Sniffing the wire

Data analysis of packet captures

Packet capture is the interception or **capture** of data **packets** moving through a computer network. After being captured, **packets** are then analyzed to help diagnose and solve network and application performance and reliability problems.

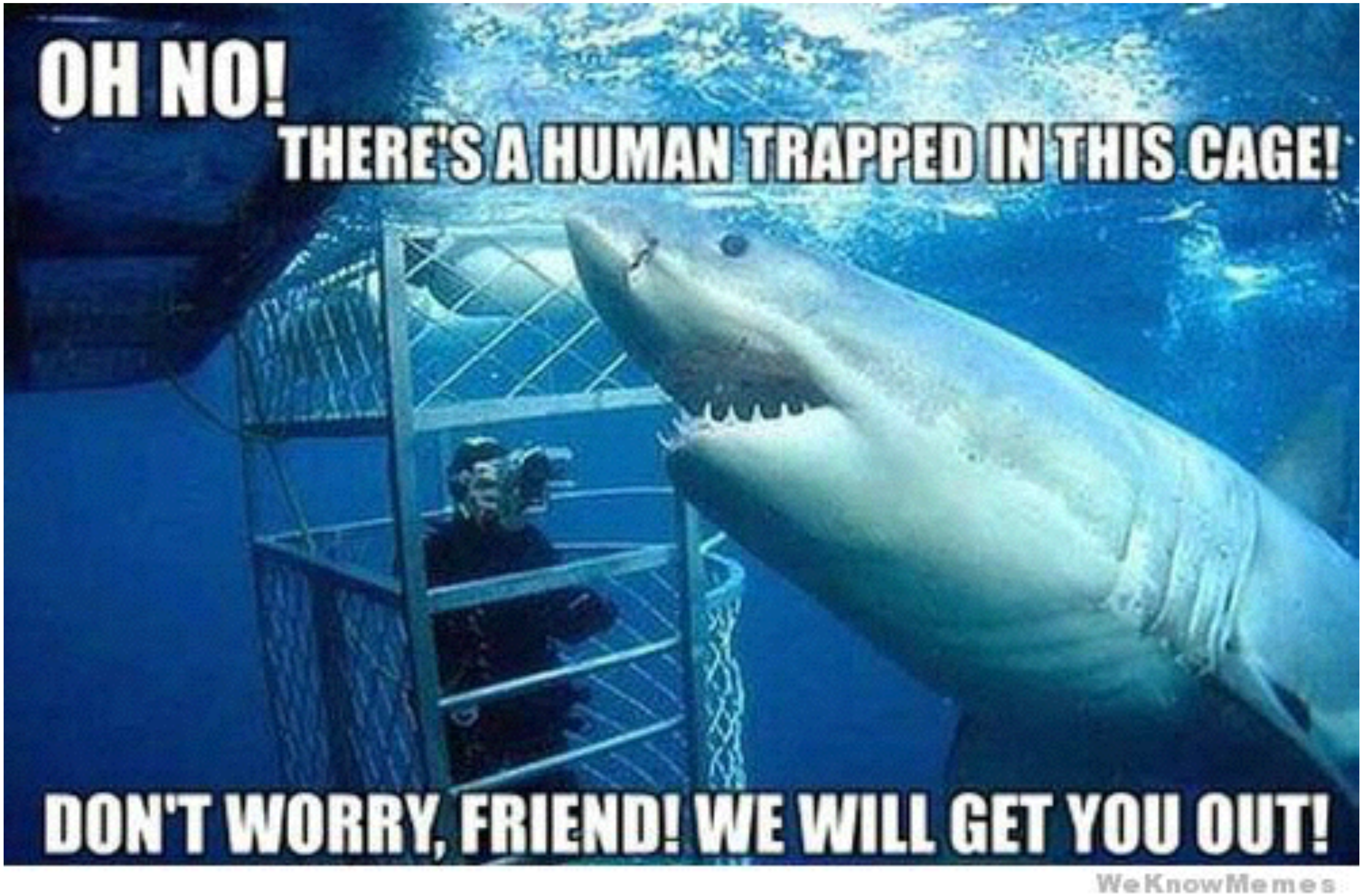
Popular Tools

Tcpdump
  Unix-based command-line tool used to intercept packets

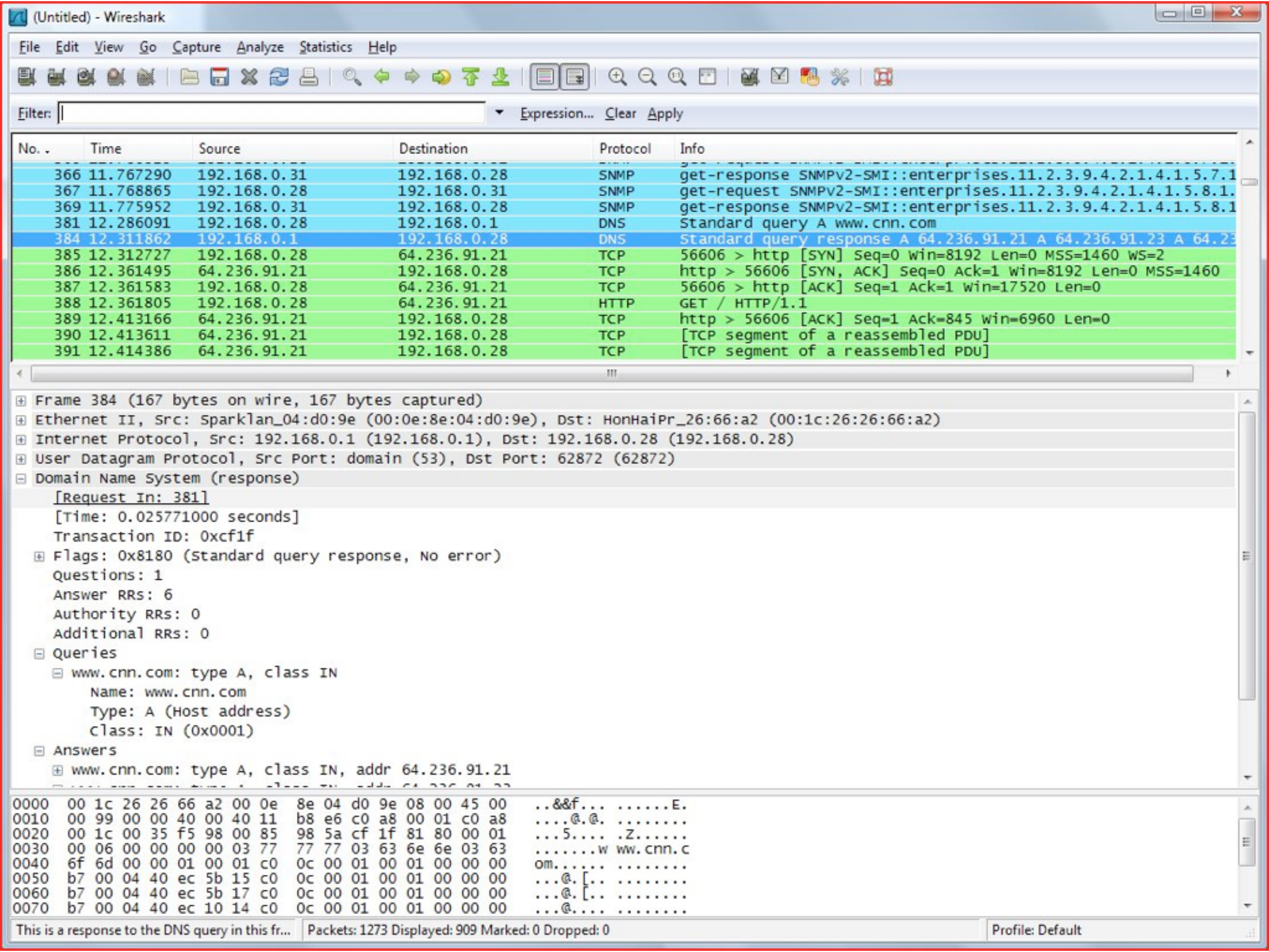
Wireshark
  GUI for displaying tcpdump/tshark packet traces

