# Using Scapy for Man-in-the-Middle Attacks

Ana Rita Martinho
Department of Electrical and Computer Engineering
Faculty of Engineering of University of Porto
up201709727@fe.up.pt

Gonçalo Xavier
Department of Electrical and Computer Engineering
Faculty of Engineering of University of Porto
up201604506@fe.up.pt

***Abstract*—A Man-in-the-Middle(MITM) attack is a type of cyberattack where an attacker secretly relays and possibility alters the communication between 2 unsuspecting parties. Most recent cryptographic protocols include some form of endpoint authentication specifically to prevent MITM attacks.**

**The goal of this project is to demonstrate, using scapy, why such measurements are needed. Scapy is a python module that enables a user to send, sniff and forge network packets, as such it provides easy methods to the types of functionalities needed to run a MITM attack.**

**4 different protocols were analyzed (FTP, SNMP, Telnet and SMTP).**

## I. Introduction

This paper will focus on the work developed for the main project of the Security for Systems and Networks course.

This projects aims to demonstrate the different procedures used to carry out a MITM attack and their potential outcomes.

3 Linux Virtual Machines (VM) were used, 2 simulating normal users of the analyzed protocols - victims - and 1 running scapy, simulating the potential attacker.

The general flow of a MITM attack is as follows: **Sniffing/Evesdropping** - the attacker detects some kind of communication or vulnerable victims - **Positioning** - the attacker position itself between both victims - **Exploitation** - the attacker stores or even alters the collected information for nefarious actions.

## II. Related work

*1) Scapy:*

Scapy is a powerful python interactive packet manipulation program. It is able to forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. Scapy can easily handle most classical tasks like scanning, tracerouting, probing, unit tests, attacks or network discovery. It can replace hping, arpspoof, arp-sk, arping, p0f and even some parts of Nmap, tcpdump, and tshark.

Scapy also performs a lot of other specific tasks that most other tools can't handle, like sending invalid frames, injecting your own 802.11 frames, combining techniques (VLAN hopping+ARP cache poisoning, VOIP decoding on WEP encrypted channel, ...), etc. As such, it allows it's users to easily craft the exact packets they would like, send them out into the network and analyse their responses.

In the context of this project, the Scapy module was used as the primary tool of the developed code, specially it's custom packet building functionalities.

*2) ARP spoofing:*

As mentioned in the section I, the general MITM attack goes through 3 main steps: Evesdropping, Positioning and Exploiting. *ARP spoofing* played a major role in this project's first and middle step.

The ARP protocol is normally used to resolve the specific MAC address associated with a particular IP, so that packets can reach the intended device - translating Layer-3 addresses into Layer-2. On normal circumstances, the sending device sends an ARP Request to discover the MAC address of a particular IP. This request is then broadcasted to the network and the device with the specified IP sends an ARP Response announcing it's MAC address.

This process can be bypassed by sending an unsolicited ARP Response to a specific user, announcing that a particular IP actually belongs to the attacker's MAC address. In the most common form of *ARP spoofing*, the attacker forges an ARP Response packet declaring that it's MAC address is the one associated with the active Access Point (AP) IP. The attacker then informs the actual AP (in a similar way) that it's MAC address is the one associated with the victims IP. This way the attacker inserts itself in the middle of all communications between the victim and the network, and is able to analyse all the victim's traffic.

*3) TCP hijack:*

An alternative method to accomplish the **Positioning** step of section I is *TCP hijacking*.

During TCP's three-hand handshake, the client initially sends a random value to be used as it's sequence number (Seq.A), and the server replies with an acknowledgment number (ACK) of Seq.A + 1 and a random value value of it's own, sequence number (Seq.B). This values are then used to re-arrange out of order packets and ensure that the correct packet is received during communication.

An attacker can impersonate one of the involved parties by forging an IP packet with the correct ongoing Seq. and ACK values. If that happens, the victim's own Seq. and ACK numbers became offsetted, and their packets are dropped when

reaching the other side of the conversation. As such, the attacker is able to ”hijack” the ongoing connection.

This approach faces some problems:

- It requires a ongoing connection - upon receiving a forged SYN packet, the server will reply with a SYN ACK to the IP of said packet. Since the real device of this IP did not start a connection, it will respond with a TCP packet whose RST bit is set, immediately closing the connection.
- The server, and client became aware of the hijack - it's possible that the large quantity of sudden dropped packets trigger some kind of security alert and this may cause the connection to be dropped.

Both these problems can be somewhat dealt with by conditioning not only when this attack may occur, but also by closing the real client's connection before the hijack, without the server realizing it.

*4) RST attack:*

## III. Solution

The final product of this projects accomplishes the 0st step through a *ping broadcast* command, identifying all available hosts present in the network. It then positions itself using *ARP spoofing*, communicating to both users that the IP to which they intend to communicate with belongs to it's MAC address. Finally, it preforms 1 types of exploitation (active and passive), depending on the service being used.

Given the nature of MITM attacks, an emphasis is given to the continuous forwarding of packets between both victims, as to not alert the users that the communication has been compromised. This usually requires a fast forwarding process to simulate a normal traffic flow.

The proposed final system overcomes this difficulty using the Linux IP forwarding mechanism

## IV. Evaluation

The evaluation results go here.

## V. Conclusion

The conclusion goes here.

## References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.