Foundational Packetry (NG)

Using SCAPY to understand
the foundations of the internet.

Kacper Why

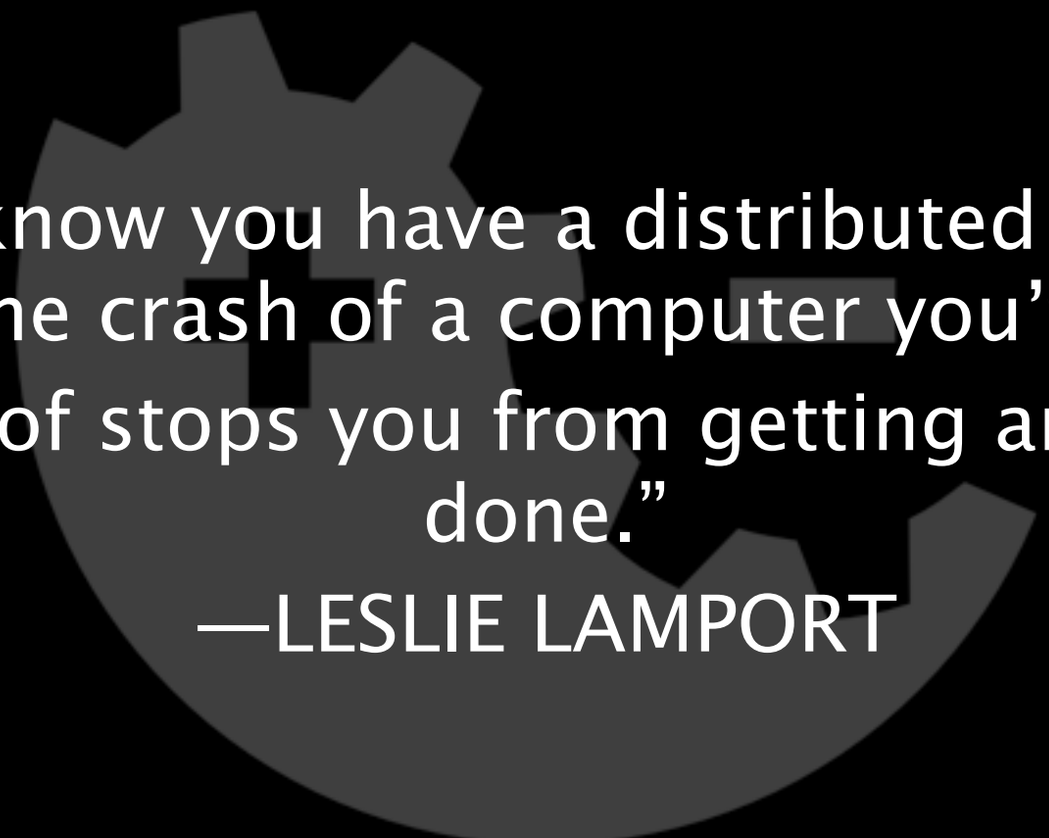
- A “serious” debugger
- Frobnosticator
- General Fault

Matt Erasmus

- An “enthusiastic amateur”
- Tinkerer
- Pretend Packet Monkey

(more likely a chaos monkey)

Why?



"You know you have a distributed system when the crash of a computer you've never heard of stops you from getting any work done."

—LESLIE LAMPORT

DAFUQ?

We’re going to understand the distributed computing of the internet.
To do that we’re going to study some simple protocols you THINK you understand.

ARP
DNS
HTTP

When you’re done you’ll be better at SCAPY, and at foundational networking.

Wireshark is installed

The Wireshark Network Analyzer [Wireshark 1.8.5]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

WIRESHARK

The World's Most Popular Network Protocol Analyzer

Version 1.8.5

Capture

Interface List

Live list of the capture interfaces (counts incoming packets)

Start

Choose one or more interfaces to capture from, then Start

- eth0
- Linux netfilter log (NFLOG) interface: nflog
- eth1
- Pseudo-device that captures on all interfaces: any

Capture Options

Start a capture with detailed options

Capture Help

How to Capture

Step by step to a successful capture setup

Network Media

Specific information for capturing on: Ethernet, WLAN, ...

Files

Open

Open a previously captured file

Wireshark: Capture Interfaces

Device	Description	IP	Packets	Packets/s
eth0		172.16.206.138	3284	285
nflog	Linux netfilter log (NFLOG) interface	none	0	0
eth1		none	0	0
any	Pseudo-device that captures on all interfaces	none	3284	285
lo		127.0.0.1	0	0

Help Start Stop Options Close

Online

Website

Visit the project's website

User's Guide

The User's Guide (online version)