Tshark
Tcpdump-like capture program that comes w/ Wireshark
  Very similar behavior & flags to tcpdump

More available at http://sectools.org/tag/sniffers/


OH NO!
THERE'S A HUMAN TRAPPED IN THIS CAGE!
DON'T WORRY, FRIEND! WE WILL GET YOU OUT!
WeKnowMemes

Wireshark

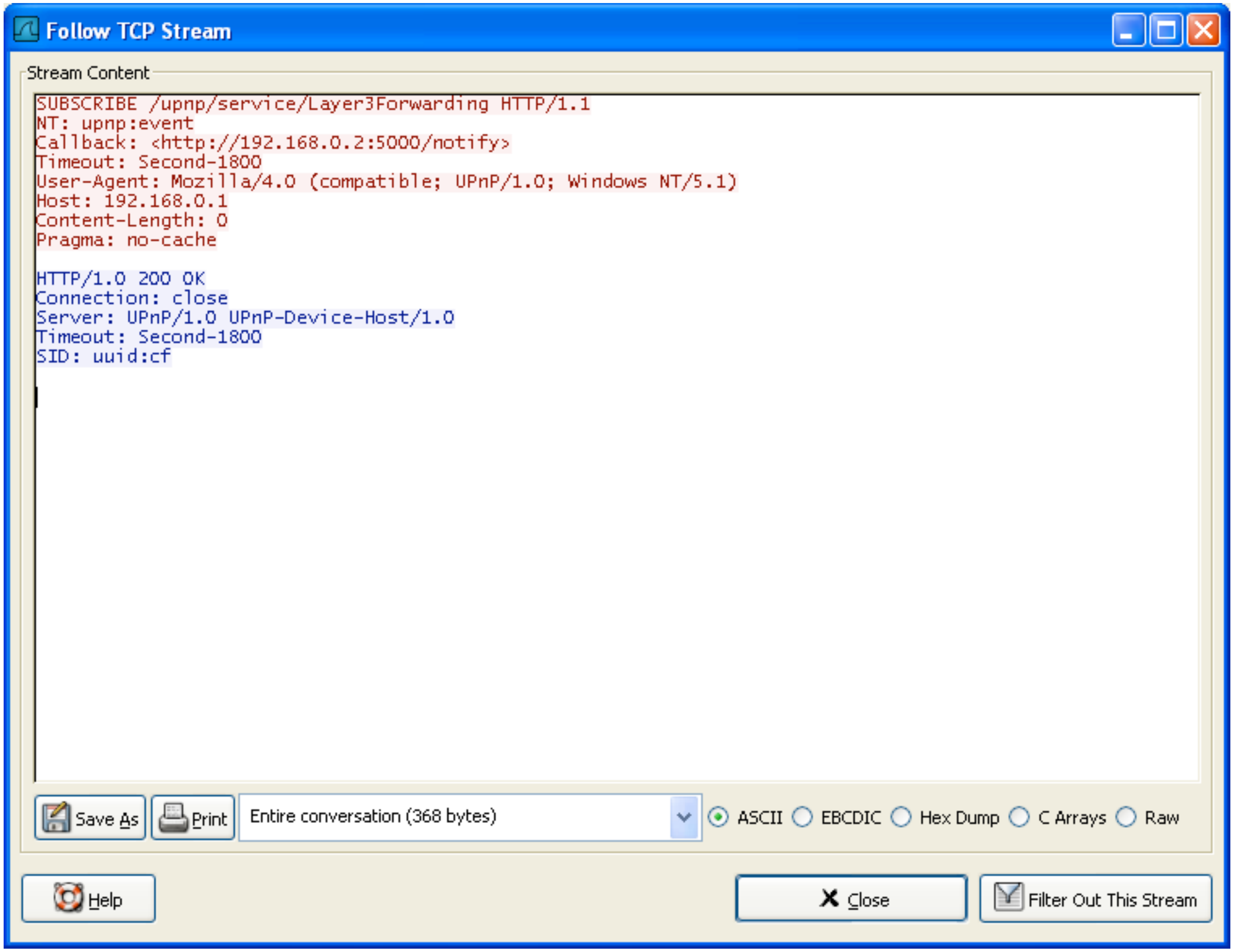

(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: Expression... Clear Apply

```
No. .  Time       Source        Destination   Protocol Info
 366 11.767290  192.168.0.31  192.168.0.28  SNMP  get-response SNMPv2-SMI::enterprises.11.2.3.9.4.2.1.4.1.5.7.1
 367 11.768865  192.168.0.28  192.168.0.31  SNMP  get-request SNMPv2-SMI::enterprises.11.2.3.9.4.2.1.4.1.5.8.1.
 369 11.775952  192.168.0.31  192.168.0.28  SNMP  get-response SNMPv2-SMI::enterprises.11.2.3.9.4.2.1.4.1.5.8.1
 381 12.286091  192.168.0.28  192.168.0.1   DNS   Standard query A www.cnn.com
 384 12.311862  192.168.0.1   192.168.0.28  DNS   Standard query response A 64.236.91.21 A 64.236.91.23 A 64.23
 385 12.312727  192.168.0.28  64.236.91.21  TCP   56606 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
 386 12.361495  64.236.91.21  192.168.0.28  TCP   http > 56606 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460
 387 12.361583  192.168.0.28  64.236.91.21  TCP   56606 > http [ACK] Seq=1 Ack=1 Win=17520 Len=0
 388 12.361805  192.168.0.28  64.236.91.21  HTTP  GET / HTTP/1.1
 389 12.413166  64.236.91.21  192.168.0.28  TCP   http > 56606 [ACK] Seq=1 Ack=845 Win=6960 Len=0
