: We should use the same format as before

receive DHCP responses only from the real DHCP server, so all the responses which does not come from the legit server are discarded. The following diagram better clarifies the explained behaviour:

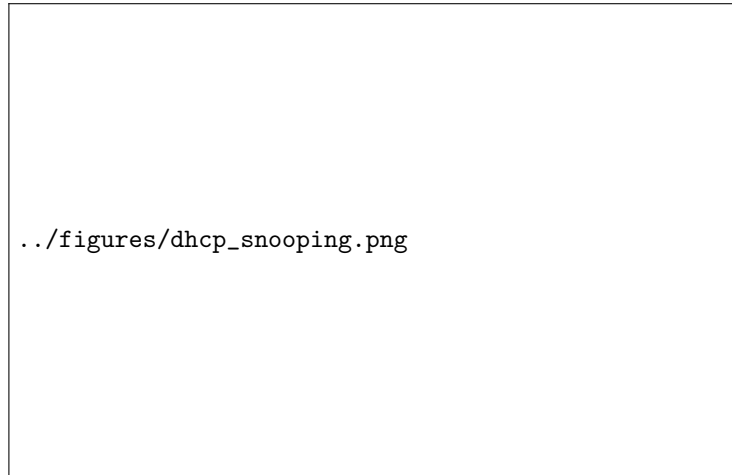


Figure 4: DHCP snooping diagram

1.2.3 Evil Twin

An evil twin is a fraudulent Wi-Fi access point (AP) that appears to be legitimate, set up to eavesdrop on wireless communications. The rogue access point must have the same Service Set Identifier (SSID) of the real one and the victim must receive the rogue AP with a stronger signal than the legitimate one.

Fixme Note:
Explain why two AP could share the same SSID, concept of ESS

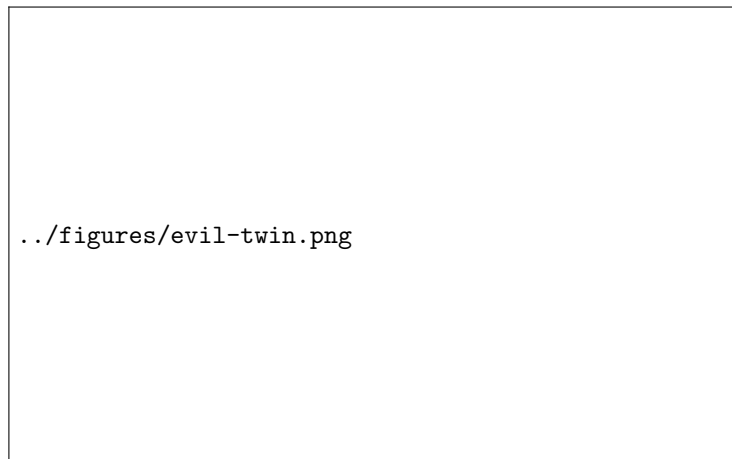


Figure 5: Evil Twin attack diagram

Countermeasures Usually, a simple authentication (WPA) does not ensure the client that the access point is legitimate, because the attacker just need to discover the key. The client must authenticate the AP using 802.1x and verify its legitimacy.

FiXme Note:
same as before

2 Laboratory activity overview

The following diagram shows up the network topology we set up in order to mount all the attacks we wanted to explain. It is quite simple and composed by:

- Victim (Client)
- Server
- Attacker
- Router



FiXme Note:
PARLARE
DELLA CON-
FIGURAZIONE
DI OPENWRT,
DIRE ANCHE
CHE TIPO DI
MACCHINE
VIRTUALI CI
SONO
INSTALLATE

Figure 6: Topology of the VMs network

3 Exercises

3.1 Exercise 1 - HTTP Interception

This first exercise is a very simple one, just to take confidence with the machines. Basically, in this part you have to capture all the traffic that flows between the victim and the router. If the traffic is not encrypted, sensitive information can be sniffed by the attacker, due to the fact that they are exchanged in clear.

So, first of all you have to mount a Man in the Middle (MitM) network attack: you can use either any of the attacks we have just illustrated to you or the ettercap¹ tool. Because this is a laboratory focusing on MitM attacks, for simplicity we will use this simple command line tool available in the dsniiff package²:

After this, from the client virtual machine you have to navigate to a non-secure webpage (<http://www.homepage.it>) and, then, using Wireshark³ sniff the HTTP traffic exchanged between the victim and the server.

