Man-in-the-middle attack using Ettercap

Group 4: Linh, Manish, Mario and Mei

Lab objectives

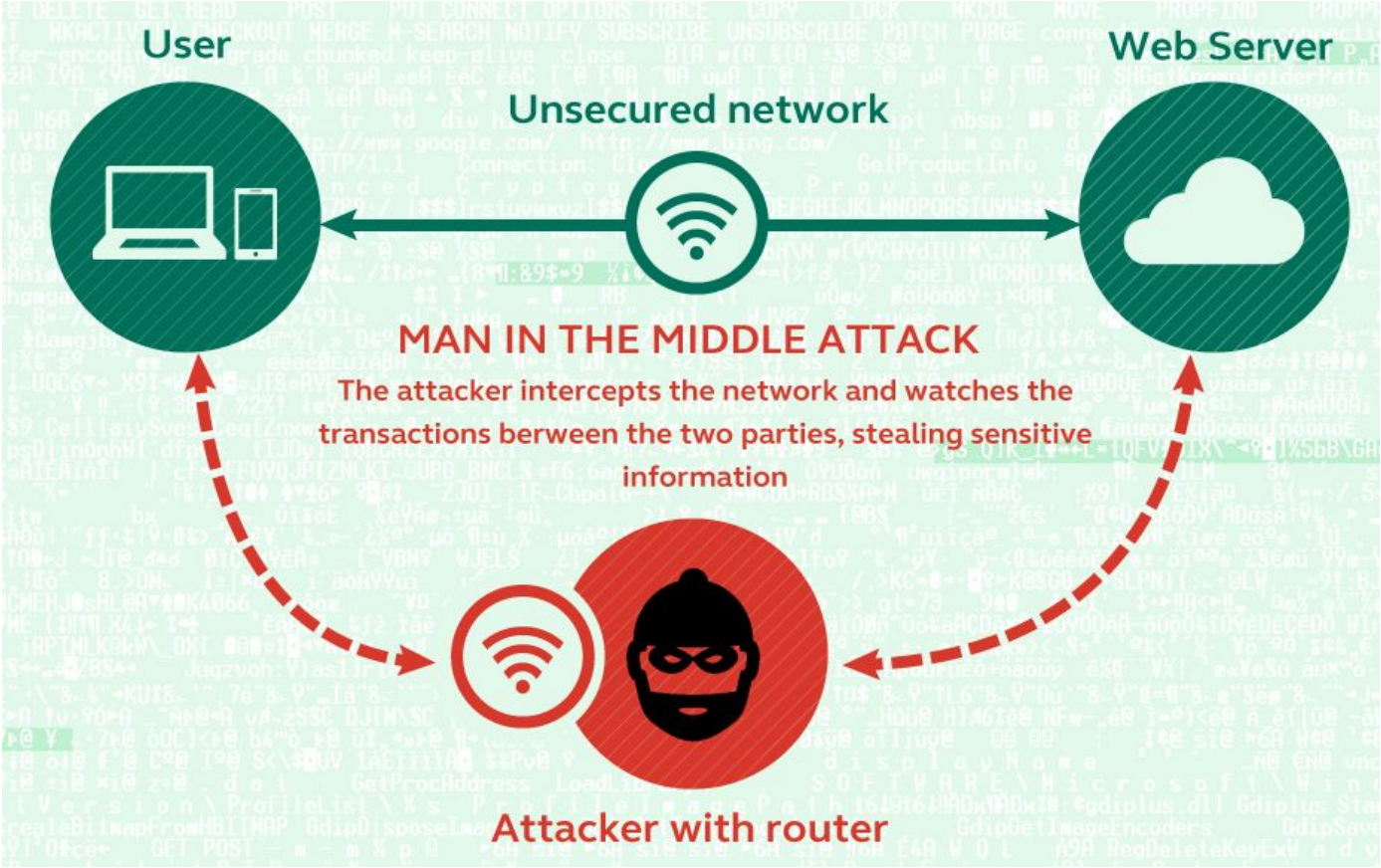
After this lab we expect all of you to know:

1. what a MITM is,
2. what are the different types of MITM attacks and
3. how you can perform them using ettercap...
4. Lastly how to save your **face** from such attacks...