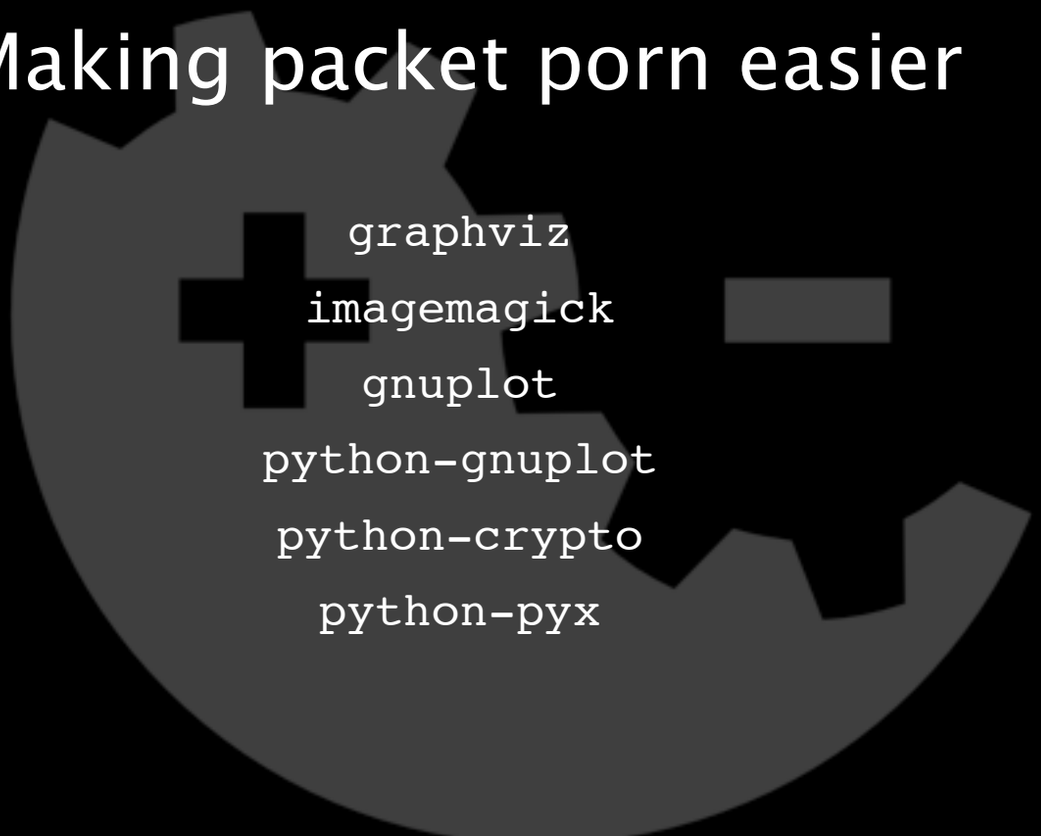
Security

Work with Wireshark as securely as possible

Ready to load or capture | No Packets | Profile: Default

Scapy is installed

Making packet porn easier



graphviz

imagemagick

gnuplot

python-gnuplot

python-crypto

python-pyx

man SCAPY

lsc()

ls(TCP)

send() / sendp()

sr() / srp()

a = rdpcap(“~/pkts/dhcp.pcap”)

a.summary()

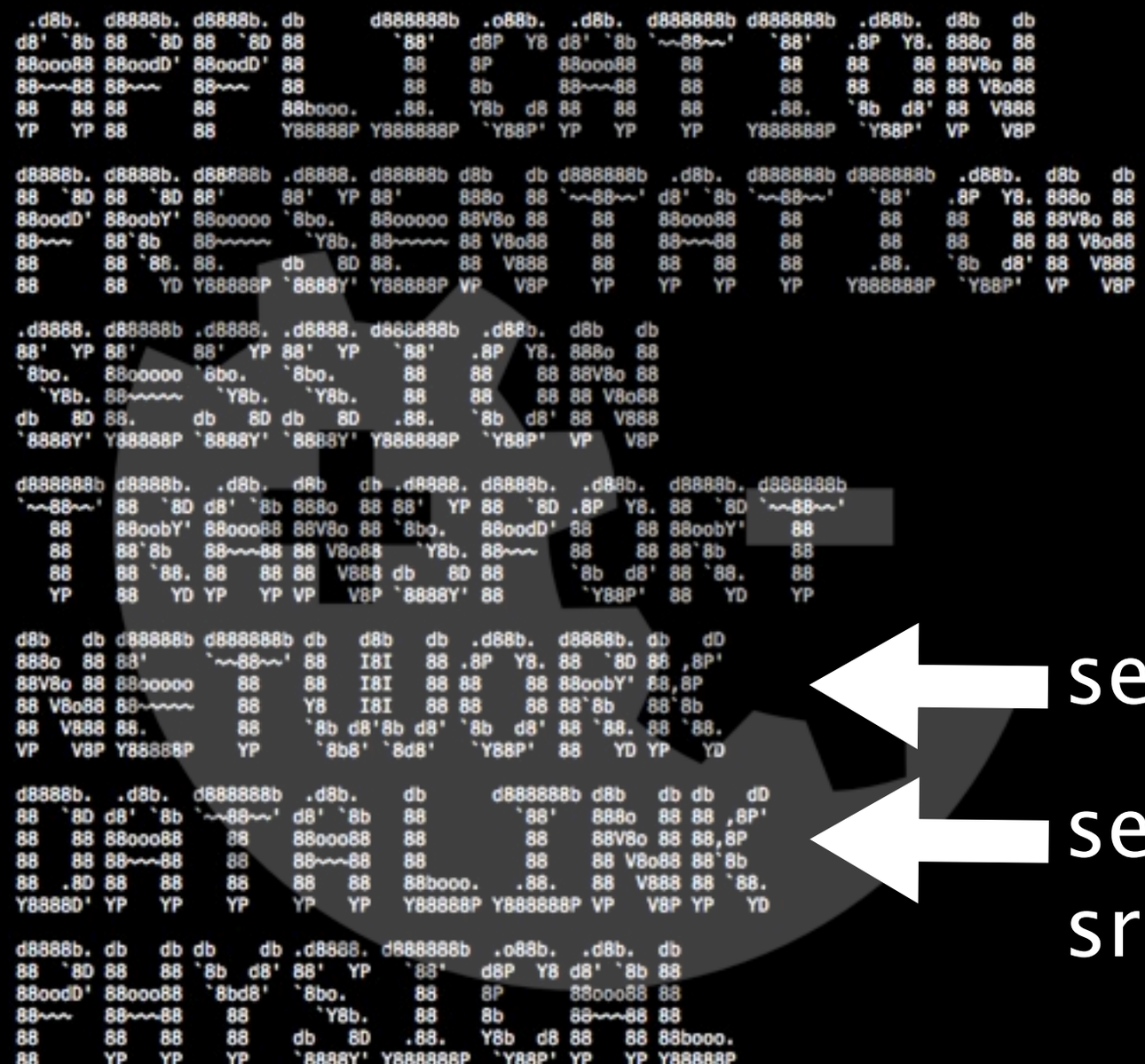
a[3].show()