 390 12.413611  64.236.91.21  192.168.0.28  TCP   [TCP segment of a reassembled PDU]
 391 12.414386  64.236.91.21  192.168.0.28  TCP   [TCP segment of a reassembled PDU]
```

```
⊞ Frame 384 (167 bytes on wire, 167 bytes captured)
⊞ Ethernet II, Src: Sparklan_04:d0:9e (00:0e:8e:04:d0:9e), Dst: HonHaiPr_26:66:a2 (00:1c:26:26:66:a2)
⊞ Internet Protocol, Src: 192.168.0.1 (192.168.0.1), Dst: 192.168.0.28 (192.168.0.28)
⊞ User Datagram Protocol, Src Port: domain (53), Dst Port: 62872 (62872)
⊟ Domain Name System (response)
    [Request In: 381]
    [Time: 0.025771000 seconds]
    Transaction ID: 0xcf1f
  ⊞ Flags: 0x8180 (Standard query response, No error)
    Questions: 1
    Answer RRs: 6
    Authority RRs: 0
    Additional RRs: 0
  ⊟ Queries
    ⊟ www.cnn.com: type A, class IN
        Name: www.cnn.com
        Type: A (Host address)
        Class: IN (0x0001)
  ⊟ Answers
    ⊞ www.cnn.com: type A, class IN, addr 64.236.91.21
```

```
0000  00 1c 26 26 66 a2 00 0e  8e 04 d0 9e 08 00 45 00   ..&&f... ......E.
0010  00 99 00 00 40 00 40 11  b8 e6 c0 a8 00 01 c0 a8   ....@.@. ........
0020  00 1c 00 35 f5 98 00 85  98 5a cf 1f 81 80 00 01   ...5.... .Z......
0030  00 06 00 00 00 00 03 77  77 77 03 63 6e 6e 03 63   .......w ww.cnn.c
0040  6f 6d 00 00 01 00 01 c0  0c 00 01 00 01 00 00 00   om...... ........
0050  b7 00 04 40 ec 5b 15 c0  0c 00 01 00 01 00 00 00   ...@.[.. ........
0060  b7 00 04 40 ec 5b 17 c0  0c 00 01 00 01 00 00 00   ...@.[.. ........
0070  b7 00 04 40 ec 10 14 c0  0c 00 01 00 01 00 00 00   ...@.... ........
```

