Définitions des champs

Scapy

# Protocole Ethernet

|  |  |  |
| --- | --- | --- |
| **Champ** | **Champ Scapy** | **Valeur(s) possible(s)** |
| Adresse MAC source | src | Adresse MAC, format 00:00:00:00:00:00, string |
| Adresse MAC destination | dst | Adresse MAC, format 00:00:00:00:00:00, string |
| Protocole encapsulé | type | Hexa, format 0x800  Valeurs possibles courantes :  IP : 0x0800  ARP : 0x806 |

Couche Ethernet, par packet.show() :

###[ Ethernet ]###

dst = 98:fc:11:3d:30:b0

src = b4:75:0e:b1:92:a1

type = 0x800

# Protocole ARP

|  |  |  |
| --- | --- | --- |
| **Champ** | **Champ Scapy** | **Valeur(s) possible(s)** |
| Type du paquet (opcode) | op | Request : who-is (hexa/int = 1)  Reply : is-at (hexa/int = 2) |
| Adresse MAC source | hwsrc | Adresse MAC, format 00:00:00:00:00:00, string |
| Adresse IP source | psrc | Adresse IP, format 255.255.255.255, string |
| Adresse MAC destination | hwdst | Adresse MAC, format 00:00:00:00:00:00, string |
| Adresse IP destination | pdst | Adresse IP, format 255.255.255.255, string |

Couche ARP, par packet.show() :

###[ ARP ]###

hwtype = 0x1

ptype = 0x800

hwlen = 6

plen = 4

op = who-has

hwsrc = 98:fc:11:3d:30:b0

psrc = 192.168.1.1

hwdst = 00:00:00:00:00:00

pdst = 192.168.1.232

# Protocole IP

|  |  |  |
| --- | --- | --- |
| **Champ** | **Champ Scapy** | **Valeur(s) possible(s)** |
| Protocole encapsulé | proto | Request : who-is (hexa/int = 1)  Reply : is-at (hexa/int = 2) |
| Adresse IP source | src | Adresse IP, format 255.255.255.255, string |
| Adresse IP destination | dst | Adresse IP, format 255.255.255.255, string |

Couche IP, par packet.show() :

###[ IP ]###

version = 4L

ihl = 5L

tos = 0x0

len = 84

id = 994

flags =

frag = 0L

ttl = 63

proto = icmp

chksum = 0x91c

src = 192.168.1.242

dst = 172.16.0.1

# Protocole ICMP

|  |  |  |
| --- | --- | --- |
| **Champ** | **Champ Scapy** | **Valeur(s) possible(s)** |
| Type du paquet | type | Echo-Request (8), Echo-Reply(0) |
| Code du paquet | code | Hexa (dépend du type) |

Couche ICMP, par packet.show() :

###[ ICMP ]###

type = echo-request

code = 0

chksum = 0x15a0

id = 0x57b3

seq = 0x0

# Protocole UDP

|  |  |  |
| --- | --- | --- |
| **Champ** | **Champ Scapy** | **Valeur(s) possible(s)** |
| Port source | sport | Int (53,67,68, etc.) |
| Port destination | dport | Int (53,67,68, etc.) |

###[ UDP in ICMP ]###

sport = domain

dport = 27998

len = 59

chksum = 0xfd01

# Protocole DNS[[1]](#footnote-1)

|  |  |  |
| --- | --- | --- |
| **Champ** | **Champ Scapy** | **Valeur(s) possible(s)** |
|  |  |  |
| Flag Query-Response | qr | A one bit field that specifies whether this message is a query (0), or a response (1). |
| Type de requête | opcode | A four bit field that specifies kind of query in this message. This value is set by the originator of a query and copied into the response. The values are:   * 0 a standard query (QUERY) * 1 an inverse query (IQUERY) * 2 a server status request (STATUS) |
| Autorité (Authoritive Answer) | aa | Authoritative Answer - this bit is valid in responses, and specifies that the responding name server is an authority for the domain name in question section.   * 0 = No authority * 1 = Authority |
| Récursion désirée | rd | Recursion Desired - this bit may be set in a query and is copied into the response. If RD is set, it directs the name server to pursue the query recursively. Recursive query support is optional. |
| Récursion disponible | ra | Recursion Available - this is set or cleared in a response, and denotes whether recursive query support is available in the name server. |
| Zéro | z | Reserved for future use. Must be zero in all queries and responses. |
| Code réponse | RCODE | Response code - this 4 bit field is set as part of responses. The values have the following interpretation:   * 0 No error condition * 1 Format error - The name server was unable to interpret the query. * 2 Server failure - The name server was unable to process this query due to a problem with the name server. * 3 Name Error - Meaningful only for responses from an authoritative name server, this code signifies that the domain name referenced in the query does not exist. |
| Nombre de questions | QDCOUNT | an unsigned 16 bit integer specifying the number of entries in the question section. |
| Nombre de réponses | ANCOUNT | an unsigned 16 bit integer specifying the number of resource records in the answer section. |
| Nombre d’entrées pour le nom des serveurs | NSCOUNT | an unsigned 16 bit integer specifying the number of name server resource records in the authority records section. |
| Nombre d’entrées pour le nombre de ressources additionnelles | ARCOUNT | an unsigned 16 bit integer specifying the number of resource records in the additional records section. |

## DNS Response:

###[ DNS ]###

id = 48265

qr = 1L

opcode = QUERY

aa = 0L

tc = 0L

rd = 1L

ra = 1L

z = 0L

ad = 0L

cd = 0L

rcode = ok

qdcount = 1

ancount = 1

nscount = 0

arcount = 0

\qd \

|###[ DNS Question Record ]###

| qname = 'www.linuxmint.com.'

Dans cette réponse, il y a d’abord la ‘question’ à laquelle on envoie une réponse (du paquet DNS query), et ensuite, la réponse envoyée par le serveur DNS

| qtype = A

| qclass = IN

\an \

|###[ DNS Resource Record ]###

| rrname = 'www.linuxmint.com.'

| type = A

| rclass = IN

| ttl = 2602

| rdlen = 4

| rdata = '213.175.215.218'

ns = None

ar = None

1. Source : <http://tools.ietf.org/html/rfc1035>, RFC 1035 sur Domain Names, Implementations and Specifications [↑](#footnote-ref-1)