

## Programmation avec Scapy

## ■ ■ ■ Écoute de réseau &amp; Rapport

1 – Écrire un programme qui permet de « *sniffer* » un réseau et de faire la liste des @MAC trouvées.

2 – Afficher les communications TCP qui s'établissent dans un réseau avec :

- la source ;
- la destination ;
- le service.

Vous pourrez utiliser l'annuaire inversé, fourni par :

```
1 MY_TCP_SERVICES={}
2 for proto in TCP_SERVICES.keys():
3     MY_TCP_SERVICES[TCP_SERVICES[proto]] = proto
```

## ■ ■ ■ Création de trafic et audit réseau

3 – Créez différents paquets :

- a. un paquet IP à destination de l'adresse « 192.168.0.10 » contenant un paquet TCP à destination du port 25, avec un SYN ;
- b. une trame à destination de l'@MAC « 00:10:de:ad:be:ef », contenant un paquet UDP à destination du port 53.

4 – Écrire un « *scanner* » réseau qui recense :

- a. les différentes machines d'un réseau à l'aide du protocole ICMP echo (le « *ping* ») ;
- b. les différentes machines d'un réseau à l'aide du protocole ARP ;
- c. les différentes ports ouverts sur une machine pour une liste des ports à tester pour des protocoles basés TCP (HTTP, SMTP, POP, IMAP) ;

Vous essaierez de l'étendre pour le protocole DNS, basé UDP.

5 – Protocole TCP :

- a. Programmez l'établissement d'une connexion TCP à l'aide de Scapy, en créant chacun des paquets (côté client).
- b. Est-ce que « tout se passe » sans problème ? Pourquoi ?
- c. Améliorez le programme pour récupérer le début de la conversation, comme par exemple, la bannière du serveur sur lequel on se connecte (pour réaliser du « banner grabbing »)

**Le « banner grabbing »**

Le « banner grabbing » est une technique visant à récupérer des informations concernant un système connecté à un réseau et les services qu'il propose. Nous verrons plus loin que cela correspond à la récupération de la bannière d'accueil du service, qui est envoyée lorsque l'on s'y connecte et/ou que l'on commence à échanger ; elle contient par défaut le nom du logiciel assurant le service et son numéro de version.

## ■ ■ ■ Deception & Injection

### Deception

« There can never be enough deception. »  
*Sun Tzu*

« Deceiving » : faire croire à une personne qu'une chose fausse est vraie.

- 6 – Ecrire un programme **simulant la présence d'une machine** capable de répondre par un message ICMP « Echo-reply » à la réception d'un message ICMP « Echo-request », c-à-d qui gère le « *ping* ».

```
###[ IP ]###
version  = 4L
ihl      = 5L
tos      = 0x0
len      = 84
id       = 0
flags    = DF
frag     = 0L
ttl      = 64
proto    = icmp
chksum   = 0x26a6
src      = 10.0.0.1
dst      = 10.0.0.3
\options \
###[ ICMP ]###
type     = echo-request
code     = 0
chksum   = 0x567a
id       = 0x654c
seq      = 0x1
###[ Raw ]###
load     = '\x94\x8b\x1d0\x9cZ\x03\x00...
```

```
###[ IP ]###
version  = 4
ihl      = None
tos      = 0x0
len      = None
id       = 1
flags    =
frag     = 0
ttl      = 64
proto    = icmp
chksum   = None
src      = 10.0.0.3
dst      = 10.0.0.1
\options \
###[ ICMP ]###
type     = echo-reply
code     = 0
chksum   = None
id       = 0x654c
seq      = 0x1
###[ Raw ]###
load     = '\x94\x8b\x1d0\x9cZ\x03\x00...
```

*Vous «inventerez» une machine fictive appartenant à votre réseau local.*

- 7 – Écrivez un programme qui permet de « *sniper* » une communication TCP déjà établie dans votre réseau local (« tuer » la connexion en cours).

*Vous devrez envoyer un segment TCP contenant un ACK/RST avec le bon décalage.*