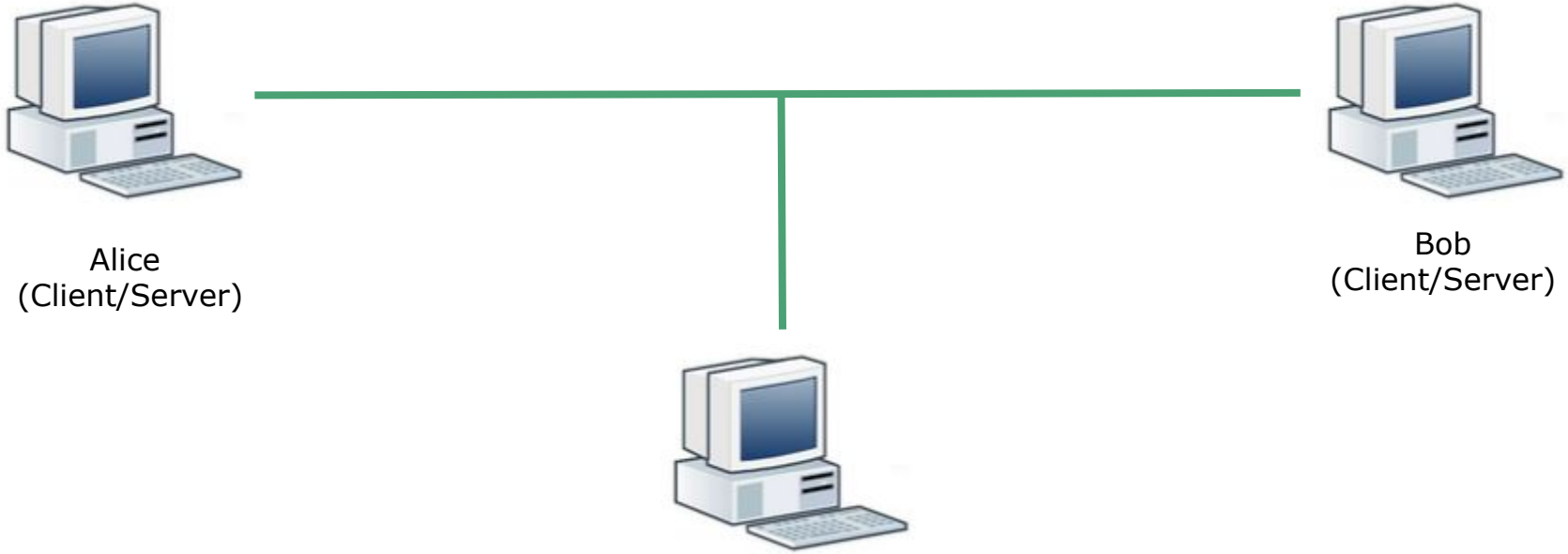
Lab Structure

- ➔ Hands-On → Make yourself home
- ➔ Listening (ARP Poisoning)
- ➔ Modification (Downgrading of SSL and SSH)
- ➔ Save your face with some countermeasures

Man-in-the-Middle


User
Web Server
Unsecured network
MAN IN THE MIDDLE ATTACK
The attacker intercepts the network and watches the transactions berween the two parties, stealing sensitive information
Attacker with router