This is a response to the DNS query in this fr... Packets: 1273 Displayed: 909 Marked: 0 Dropped: 0 Profile: Default



```
SUBSCRIBE /upnp/service/Layer3Forwarding HTTP/1.1
NT: upnp:event
Callback: <http://192.168.0.2:5000/notify>
Timeout: Second-1800
User-Agent: Mozilla/4.0 (compatible; UPnP/1.0; Windows NT/5.1)
Host: 192.168.0.1
Content-Length: 0
Pragma: no-cache

HTTP/1.0 200 OK
Connection: close
Server: UPnP/1.0 UPnP-Device-Host/1.0
Timeout: Second-1800
SID: uuid:cf
```

Follow the stream to see content in packet

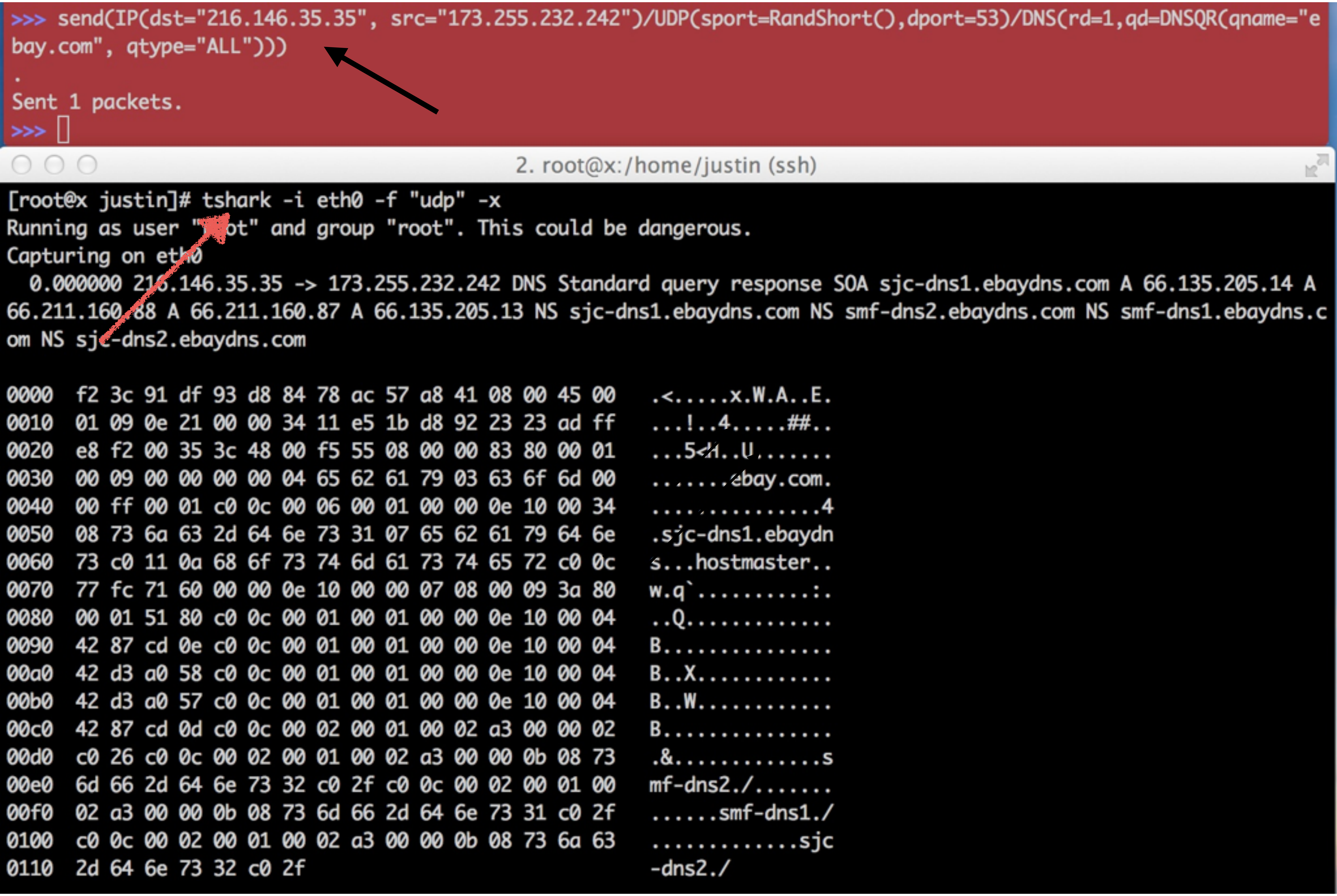
Wireshark Demo

tshark

tshark -h

```
Usage: tshark [options] ...

Capture interface:
  -i <interface>           name or idx of interface (def: first non-loopback)
  -f <capture filter>      packet filter in libpcap filter syntax
  -s <snaplen>             packet snapshot length (def: 65535)
  -I                       capture in monitor mode, if available


Input file:
  -r <infile>              set the filename to read from (- to read from stdin)


Output:

  -T pdml|ps|psml|text|fields
                           format of text output (def: text)
  -e <field>               field to print if -Tfields selected (e.g. tcp.port,
                           _ws.col.Info)
                           this option can be repeated to print multiple fields

     aggregator=,|/s|<char> select comma, space, printable character as
                           aggregator
     quote=d|s|n           select double, single, no quotes for values
  -t a|ad|d|dd|e|r|u|ud    output format of time stamps (def: r: rel. to first)
  -u s|hms                 output format of seconds (def: s: seconds)
  -l                       flush standard output after each packet
  -q                       be more quiet on stdout (e.g. when using statistics)
  -z <statistics>          various statistics, see the man page for details
```

```
tshark -r <your.pcap>
-T fields -e tcp.stream
```



```
>>> send(IP(dst="216.146.35.35", src="173.255.232.242")/UDP(sport=RandShort(),dport=53)/DNS(rd=1,qd=DNSQR(qname="e
bay.com", qtype="ALL")))
.
Sent 1 packets.
>>> 
```