a.psdump("~/pkts/dhcp.eps", layer_shift=1)

a.pdfdump(“~/pkts/dhcp.pdf”, layer_shift=1)

wrpcap(“/pkts/dhcp.pcap”, packets)

wireshark(a)



send() and sr()

sendp() and srp()

Ether / ARP Who-has

Ether / ARP Is-at

Ether / IP / UDP / DNS Query

Ether / IP / UDP / DNS Response

Ether / IP / TCP / HTTP GET

Ether / IP / TCP / HTTP RESPONSE

Less talky talky, more packety packety

```
ip = IP(dst="192.168.2.254")
ip.show()

ping = ICMP()
ping.show()
packet = ip/ping
packet.show()

fling = sr1(packet)
fling.show()
```

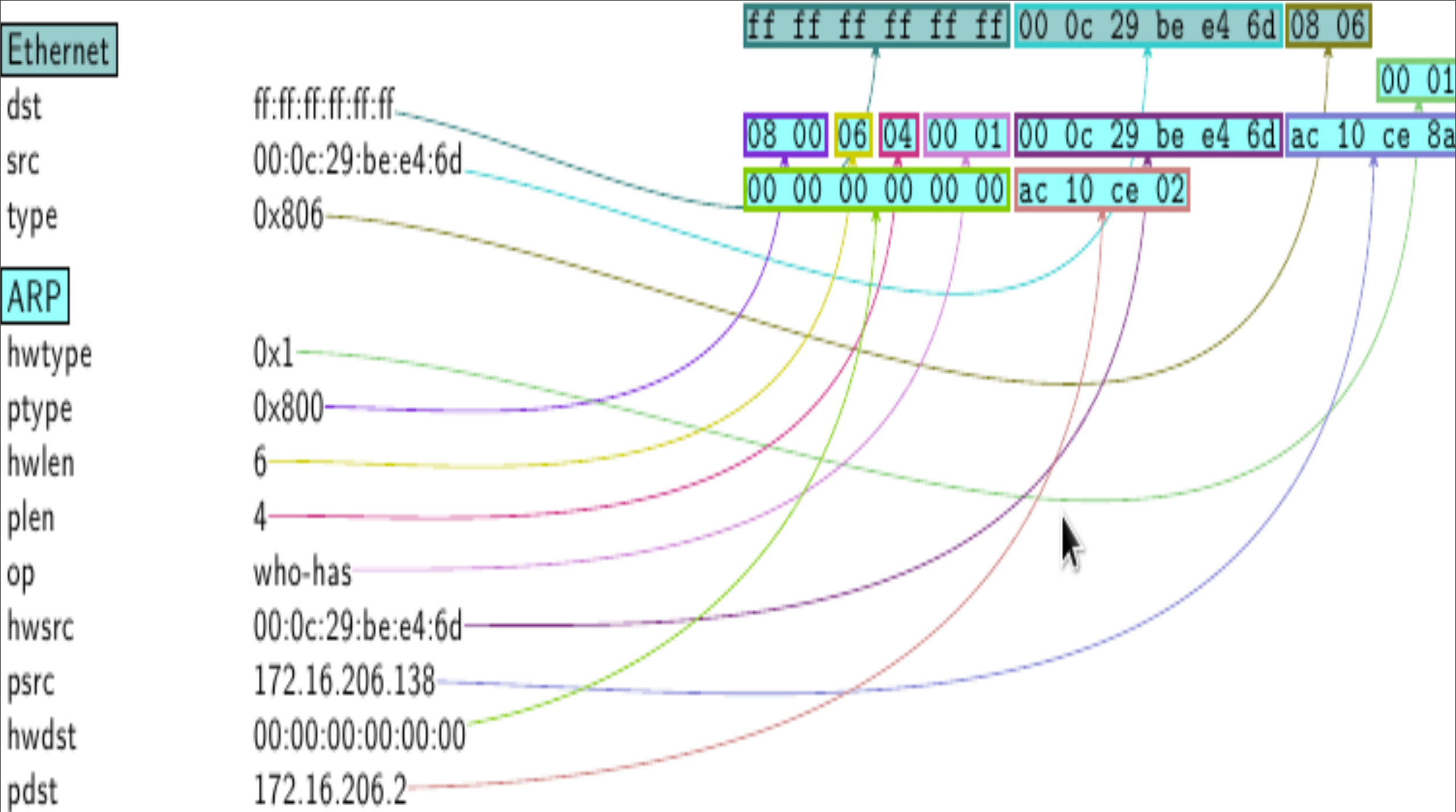