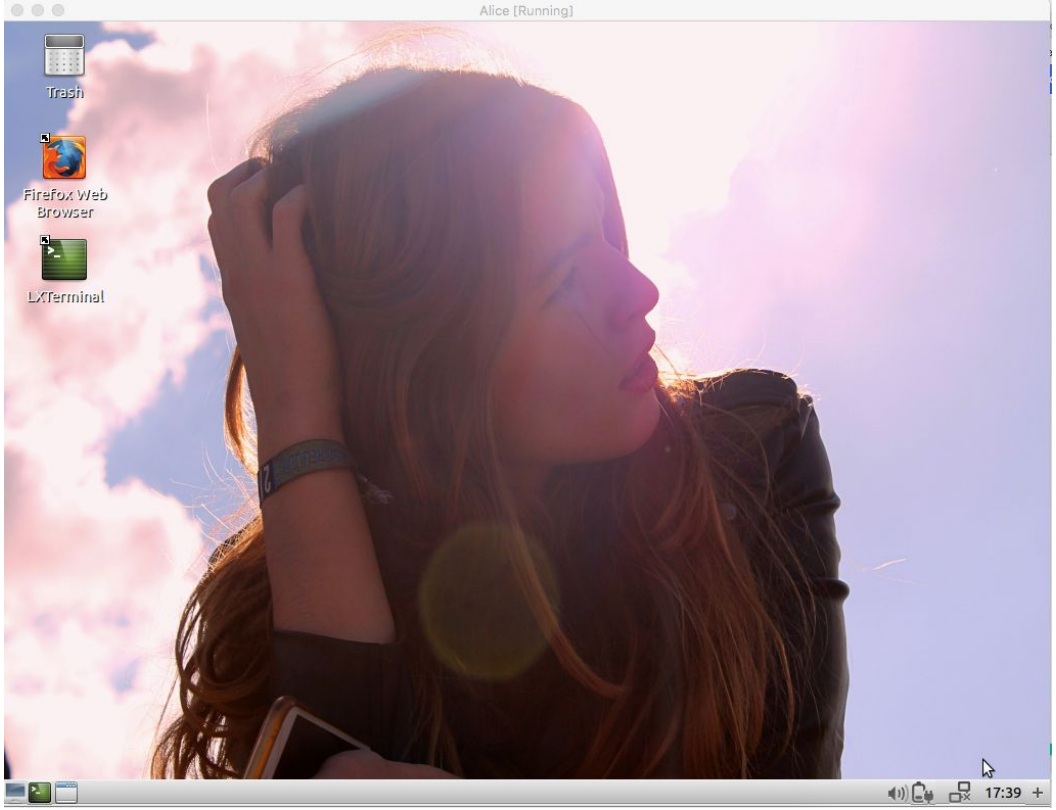
Lab setup



Victim VM → Alice

- Victim VM → Lubuntu 15.10
  - **ifconfig**
    - It should be 192.168.56.7

- root password → 123

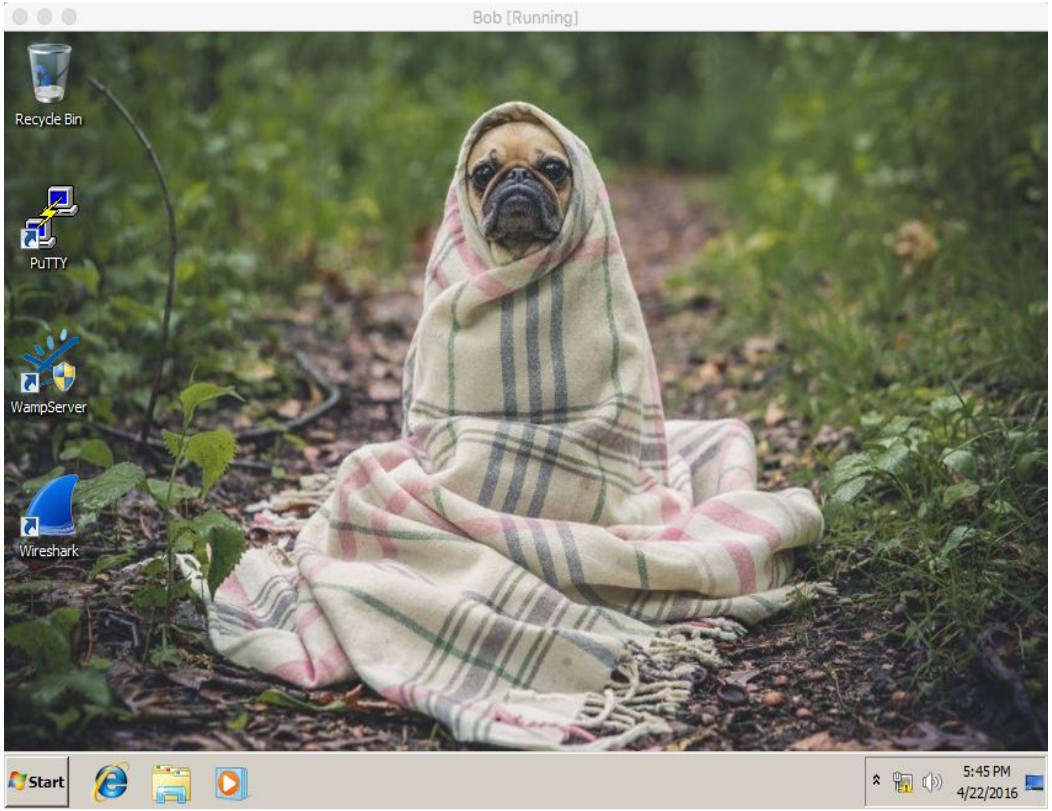


Victim VM → Bob

Victim VM → Windows 7

- Open the terminal
  - **ipconfig**
    - It should be 192.168.56.8
  - Start the WampServer

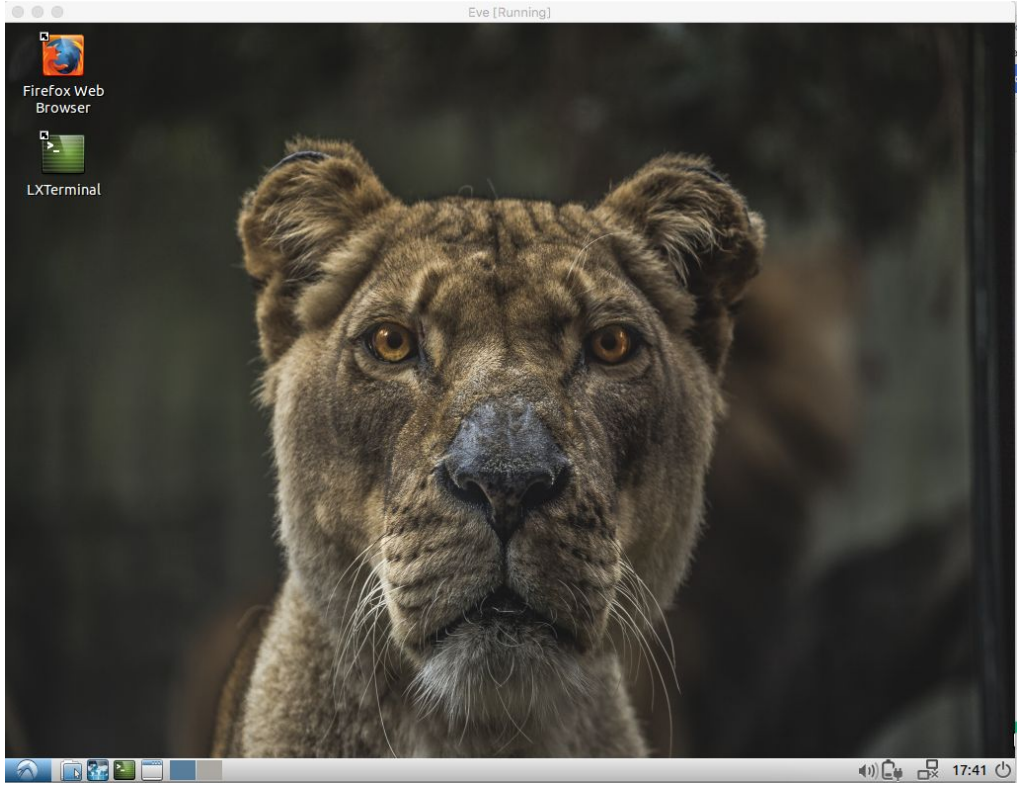
- root password → 123



Attacker VM → Eve

- Attacker VM → Lubuntu 15.10
  - **ifconfig**
    - It should be 192.168.56.9

- root password → 123



Wireshark

sudo wireshark

WireShark

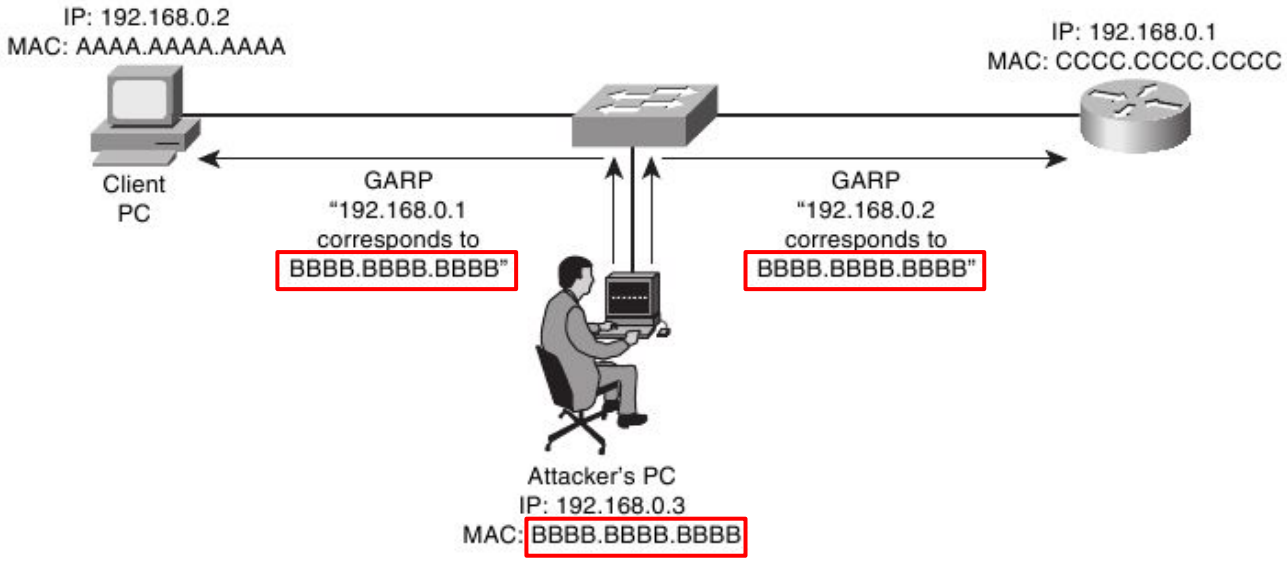
In attacker’s machine:

- Open LxTerminal from desktop and type **sudo wireshark**
- Check Interface with **ifconfig** in LxTerminal.
- Select Interface in wireshark program **(enp0s3)**
- Happy Sniffing
- Filters **(http/ssh)**
- Packets **(GET/POST)**

Wireshark

ARP Poisoning: Concept

An attacker associates his MAC address with the IP address of another host, causing any traffic meant for that IP address to be sent to the attacker instead.



ettercap

sudo ettercap -G

Overview Ettercap

Ettercap is a free and open source network security tool for man-in-the-middle attacks on LAN used for computer network protocol analysis and security auditing.

It:

- **intercepts** and alters traffic on a network segment,
- **captures** passwords,
- Has powerful (and easy to use) filtering language that **allows for custom scripting**
- conducts active **eavesdropping against a number of common protocols**: TELNET, FTP, POP, IMAP, rlogin, SSH1, ICQ, SMB, MySQL, HTTP, NNTP, X11, Napster, IRC, RIP, BGP, SOCKS 5, IMAP 4, VNC, LDAP, NFS, SNMP, Half-Life, Quake 3, MSN, YMSG!

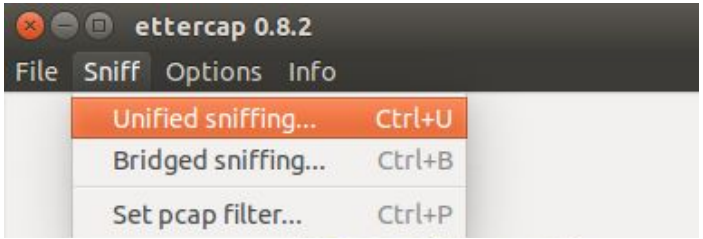
ARP Poisoning (1)

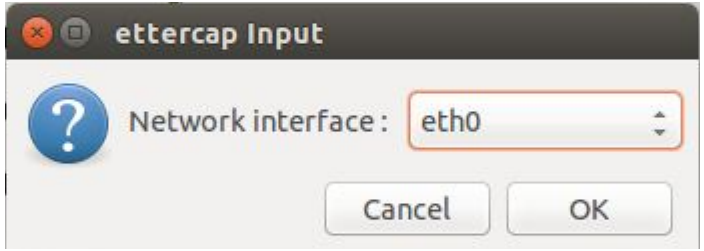
- Check **victims'** arp table before the attack
  - **arp -a**

```
mitm@mitm-VirtualBox:~$ arp -a
? (192.168.56.9) at 08:00:27:0d:ac:f4 [ether] on enp0s3
? (192.168.56.8) at 08:00:27:9d:c7:20 [ether] on enp0s3
```

ARP Poisoning (2)

- In **attacker's** machine:
  - on menu bar, click **Sniff**, → **Unified sniffing...**
  - select the proper network interface enp0s3
  - click OK





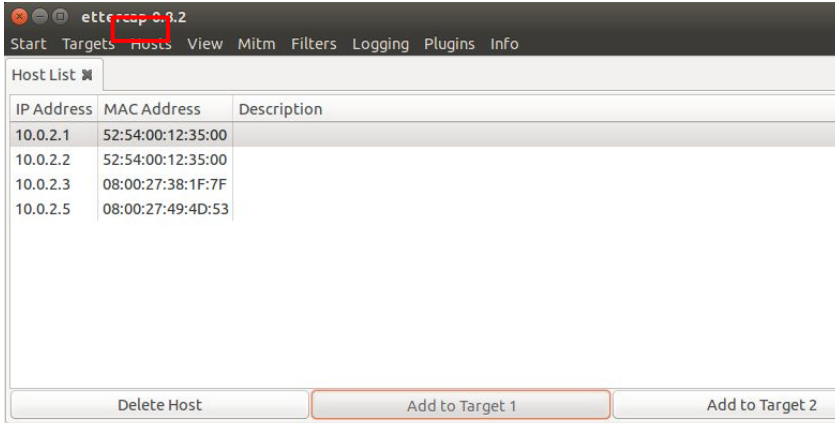
ARP Poisoning (3): attacker machine (cont.)