Countermeasures To prevent this kind of attack, an encrypted channel can preserve the confidentiality of the data:

- SSL/TLS, which is a cryptographic protocol aimed at preserving privacy and data integrity over a computer network between two communicating computer applications
- VPN, based on encryption. A VPN uses not only an encrypted tunnel, but data are encrypted as well

FiXme Note:
(l'abbiamo già
detto sopra)

¹<http://www.ettercap-project.org/ettercap/>

²<https://www.monkey.org/~dugsong/dsniiff/>

³<https://www.wireshark.org/>

3.2 Exercise 2 - SSL Stripping

Roughly speaking, SSL Strip is a technique by which every occurrence of `https://` webpage gets replaced in `http://`. In this type of attack, all the traffic from the victim's machine is routed into the attacker's one via a transparent proxy (`sslstrip`⁴) that is created by the attacker himself. The interesting thing is that the victim will not get any notification by the browser regarding SSL certificate errors and, in the end, he will not have any clue that attack is happening. Naturally, all this stuff works in a non-secure communication (HTTP). In the following image you can see that the client is supposed to directly communicate with the server, instead all the traffic is redirected into the attacker's machine and, then, to the server: The diagram displayed in figure 7 better explain all

FiXme Note:
check the form



FiXme Note: we
should remove
the h parameter
in the figure and
let latex sor the
images, then we
should refer to
the images using
ref

Figure 7: SSL Stripping attack diagram

the temporal sequence of the attack. The user agent makes a request for a non-secure webpage, all the traffic is redirected into the attacker's virtual machine. Using iptables you redirect all the traffic from the port 80 to the port where `sslstrip` is running and, then, all the traffic is forwarded to the server. When the server response comes back, it passes again through the proxy and all the links in the webpage get downgraded from `https://` to `http://`. Finally, the forged webpage is returned to the user agent. To properly mount this attack, you have to follow same specific rules:

FiXme Note:
avoid first person
(you)

- Mount a man in the Middle attack
- Setup `sslstrip` to manipulate the HTTP traffic using this command line tool: `sslstrip -l <port>`

⁴<https://moxie.org/software/sslstrip/>

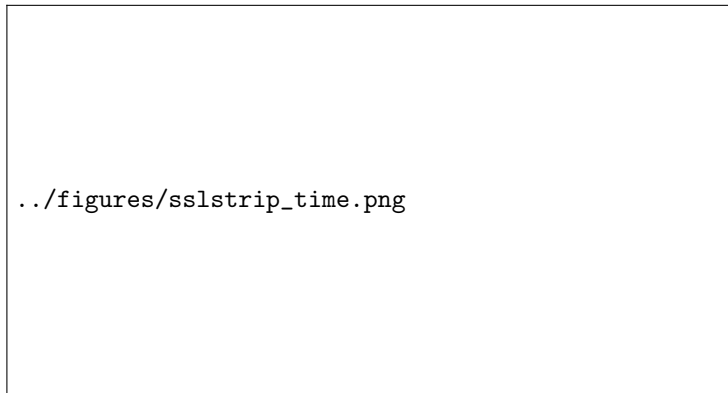


Figure 8: SSL Stripping attack diagram

- Create a specific rule to redirect the traffic from the port 80 to the port that sslstrip is running, using iptables (mettere la riga di codice?)
- Navigate to `www.homepage.it` and click to the URL within the page
- Intercept the traffic using Wireshark and analyze the behaviour of sslstrip

Countermeasures In order to prevent this kind of attack, there is a web security policy to protect against protocol downgraded attacks, called **HTTP Strict Transport Security** (HSTS). Declaring the HSTS policy, the web server forces a browser to use HTTPS and never insecure HTTP protocol. The HSTS Policy is communicated by the server to the user agent via an HTTPS response header field named *Strict-Transport-Security*.

3.3 Exercise 3 - HSTS Bypassing

The HSTS policy is associated with a specific domain name and changing just by one letter the domain name, the browser will not apply the policy anymore. For instance, an 'l' (uncapital L) could become an 'I' (capital i). The following diagram better shows the behaviour of this technique:



Figure 9

the first part is very similar to the one presented in the previous exercise, where all the links in the webpage get downgraded from `https://` to `http://`, adding for instance a fourth 'w' in the domain name (it can be whatever you want, the choice of adding a fourth w is due to the fact that it is not very visible).

When the user clicks on a link within the page the DNS queries must be manipulated as well, using the `dns2proxy` tool. The DNS request from the user passes through `dns2proxy` and is manipulated in order to be solved by the DNS server. Then, the response flows into `dns2proxy` again, which manipulate the response to redirect the user to a fake webpage and not to one the user was supposed to navigate.

To correctly mount this attack, you have to follow this rules:

- Mount a MitM attack
- Implement the missing code in `sslstrip/URLMonitor.py`
- Create a specific rule to redirect the traffic from the port 80 to the port that `sslstrip` is running, using `iptables` (mettere la riga di codice?)

- Implement the missing code in `dns2proxy.py`
- Create a specific rule to redirect all the traffic in the attacker virtual machine changing the destination ip
- Analyze the behaviour using Wireshark, being sure that all this trick has being working properly

Fixme Note: add
the solutions

Countermeasures Unfortunately, there is no default technique to prevent this type of attack. The user must always check the correctness of the URL in the address bar.