This is where Kacper show us cool stuff

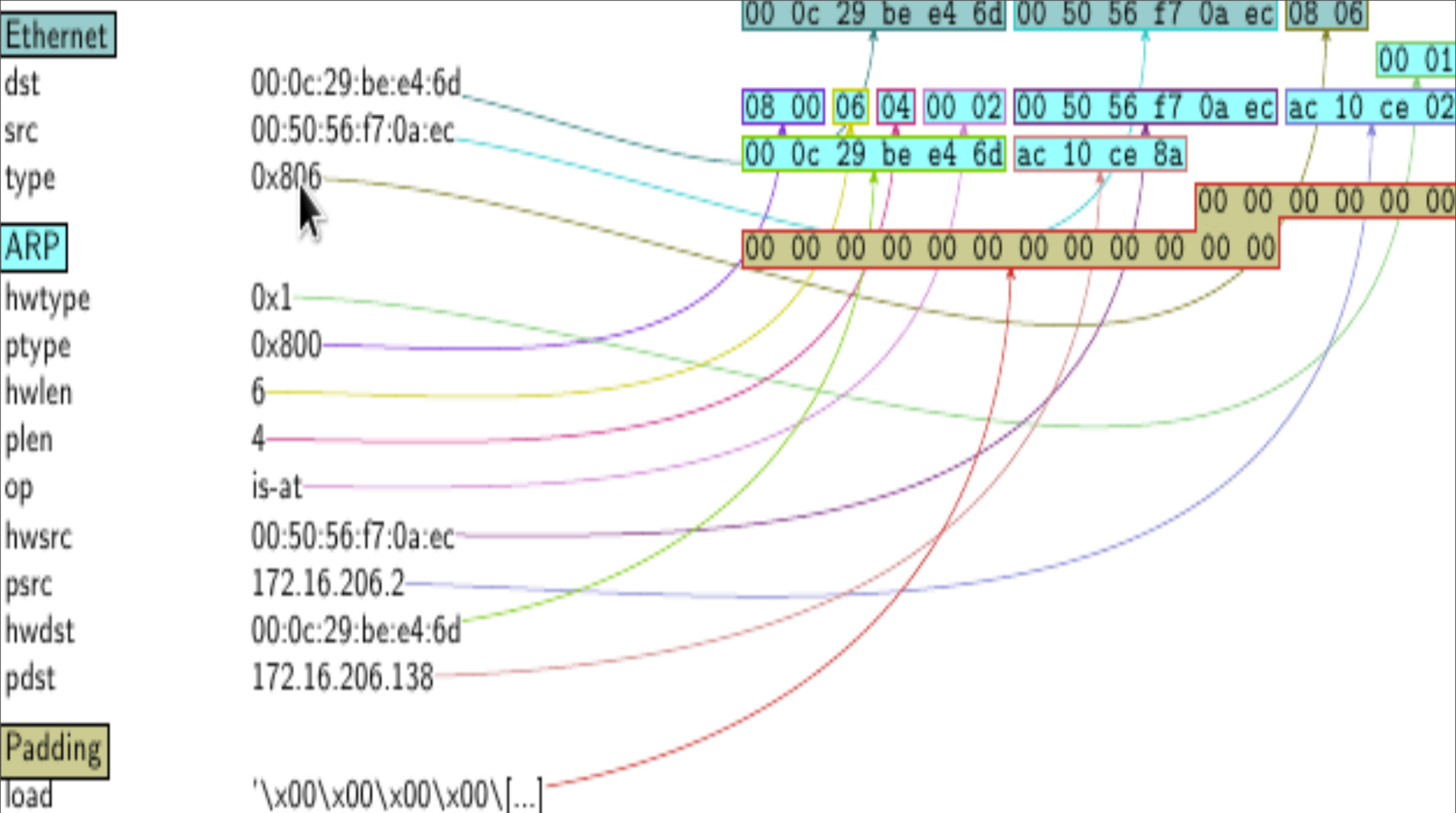
ARP Explanation

Message Types:

- Probes
- Announcements
  - Request (who-has)
  - Reply (is-at)

Resolves network layer addresses into link layer addresses


Ethernet
dst
ff:ff:ff:ff:ff:ff
src
00:0c:29:be:e4:6d
type
0x806
ARP
hwtype
0x1
ptype
0x800
hwlen
6
plen
4
op
who-has
hwsrc
00:0c:29:be:e4:6d
psrc
172.16.206.138
hwdst
00:00:00:00:00:00
pdst
172.16.206.2
ff ff ff ff ff ff
00 0c 29 be e4 6d
08 06
00 01
08 00
06
04
00 01
00 0c 29 be e4 6d
ac 10 ce 8a
00 00 00 00 00 00
ac 10 ce 02


