

# Scapy : les commandes de base

## Utilisation de Scapy en interactif

Pour pouvoir manipuler les paquets réseaux, il est nécessaire d'être root pour une majorité de tâches lors du lancement de Scapy

- Lancement de l'interpréteur Python avec Scapy : `shell > root@debian:~# scapy` INFO: Please, report issues to <https://github.com/phaethon/scapy> WARNING: IPython not available. Using standard Python shell instead. Welcome to Scapy (3.0.0)

```
|
|
|
<
/code>
```

## Utilisation de Scapy dans un script Python

Pour pouvoir **utiliser** Scapy dans un **script Python**, il faut inclure la bibliothèque Scapy avec l'instruction : `python> from scapy.all import *` *==== Les commandes de base ====* \* Connaître la **liste des protocoles** supportés par Scapy : `python> >> ls()` AH : AH ARP : ARP ASN1\_Packet : None BOOTP : BOOTP CAN : CAN CookedLinux : cooked linux DHCP : DHCP options DHCP6 : DHCPv6 Generic Message) ...

Plus de 150 protocoles réseaux supportés dont Ethernet, IP, IPv6, TCP, UDP, DNS, ICMP, DHCP, ARP, BOOTP, NetBIOS, NTP, Radius, SNMP, TFTP, etc.

Pour **visualiser** les informations contenues dans un objet de protocole (avec les valeurs par défaut) : `python> >> ls(ARP)` hwtype : XShortField = (1) ptype : XShortEnumField = (2048) hwlen : ByteField = (6) plen : ByteField = (4) op : ShortEnumField = (1) hwsrc : ARPSourceMACField = (None) psrc : SourceIPField = (None) hwdst : MACField = ('00:00:00:00:00:00') pdst : IPField = ('0.0.0.0') >>

\* connaître les **fonctions de base** de Scapy (environ une vingtaine): `>> ls()` arpcachepoison : Poison target's cache with (your MAC,victim's IP) couple arping : Send ARP who-has requests to determine which hosts are up bindlayers : Bind 2 layers on some specific fields' values bridgeandsniff : Forward traffic between two interfaces and sniff packets exchanged corruptbits : Flip a given percentage or number of bits from bytes ...

\* se documenter sur une fonction de Scapy. Exemple pour la fonction **send** : `python> >> help(send)` Help on function send in module scapy.sendrecv: send(x, inter=0, loop=0, count=None, verbose=None, realtime=None, \*args, kargs) Send packets at layer 3 send(packets, [inter=0], [loop=0], [verbose=conf.verb]) → None (END)

\* se documenter sur une méthode proposée par un protocole supporté par Scapy. Exemple pour le protocole IP : `python> >> dir(IP)` ['bool', 'bytes', 'class', 'contains', 'delattr', 'delitem', 'dict', 'dir', 'div', 'doc', 'eq', 'format', 'ge', 'getattr', 'getattribute', 'getitem', 'gt', 'hash', 'init', 'iter', 'le', 'len', 'lt', 'module', 'mul', 'ne', 'new', 'rdiv', 'reduce', 'reduceex', 'repr', 'rmul', 'rtruediv', 'setattr', 'setitem', 'sizeof', 'str', 'subclasshook', 'truediv', 'weakref', 'dosummary', 'addpayload', 'addunderlayer', 'aliastypes', 'answers', 'build', 'builddone', 'buildpadding', 'buildps', 'canvasdump', 'clonewith', 'command', 'copy', 'decodepayloadas', 'defaultpayloadclass', 'delfieldval', 'display', 'dissect', 'dissectiondone', 'dobuild', 'dobuildpayload', 'dobuildps', 'dodissect', 'dodissectpayload', 'doinitfields', 'explicit', 'extractpadding', 'fieldsdesc', 'firstlayer', 'fragment', 'fromhexcap', 'getfield', 'getbyteval', 'getdictval', 'getfieldandval', 'getfieldval', 'getlayer', 'getstrval', 'guesspayloadclass', 'hashret', 'haslayer', 'hidedefaults', 'hops', 'initfields', 'initialized', 'isprivaddr', 'lastlayer', 'libnet', 'lowerbonds', 'mysummary', 'name', 'otl', 'overloadfields', 'payloadguess', 'pdfdump', 'postbuild', 'postdissect', 'postdissection', 'predissect', 'psdump', 'rawpacketcache', 'removepayload', 'removeunderlayer', 'route', 'selfbuild', 'send', 'senttime', 'setfieldval', 'show', 'show2', 'showindent', 'sprintf', 'summary', 'underlayer', 'upperbonds', 'whois']

==== Retour à Python : la bibliothèque Scapy ...

- Python : la bibliothèque Scapy pour manipuler les paquets réseau

From:

[/ - Les cours du BTS SIO](#)

Permanent link:

[/doku.php/dev/python/scapy/scapybase](#)

Last update: **2017/11/03 22:32**

