1 Forger et envoyer des paquets avec Scapy

1.1 Préambule

Scapy propose aussi un mode interactif de la même manière que Python, se lançant avec la commande `$scapy`. Afin de pouvoir envoyer des trames Ethernet (couche 2), nous lancerons Scapy en mode administrateur `$sudo scapy`.

1.2 Forger

Scapy permet de construire avec une grande aisance des paquets de toutes sortes. Dans le mode interactif, essayez les commandes suivantes :

```
>>> IP()
>>> IP().show()
>>> ls(IP())
```

La fonction `IP()` construit un objet de même type et on peut conserver cet objet dans une variable en faisait par exemple :

```
>>> trame = IP()
```

On peut directement afficher un attribut. Par exemple, pour afficher la source en reprenant l'exemple précédent :

```
>>> trame.src
```

Dans Scapy, il existe des objets Python « champs », `Field`, et des objets Python « paquets », `Packet`, qui sont constitués de ces champs. Lorsque l'on construit un paquet réseau complet, il est nécessaire d'empiler les différentes couches protocolaires, `layers`. L'empilement des couches se fait avec l'opérateur « `/` ».

```
>>> p = IP()/UDP()
```

On peut aussi intervertir les différents champs à des fins expérimentales, même si c'est contraire aux protocoles.

```
>>> p = TCP()/IP()/UDP()
```

On peut aussi accéder Ã l'une des couches en utilisant l'opérateur `in`.

```
>>> p = IP()/TCP()
>>> TCP in p
    True
```

Lorsque l'on change la valeur d'un attribut, il faut faire attention au type. Par exemple, les attributs `src` et `dst` sont des chaînes de caractères, alors que `sport` et `dport` sont des entiers.

```
>>> trame = IP()/TCP()
>>> trame.dst = "192.168.1.2"
>>> trame.dport = 42
#Et en version abrégée :
>>> trame = IP(dst="192.168.1.2")/TCP(dport=42)
```

Remarques : Afin d'obtenir le calcul automatique ou la valeur par défaut de certains champs qui ont redéfinis, il faut les effacer .

```
>>> del(trame.ttl)
```

Erreur classique : Certains protocoles ont un champ qui ont le même nom (exemple dst dans Ether et IP) pour accéder à cette valeur il faut préciser

```
>>> a=Ether()/IP()
>>> a[IP].dst
```

De manière générale précisez toujours le protocole quand vous faites un test sur une valeur ou voulez la modifier.

Astuces : Pour récupérer le contenu d'une payload, on peut faire :

```
>>> payload = p.payload
```

Pour lire une entrée clavier avec Python :

```
input = raw_input("Message : ")
```

Envoie/réponses : Pour envoyer un paquet au niveau 2 on peut utiliser

```
>>> sendp(a)
```

Si vous voulez envoyer le paquet au niveau 3 il faut utiliser

```
>>> send(a)
```

Attention Scapy utilise sa propre table ARP pour savoir à qui il doit envoyer le paquet. Il est possible d'envoyer un paquet en attendant une réponse avec

```
>>> rep=sr1(a)
```

Ou srp si vous envoyez au niveau 2. Dans ce cas la réponse attendue au paquet est stockée dans rep.

1.3 Scapy

1.3.1 Importation du module

```
from scapy.all import *
```

1.3.2 Fonction sniff()

```
sniff(filter="", count=0, prn=None, lfilter=None, timeout=None, iface=All)
```

- **count** : nombre de paquets à capturer. 0 : pas de limite.
- **timeout** : stoppe le sniff après un temps donné.
- **iface** : désigne l'interface sur laquelle sniffer. La liste de vos interfaces est donnée par la commande ifconfig.
- **filter** : filtre les paquets Ã garder d'après une chaîne de caractère. Exemple : filter="port 80" filtre les paquets ayant un lien avec le port 80.
- **lfilter** : même chose, mais utilise une fonction qui revoit vrai ou faux plutôt qu'une chaîne. Exemple : lfilter=lambda x : x[1].src=='192.168.1.14' filtre les paquets émis par 192.168.1.14.
- **prn** : fonction Ã appliquer Ã chaque paquet. Si la fonction retourne quelque chose, cela s'affiche. Exemple : prn = lambda x : x.show() va afficher le détail de chaque paquet.

Exemples complets : Voici quelques exemples d'utilisation de la fonction `sniff()`.

```
from scapy.all import *

def mon_filtre(p):
    if IP in p and p[IP].src=="192.168.102.186" and p[IP].dport==8888:
        return True
    else:
        return False


def afficher(p):
        p.show()
        print p[IP].src
        if Raw in p:
                print p[Raw].load


#sniff(count=0, filter="host 192.168.102.186 and dst port 8888", prn=afficher)

sniff(count=0, lfilter=mon_filtre, prn=afficher)
```