Ethernet
dst
00:0c:29:be:e4:6d
src
00:50:56:f7:0a:ec
type
0x806
ARP
hwtype
0x1
ptype
0x800
hwlen
6
plen
4
op
is-at
hwsrc
00:50:56:f7:0a:ec
psrc
172.16.206.2
hwdst
00:0c:29:be:e4:6d
pdst
172.16.206.138
Padding
load
'\x00\x00\x00\x00\[...]
00 0c 29 be e4 6d
00 50 56 f7 0a ec
08 06
00 01
08 00
06
04
00 02
00 50 56 f7 0a ec
ac 10 ce 02
00 0c 29 be e4 6d
ac 10 ce 8a
00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00

ARP Solution

```
e = Ether(dst=”ff:ff:ff:ff:ff:ff”,type=0x806)
a = ARP(hwtype=0x1,op="who-has",hwsrc="00:0c:29:be:e4:6d",pdst="192.168.2.254")

packet = e/a

ans,unans = srp(packet, iface="eth0")
for snd,rcv in ans:
    macaddy = rcv.sprintf(r"%Ether.src%")

print "MAC address for: " + a.pdst + “is “ + macaddy
```

DNS Explanation

Given a domain get an IP address.
Can also do reverse DNS
Sometimes used for other things
Simple query -> response format
IP / UDP / DNS
DNS is considered Application Layer

```
45 00 00 3c 00 01 00 00 40 11 f8 fd c0 a8 00 61
c0 a8 00 01 00 35 00 35 00 28 e0 d2 00 00 01 00
00 01 00 00 00 00 00 00 03 77 77 77 06 62 72 75
63 6f 6e 03 6f 72 67 00 00 01 00 01
```

IP

version	4L
ihl	5L
tos	0x0
len	60
id	1
flags	
frag	0L
ttl	64
proto	udp
chksum	0xf8fd
src	192.168.0.97
dst	192.168.0.1
options	[]

UDP

sport	domain
dport	domain
len	40
chksum	0xe0d2

DNS

id	0
qr	0L
opcode	QUERY
aa	0L
tc	0L
rd	1L
ra	0L
z	0L
rcode	ok
qdcount	1
ancount	0
nscount	0
arcount	0
qd	œ[0m<œ[0mœ[31mœ[1m[...]
an	None
ns	None
ar	None

IP

version	4L
ihl	5L
tos	0x0
len	147
id	0
flags	DF
frag	0L
ttl	64
proto	udp
chksum	0xb8a7
src	192.168.0.1
dst	192.168.0.97
options	[]

UDP

sport	domain
dport	domain
len	127
chksum	0x3fc9

Raw

load	'\x00\x00\x81\x80\[...]

Padding

load	'\x00\x00\x02\x00\[...]

```
45 00 00 93 00 00 40 00 40 11 b8 a7 c0 a8 00 01
c0 a8 00 61 00 35 00 35 00 7f 3f c9 00 00 81 80
00 01 00 01 00 02 00 01 03 77 77 77 06 62 72 75
63 6f 6e 03 6f 72 67 00 00 01 00 01 03 77 77 77
06 62 72 75 63 6f 6e 03 6f 72 67 00 00 01 00 01
00 00 0e 10 00 04 58 be 10 db 06 62 72 75 63 6f
6e 03 6f 72 67 00 00 02 00 01 00 00 0e 10 00 18
04 6e 73 33 32 0d 64 6f 6d 61 69 6e 63 6f 6e 74
72 6f 6c 03 63 6f 6d 00 06 62 72 75 63 6f 6e 03
6f 72 67 00 00 02 00 01 00 00 0e 10 00 18 04 6e
73 33 31 0d 64 6f 6d 61 69 6e 63 6f 6e 74 72 6f
6c 03 63 6f 6d 00 04 6e 73 33 31 0d 64 6f 6d 61
69 6e 63 6f 6e 74 72 6f 6c 03 63 6f 6d 00 00 01
00 01 00 01 4c 3d 00 04 d8 45 b9 10
```

DNS Solution

```
packet = IP(dst="192.168.2.254")/UDP()/DNS(rd=1,qd=DNSQR(qname="2013.hack.lu"))

x = sr1(packet)
x.display()
```


“ Don’t worry darling, you didn’t burn the beer!”
Schlitz

HTTP Explanation

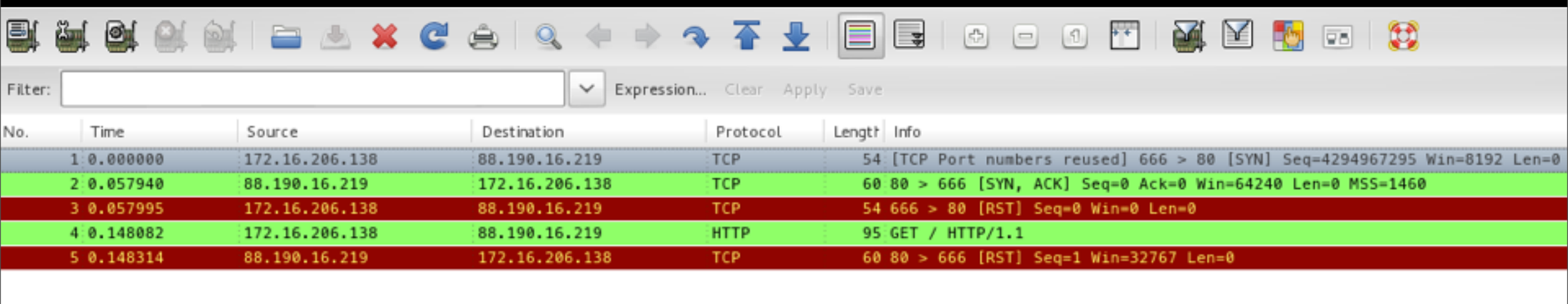
HTTP Requests:

- GET
- POST
- PUT
- DELETE
- OPTIONS
- etc. etc.