2. root@x:/home/justin (ssh)

```
[root@x justin]# tshark -i eth0 -f "udp" -x
Running as user "root" and group "root". This could be dangerous.
Capturing on eth0
  0.000000 216.146.35.35 -> 173.255.232.242 DNS Standard query response SOA sjc-dns1.ebaydns.com A 66.135.205.14 A
66.211.160.88 A 66.211.160.87 A 66.135.205.13 NS sjc-dns1.ebaydns.com NS smf-dns2.ebaydns.com NS smf-dns1.ebaydns.c
om NS sjc-dns2.ebaydns.com

0000  f2 3c 91 df 93 d8 84 78 ac 57 a8 41 08 00 45 00   .<.....x.W.A..E.
0010  01 09 0e 21 00 00 34 11 e5 1b d8 92 23 23 ad ff   ...!..4.....##..
0020  e8 f2 00 35 3c 48 00 f5 55 08 00 00 83 80 00 01   ...5<H..U.......
0030  00 09 00 00 00 00 04 65 62 61 79 03 63 6f 6d 00   .......ebay.com.
0040  00 ff 00 01 c0 0c 00 06 00 01 00 00 0e 10 00 34   ...............4
0050  08 73 6a 63 2d 64 6e 73 31 07 65 62 61 79 64 6e   .sjc-dns1.ebaydn
0060  73 c0 11 0a 68 6f 73 74 6d 61 73 74 65 72 c0 0c   s...hostmaster..
0070  77 fc 71 60 00 00 0e 10 00 00 07 08 00 09 3a 80   w.q`..........:.
0080  00 01 51 80 c0 0c 00 01 00 01 00 00 0e 10 00 04   ..Q.............
0090  42 87 cd 0e c0 0c 00 01 00 01 00 00 0e 10 00 04   B...............
00a0  42 d3 a0 58 c0 0c 00 01 00 01 00 00 0e 10 00 04   B..X............
00b0  42 d3 a0 57 c0 0c 00 01 00 01 00 00 0e 10 00 04   B..W............
00c0  42 87 cd 0d c0 0c 00 02 00 01 00 02 a3 00 00 02   B...............
00d0  c0 26 c0 0c 00 02 00 01 00 02 a3 00 00 0b 08 73   .&.............s
00e0  6d 66 2d 64 6e 73 32 c0 2f c0 0c 00 02 00 01 00   mf-dns2./.......
00f0  02 a3 00 00 0b 08 73 6d 66 2d 64 6e 73 31 c0 2f   ......smf-dns1./
0100  c0 0c 00 02 00 01 00 02 a3 00 00 0b 08 73 6a 63   .............sjc
0110  2d 64 6e 73 32 c0 2f                              -dns2./
```

How do we bring this response back to python?

import ???

```
import subprocess
```

subprocess module

- The subprocess module allows you to spawn new processes, connect to their input/output/error pipes, and obtain their return codes. This module intends to replace several older modules and functions

source: https://docs.python.org/2/library/subprocess.html

subprocess.Popen

- Popen(['/bin/sh', '-c', args[0], args[1], ...])

tshark and subprocess

- pcap = ‘your_file.pcap’
- collect_streams = subprocess.Popen(["tshark", "-r", pcap, "-T", "fields", "-e", "tcp.stream"], stdout=subprocess.PIPE)
- streams = collect_streams.stdout.read().splitlines()

print(streams)

```
b'1', b'1', b'1', b'1', b'1', b'1', b'1', b'1', b'1', b'1', b'1', b'1', b'1',
b'1', b'2', b'2', b'2', b'2', b'2', b'2', b'2', b'2', b'2', b'2', b'2', b'2',
b'2', b'2', b'3', b'3', b'3', b'3', b'4', b'3', b'4', b'4', b'4', b'4', b'5',
b'3', b'3', b'5', b'5', b'5', b'5', b'6', b'6', b'6', b'6', b'7', b'6', b'7',
b'7', b'7', b'7', b'3', b'3', b'8', b'8', b'8', b'8', b'8', b'3', b'3', b'3',
b'9', b'9', b'9', b'9', b'9', b'3', b'3', b'4', b'4', b'4', b'4', b'4', b'4',
b'4', b'10', b'10', b'10', b'10', b'10', b'4', b'4', b'5', b'5', b'5',
b'11', b'11', b'11', b'11', b'11', b'5', b'5', b'6', b'6', b'6', b'6',
b'6', b'7', b'7', b'7', b'7', b8'…
```

Consolidate all streams (we only want unique)

```
unique_streams = sorted(set([str(s, 'UTF8') for s in streams if len(s) > 0]))
```

Retrieve Stream Content

```
stream_follower = subprocess.Popen(["tshark", "-r", pcap, "-qz",
"follow,tcp,ascii,"+stream], stdout=subprocess.PIPE)
```

Generates a text file with stream number and content

```
content = stream_follower.stdout.read()
```

Consolidate all streams (we only want unique)

```
unique_streams = sorted(set([str(s, 'UTF8') for s in streams if len(s) > 0]))
```

Streams Demo

Questions?

Exercise

Open up the sharkDissector skeleton to get started