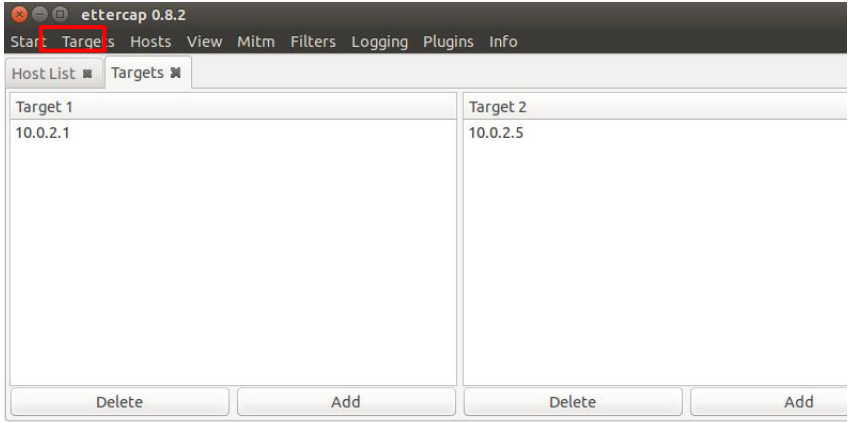
- to show list of machines connected to the interface:
  - on menu bar, click **Hosts → Scan for hosts**
  - **Hosts → Hosts list**

- to select the target for the attack
  - show hosts list
  - select **server**'s IP address, click **Add to Target 1***
  - select **victim**'s IP address, click **Add to Target 2***

- to see our current target:
  - on menu bar, click **Targets → Current targets**
  - Here we can see both the IP address in their "boxes".

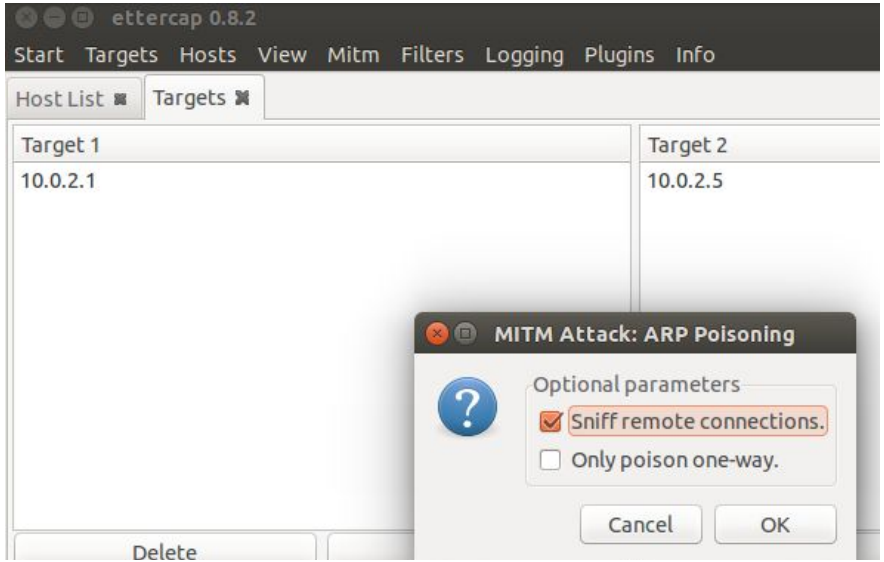
**Can be viceversa. There is no concept of SRC or DST*





ARP Poisoning (4): attacker machine (cont.)

- To perform the ARP poisoning attack:
  - on menu bar, click **Mitm → ARP Poisoning...**
  - check **Sniff remote connections**
  - click OK



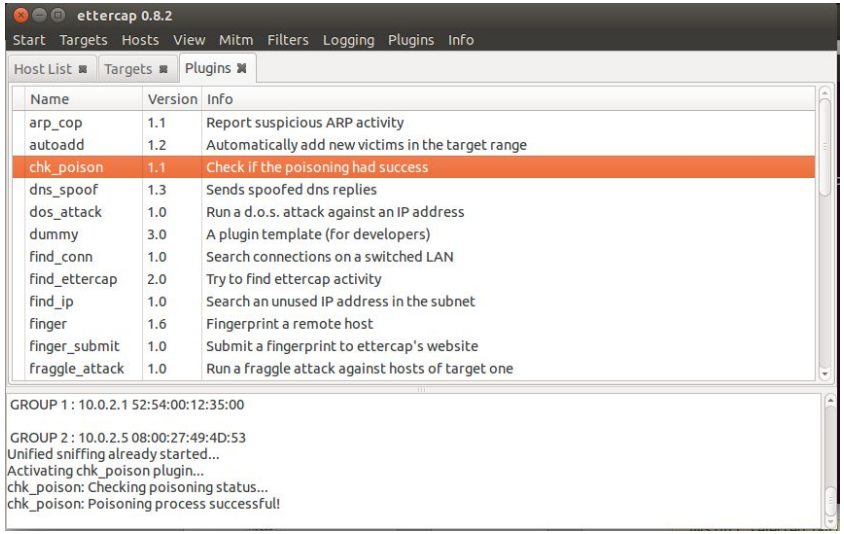
ARP Poisoning (5): Check ARP Tables

How do we know that we are successful to attack the victim?

- Check victim's ARP table
  - **arp -a**
  - victim's machine has **attacker's MAC** address for **server's IP** address

- or use the chk-poison plugin in Ettercap
  - on menu bar, click **Plugin**
  - select **chk-poison**
  - look at what ettercap says on the lowerside window

```
mitm@mitm-VirtualBox:~$ arp -a
? (192.168.56.9) at 08:00:27:0d:ac:f4 [ether] on enp0s3
? (192.168.56.8) at 08:00:27:0d:ac:f4 [ether] on enp0s3
```



ARP Poisoning (6): Sniffing the conversation

In the **attacker's** machine:

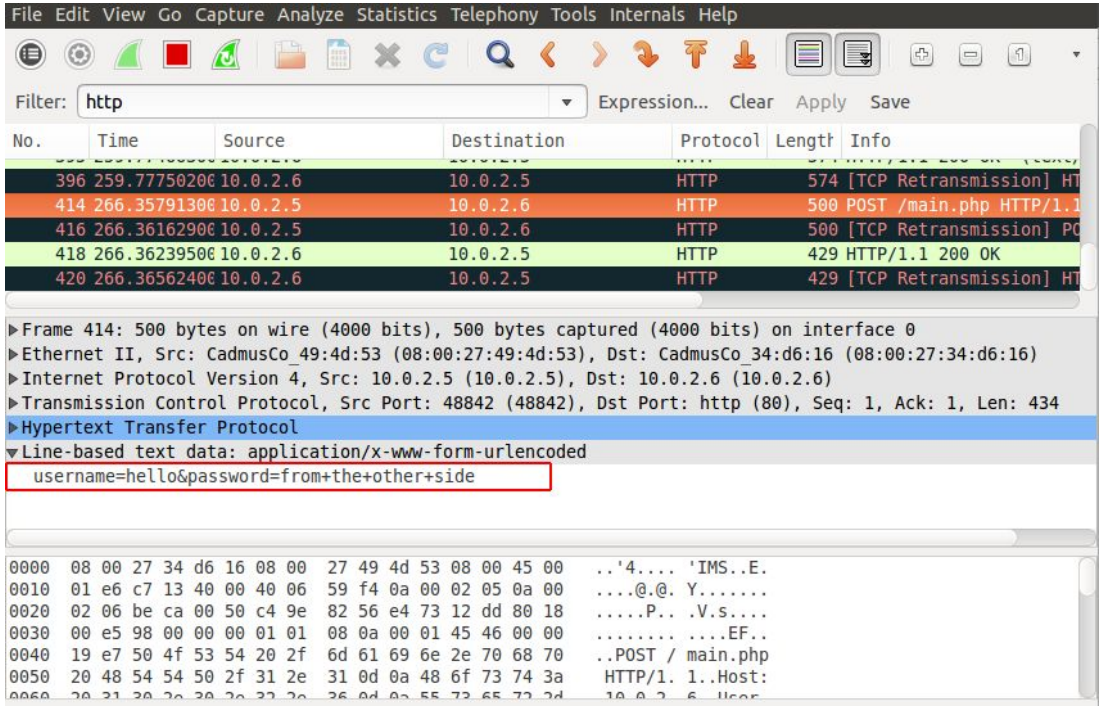
- Clean previous WireShark's results in your attacker's machine

In the victim's machine:

- open browser
- go to page: **192.168.56.8/ab**
- Enter any **firstname** and **lastname**

Username and password can be directly be seen in clear by looking at the captured packets in Wireshark.

- Look for **POST** in **Info** column to sniff firstname and lastname.



Let’s take a minute…

…and think how terrifying it is to know that you can be watched so easily while using networks like your home, or office or cafe…

But there is a good/bad news, that in real world a lot of our beloved known websites don’t work like this. They use https to secure the communication…

So as an attacker what would you do?

Example

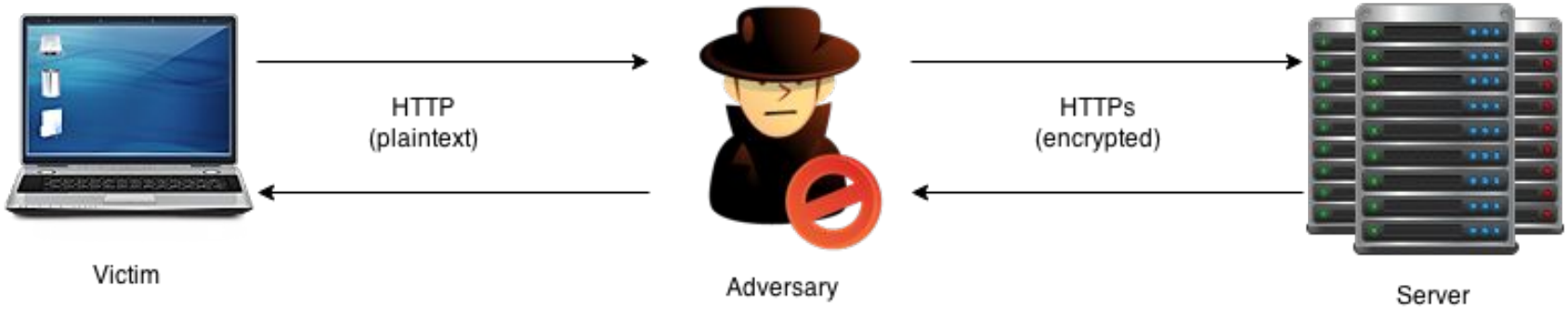
In the victim's machine:

- open browser
- go to page: **192.168.56.8/wordpress**
- go to the Login In
- enter **username** and **password**

In the attacker's machine:

- unlike before, we can't see what victim has entered.

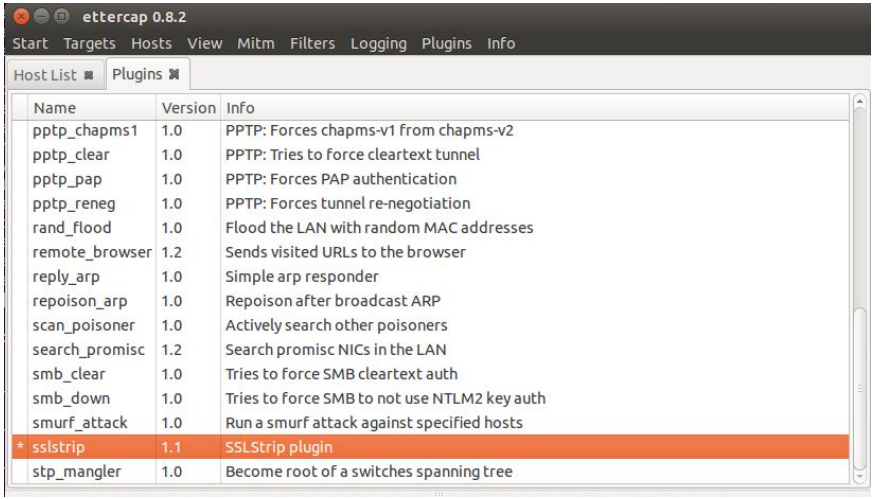
SSLStrip - Concept



SSLstrip (1): attacker’s machine

In the **attacker’s** machine:

- activate SSLstrip plugin
  - on menu bar, click **Plugins → Manage the plugins**
  - **double click** on **sslstrip** plugins
  - there is (*) sign on the left side once it is activated

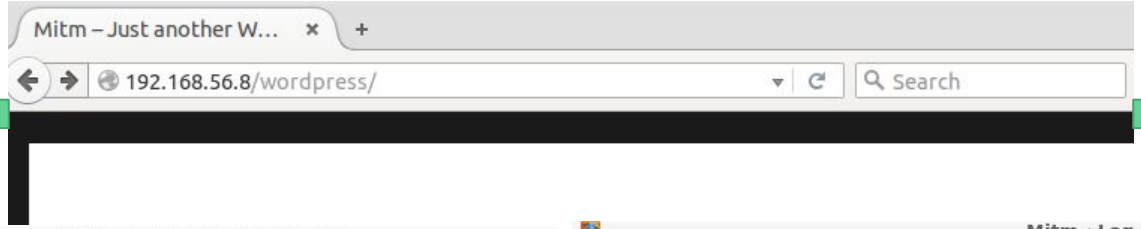


SSLstrip (2): victim’s machine

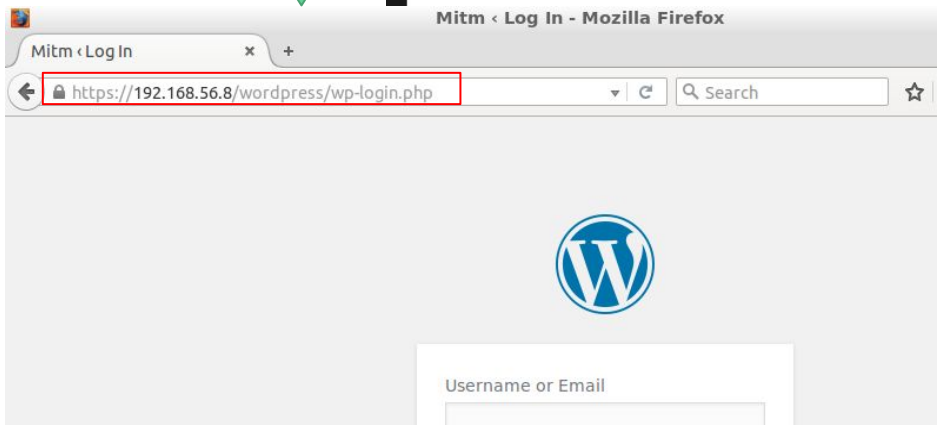
In the **victim’s** machine:

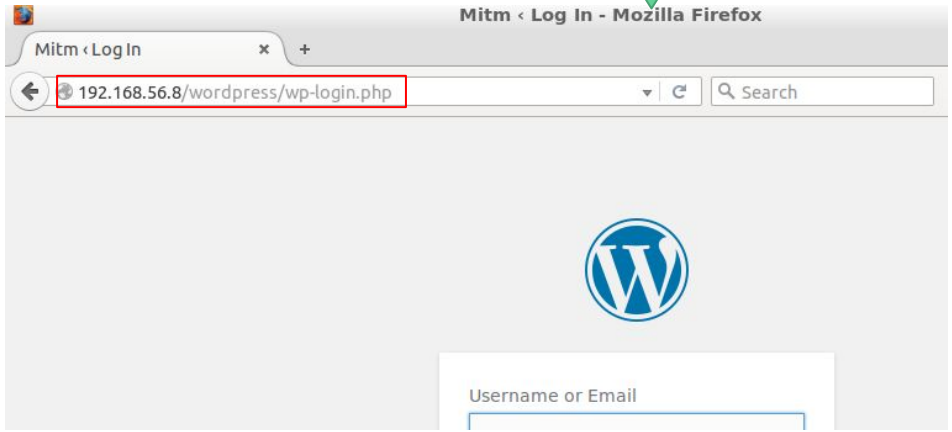
- go to wordpress’s login page
- Now, there is **no error** and it’s not going to be https, but http

Normal case: Browser redirects user to http page



SSLStrip: Converts https to http





SSLStrip (3): Sniffing conversation

Always in the **victim's** machine:

- enter **username** and **password**

As before, on **attacker's** machine:

- Wireshark will show the sniffed credentials

Ettercap Plugins

SSH Downgrade

This can be used once in "the man in the middle" position.

The principle is to downgrade a protocol version by changing data inside packets, to another version known to be vulnerable (such as SSH1 protocol).

The **client** sends a request to establish a SSH link to the server and asks it for the version it supports.

The **server** answers with either:

- ssh-2.xx ➔ The server supports only SSH2
- ssh-1.99 ➔ The server supports SSH1 and SSH2
- ssh-1.51 ➔ The server supports only SSH1

SSH Downgrade - Example

In this example:

- Alice is the **ssh server**
- Bob is the **ssh client** (putty)

In Alice's machine:

- Open terminal
    - Type **telnet 192.168.56.7 22**
    - Check if it's running
    - **If it's not running: type /etc/init.d/ssh restart**

- Let try to connect it with putty from Bob.
- Check Wireshark with **ssh filter** and you can see it's version 2!