Application Level Protocol

* Stateless
* Request <-> Response
* Plaintext
* Much simpler

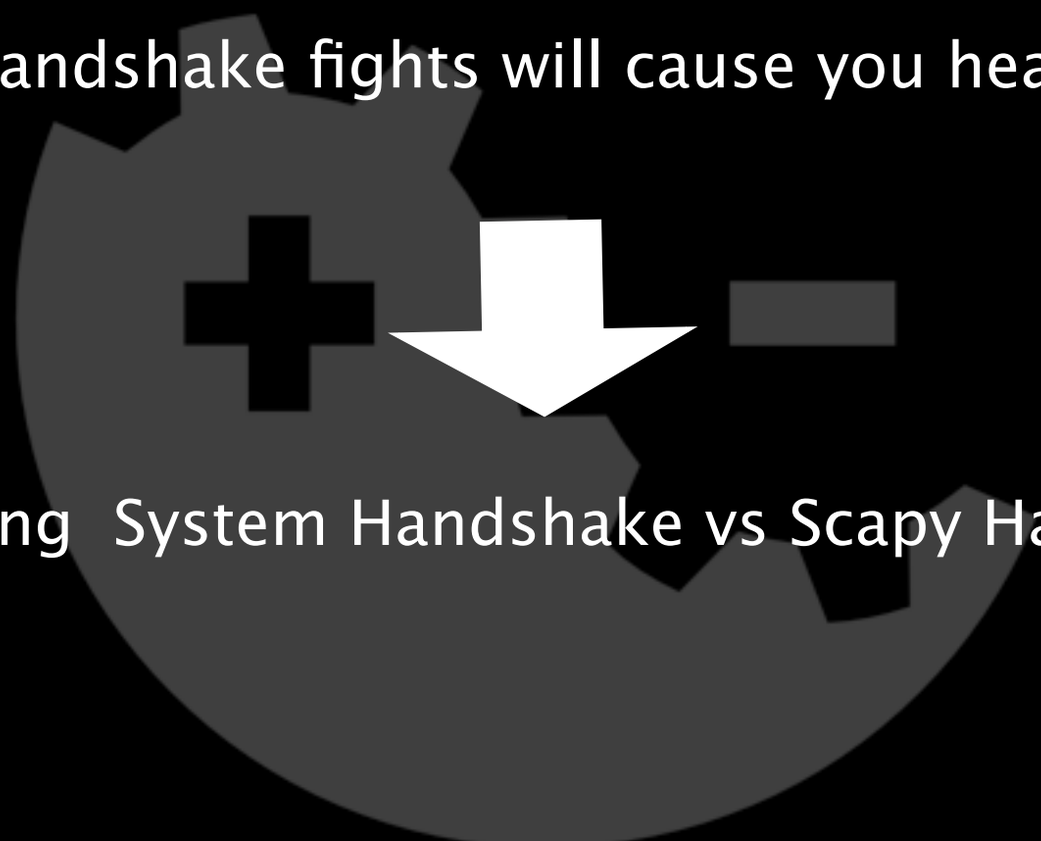
(not quite)



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.206.138	88.190.16.219	TCP	54	[TCP Port numbers reused] 666 > 80 [SYN] Seq=4294967295 Win=8192 Len=0
2	0.057940	88.190.16.219	172.16.206.138	TCP	60	80 > 666 [SYN, ACK] Seq=0 Ack=0 Win=64240 Len=0 MSS=1460
3	0.057995	172.16.206.138	88.190.16.219	TCP	54	666 > 80 [RST] Seq=0 Win=0 Len=0
4	0.148082	172.16.206.138	88.190.16.219	HTTP	95	GET / HTTP/1.1
5	0.148314	88.190.16.219	172.16.206.138	TCP	60	80 > 666 [RST] Seq=1 Win=32767 Len=0

HTTP Gotchas

TCP handshake fights will cause you headaches



Operating System Handshake vs Scapy Handshake

Ethernet

dst	00:0c:29:be:e4:6d
src	00:50:56:f7:0a:ec
type	0x800

IP

version	4L
ihl	5L
tos	0x0
len	40
id	1471
flags	
frag	0L
ttl	128
proto	tcp
chksum	0x50dd
src	88.190.16.219
dst	172.16.206.138
options	[]

