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 : <code 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 : <code python> from scapy.all import *</code> ===== *Les commandes de base* ===== * *Connaître la **liste des protocoles** supportés par Scapy : <code python > »> ls() AH : AH ARP : ARP ASN1_Packet : None BOOTP : BOOTP CAN : CAN CookedLinux : cooked linux DHCP : DHCP options DHCP6 : DHCPv6 Generic Message) ... </code>*

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) : <code 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') »> </code> * connaître les **fonctions de base** de Scapy (environ une vingtaine): <code > »> lsc() arpcachepoison : Poison target's cache with (your MAC,victim's IP) couple arping : Send ARP who-has requests to determine which hosts are up bind*layers : Bind 2 layers on some specific fields' values bridge*and*sniff : Forward traffic between two interfaces and sniff packets exchanged corrupt*bits : Flip a given percentage or number of bits from bytes ... </code> * se documenter sur une fonction de Scapy. Exemple pour la fonction **send** : <code 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) </code> * se documenter sur une méthode proposée par un protocole supporté par Scapy. Exemple pour le protocole IP : <code 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', 'reduce*ex', 'repr', 'rmul', 'rtruediv', 'setattr', 'setitem', 'sizeof', 'str', 'subclasshook', 'truediv', 'weakref', 'do*summary', 'add*payload', 'add*underlayer', 'aliastypes', 'answers', 'build', 'build*done', 'build*padding', 'build*ps', 'canvas*dump', 'clone*with', 'command', 'copy', 'decode*payload*as', 'default*payload*class', 'delfieldval', 'display', 'dissect', 'dissection*done', 'do*build', 'do*build*payload', 'do*build*ps', 'do*dissect', 'do*dissect*payload', 'do*init*fields', 'explicit', 'extract*padding', 'fields*desc', 'firstlayer', 'fragment', 'from*hexcap', 'get*field', 'getbyteval', 'getdictval', 'getfield*and*val', 'getfieldval', 'getlayer', 'getstrval', 'guess*payload*class', 'hashret', 'haslayer', 'hide*defaults', 'hops', 'init*fields', 'initialized', 'isprivaddr', 'lastlayer', 'libnet', 'lower*bonds', 'mysummary', 'name', 'ottl', 'overload*fields', 'payload*guess', 'pdfdump', 'post*build', 'post*dissect', 'post*dissection', 'pre*dissect', 'psdump', 'raw*packet*cache', 'remove*payload', 'remove*underlayer', 'route', 'self*build', 'send', 'sent*time', 'setfieldval', 'show', 'show2', 'show*indent', 'sprintf', 'summary', 'underlayer', 'upper*bonds', 'whois'] </code> ==== Retour à Python : la bibliothèque Scapy ... ====**

- Python : la bibliothèque Scapy pour manipuler les paquets réseau