SSH Downgrade - Solution

In the **attacker's** machine:

- Open explorer
- Go to **/usr/share/ettercap**
- Copy **etter.filter.ssh** to Desktop
- **Double click it.**

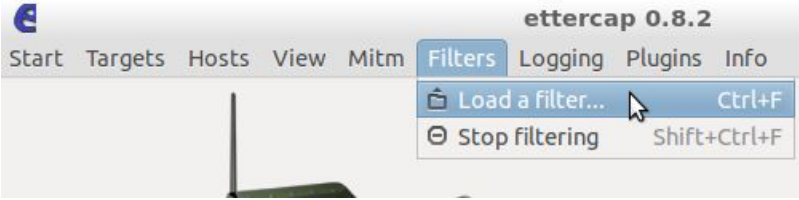
```
if (ip.proto == TCP) {
   if (tcp.src == 22) {
      if ( replace("SSH-1.99", "SSH-1.51") ) {
         msg("[SSH Filter] SSH downgraded from version 2 to 1\n");
      } else {
         if ( search(DATA.data, "SSH-2.00") ) {
            msg("[SSH Filter] Server supports only SSH version 2\n");
         } else {
            if ( search(DATA.data, "SSH-1.51") ) {
               msg("[SSH Filter] Server already supports only version 1\n");
            }
         }
      }
   }
}
```

SSH Downgrade

Ettercap offers a predefined configuration file for the SSH downgrade attack.

In **attacker's** machine:

- **cd ~/Desktop**
- compile the file to create the filter by: **etterfilter etter.filter.ssh -o etter.filter.ssh.co**
- load the filter: **Filters** --> **Load a filter...**



In the **bob's** machine:

- Perform the ssh from **putty.**

SSH Downgrade: countermeasures

How to avoid SSH downgrade attacks ?

- **SSH1 must NEVER be used on a SSH server** and SSH2 forced on the client.
- By default, only SSHv2 is enabled on the OpenSSH server while it is frequent to see SSHv1 and SSHv2 enabled on the clients such as Putty.

Let's see how we can secure the SSH client and server:

SSH server
- ➔ Open the /etc/ssh/sshd_config file
- ➔ Check that only the SSH2 protocol is enabled.

SSH client
- ➔ Open Putty
- ➔ Check that only SSH2 protocol is enabled.

ARP Poisoning - Countermeasures

Fighting effectively against ARP poisoning with efficiency is **not an easy task** because the ARP protocol provides **no possibilities to establish the authenticity of the source** of incoming packets.

Despite all, there are some ways to protect your machines against spoofers/poisoners by using:

- **Static ARP**

- **Surveillance tools** (such as Arpwatch, Ettercap or Snort IDS)

Static ARP

Static ARPing means that you manually configure IP to MAC mappings and are kept in the cache on a permanent basis (as for the communication with a known router).

So let's configure the IP address of the Server in a static way.

- In **victim's** machine:
  - Open terminal
  - **arp -s ip_server hw_address_server**

```
victim@victim-VirtualBox:~$ sudo arp -s 10.0.2.7 08:00:27:34:d6:16
[sudo] password for victim:
victim@victim-VirtualBox:~$ arp -a
? (10.0.2.7) at 08:00:27:34:d6:16 [ether] PERM on eth0
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on eth0
? (10.0.2.3) at 08:00:27:ff:33:ea [ether] on eth0
```

- Check victim's arp table
  - **arp -a**
  - Check that flag is set as **PERM**

- Try to perform the arp poisoning attack with Ettercap
  - Use **chk_poison** plugin

```
Activating chk_poison plugin...
chk_poison: Checking poisoning status...
chk_poison: No poisoning between 10.0.2.5 -> 10.0.2.1
```

Thanks for the attention!

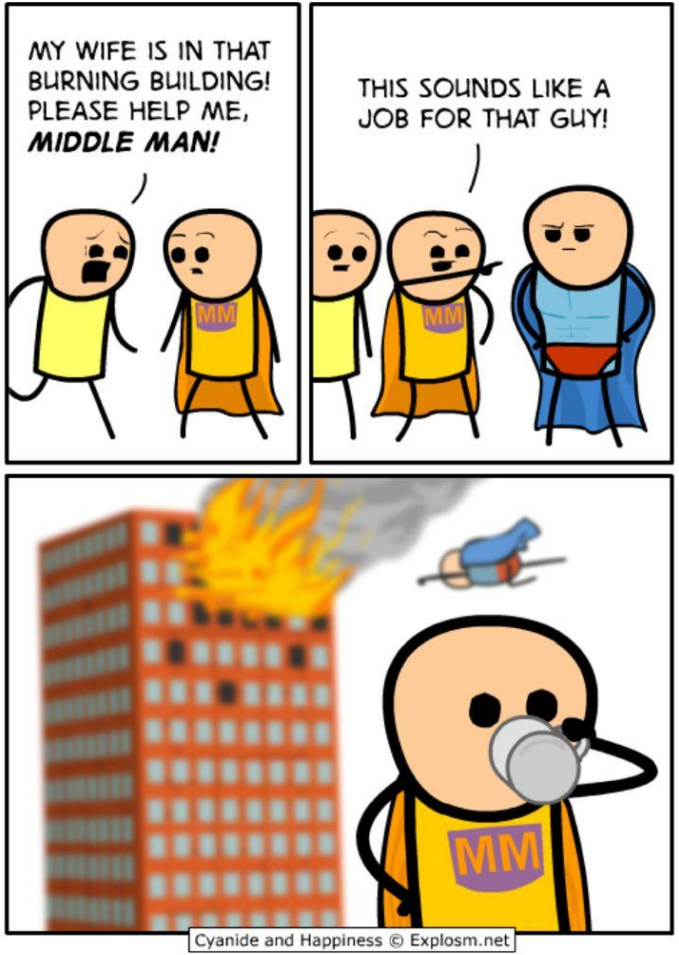


Image filtering

In **attacker's** machine

- Go to the folder filters in the Desktop
- Try to follow the steps we performed in the SSH downgrade **to compile and run**.
- Open the browser and...
- See the results!