TCP

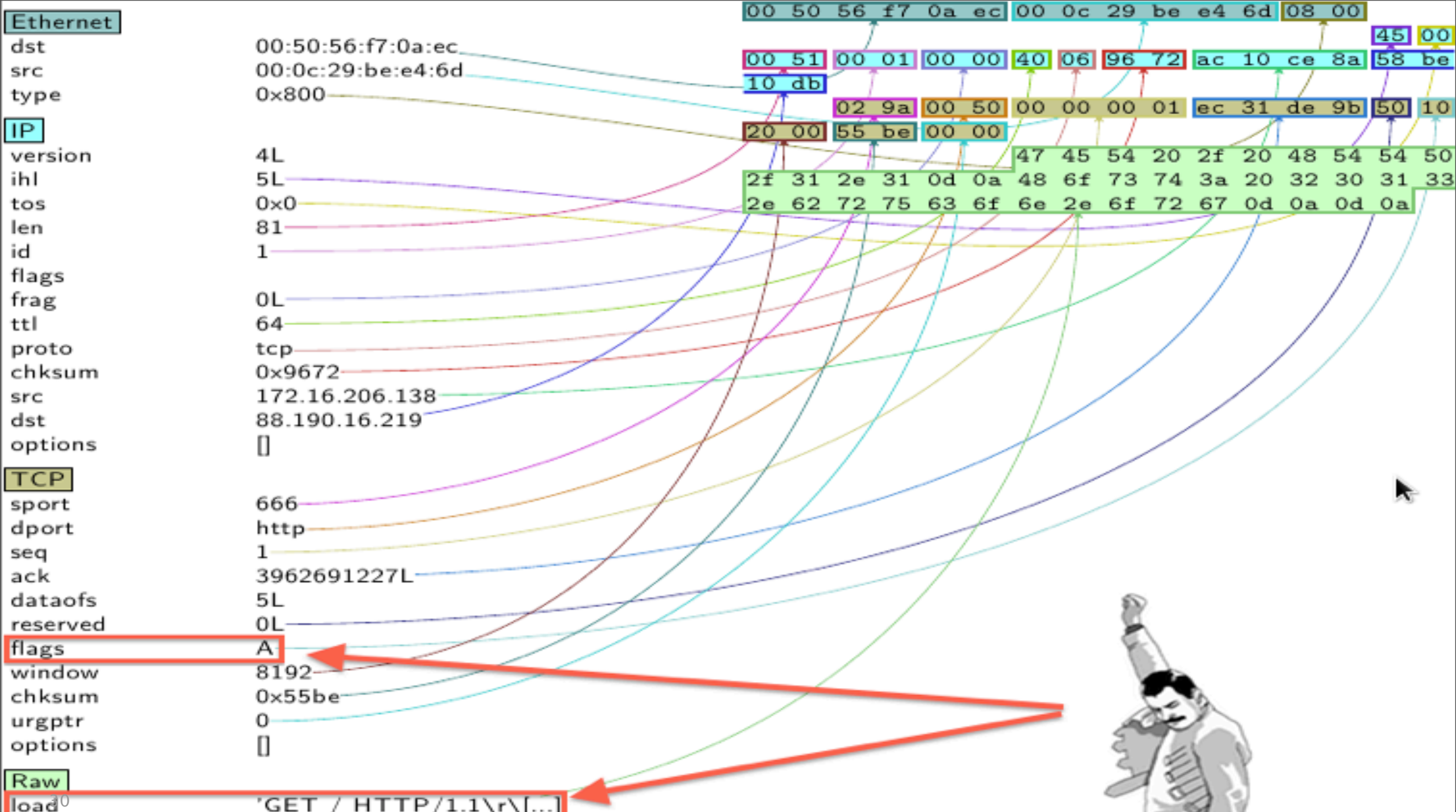
sport	http
dport	666
seq	901228188
ack	0
dataofs	5L
reserved	0L
flags	R
window	32767
chksum	0x6c6f
urgptr	0
options	{}

Padding

load	'\x00\x00\x00\x00\[...]

```
00 0c 29 be e4 6d 00 50 56 f7 0a ec 08 00
45 00
00 28 05 bf 00 00 80 06 50 dd 58 be 10 db ac 10
ce 8a
00 50 02 9a 35 b7 a6 9c 00 00 00 00 50 04
7f ff 6c 6f 00 00
00 00 00 00 00 00
```

```
iptables
-A OUTPUT
-p tcp
-d 192.168.2.254
-s 192.168.2.123
--dport 80
--tcp-flags RST RST
-j DROP
```


Ethernet
dst 00:50:56:f7:0a:ec
src 00:0c:29:be:e4:6d
type 0x800
IP
version 4L
ihl 5L
tos 0x0
len 81
id 1
flags
frag 0L
ttl 64
proto tcp
chksum 0x9672
src 172.16.206.138
dst 88.190.16.219
options []
TCP
sport 666
dport http
seq 1
ack 3962691227L
dataofs 5L
reserved 0L
flags A
window 8192
chksum 0x55be
urgptr 0
options []
Raw
load 'GET / HTTP/1.1\r\[...]
00 50 56 f7 0a ec 00 0c 29 be e4 6d 08 00
45 00
00 51 00 01 00 00 40 06 96 72 ac 10 ce 8a 58 be
10 db
02 9a 00 50 00 00 00 01 ec 31 de 9b 50 10
20 00 55 be 00 00
47 45 54 20 2f 20 48 54 54 50
2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 32 30 31 33
2e 62 72 75 63 6f 6e 2e 6f 72 67 0d 0a 0d 0a

Ethernet

dst	00:0c:29:be:e4:6d
src	00:50:56:f7:0a:ec
type	0x800

IP

version	4L
ihl	5L
tos	0x0
len	523
id	10291
flags	
frag	0L
ttl	128
proto	tcp
chksum	0x2c86
src	88.190.16.219
dst	172.16.206.138
options	[]

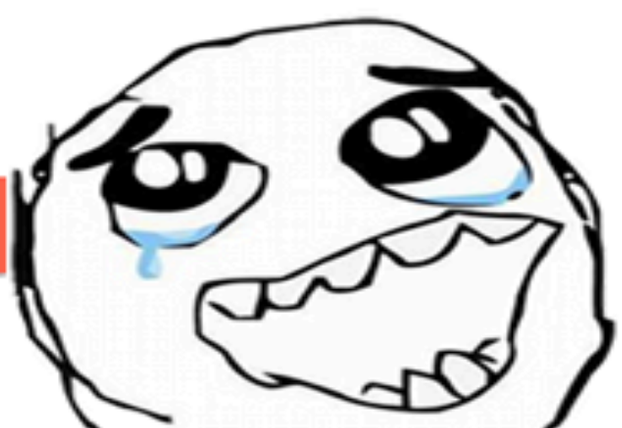
TCP

sport	http
dport	666
seq	3962691227L
ack	42
dataofs	5L
reserved	0L
flags	PA
window	64240
chksum	0x8ca3
urgptr	0
options	[]

Raw

load	'HTTP/1.1 301 Move[...]

```
00 0c 29 be e4 6d 00 50 56 f7 0a ec 08 00
45 00
02 0b 28 33 00 00 80 06 2c 86 58 be 10 db ac 10
ce 8a
00 50 02 9a ec 31 de 9b 00 00 00 2a 50 18
fa f0 8c a3 00 00
48 54 54 50 2f 31 2e 31 20 33
30 31 20 4d 6f 76 65 64 20 50 65 72 6d 61 6e 65
6e 74 6c 79 0d 0a 53 65 72 76 65 72 3a 20 41 70
61 63 68 65 0d 0a 58 2d 43 6f 6e 74 65 6e 74 2d
54 79 70 65 2d 4f 70 74 69 6f 6e 73 3a 20 6e 6f
73 6e 69 66 66 0d 0a 56 61 72 79 3a 20 41 63 63
65 70 74 2d 45 6e 63 6f 64 69 6e 67 2c 43 6f 6f
6b 69 65 0d 0a 45 78 70 69 72 65 73 3a 20 54 68
75 2c 20 30 31 20 4a 61 6e 20 31 39 37 30 20 30
30 3a 30 30 3a 30 30 20 47 4d 54 0d 0a 43 61 63
68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 70 72 69 76
61 74 65 2c 20 6d 75 73 74 2d 72 65 76 61 6c 69
64 61 74 65 2c 20 6d 61 78 2d 61 67 65 3d 30 0d
0a 4c 61 73 74 2d 4d 6f 64 69 66 69 65 64 3a 20
53 61 74 2c 20 30 37 20 53 65 70 20 32 30 31 33
20 31 31 3a 32 32 3a 34 36 20 47 4d 54 0d 0a 4c
6f 63 61 74 69 6f 6e 3a 20 68 74 74 70 3a 2f 2f
32 30 31 33 2e 62 72 75 63 6f 6e 2e 6f 72 67 2f
69 6e 64 65 78 2e 70 68 70 2f 4d 61 69 6e 5f 50
61 67 65 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70
65 78 74 2f 68 74 6d 6c 3b 20 63 68
4 3d 75 74 66 2d 38 0d 0a 54 72 61
2 2d 45 6e 63 6f 64 69 6e 67 3a 20
b 65 64 0d 0a 44 61 74 65 3a 20 53
0 37 20 53 65 70 20 32 30 31 33 20
2 3a 34 36 20 47 4d 54 0d 0a 58 2d
9 73 68 3a 20 31 34 35 37 39 30 37
a 41 67 65 3a 20 30 0d 0a 56 69 61
1 20 76 61 72 6e 69 73 68 0d 0a 43
3 74 69 6f 6e 3a 20 6b 65 65 70 2d
5 0d 0a 0d 0a
```



HTTP Solution

```
i = IP(dst="2013.hack.lu")
t = TCP(sport=2203, dport=80, flags='S')
syn_packet =  i/t

syn_ack = sr1(syn_packet)

i = IP(dst="www.hack.lu")
t = TCP(dport=80, sport=syn_ack[TCP].dport, \ seq=syn_ack[TCP].ack,
ack=syn_ack[TCP].seq + 1, flags='A')
request =  i/t/"GET / HTTP/1.1\r\nHost: 2013.hack.lu\r\n\r\n"
reply = sr1(request)

reply.display()
```

So, In Summary...

Ether / ARP who has 192.168.2.254 says 192.168.2.253

Ether / ARP is at 00:50:56:f7:0a:ec says 192.168.2.254 / Padding

Ether / IP / UDP / DNS Qry "2013.hack.lu."

Ether / IP / UDP / DNS Ans "192.168.2.254"

Ether / IP / TCP 192.168.2.253:666 > 192.168.2.254:http S

Ether / IP / TCP 192.168.2.254:http > 192.168.2.253:666 SA / Padding

Ether / IP / TCP 192.168.2.253:666 > 192.168.2.254:http A / Raw

Ether / IP / TCP 192.168.2.254:http > 192.168.2.253:666 A / Padding

Ether / IP / TCP 192.168.2.254:http > 192.168.2.253:666 PA / Raw

Ether / IP / TCP 192.168.2.254:http > 192.168.2.253:666 PA / Raw / Padding

Ether / IP / TCP 192.168.2.254:http > 192.168.2.253:666 PA / Raw

http://ctf.zonbi.org/

matt@zonbi.org

Thank you !
Questions ?
BEER !!!