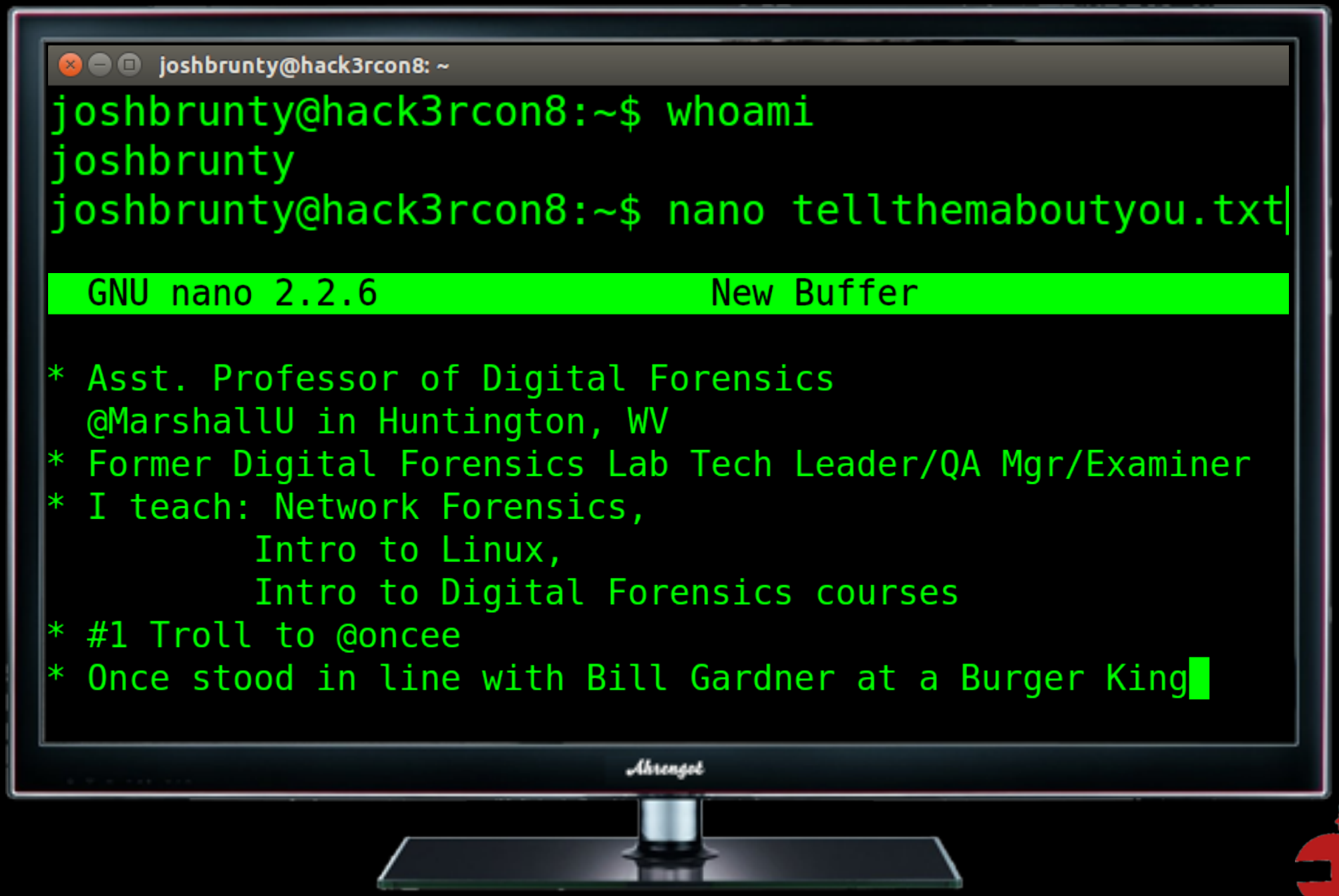
Introduction to Packet Capture & Analysis using TCPDUMP & WIRESHARK

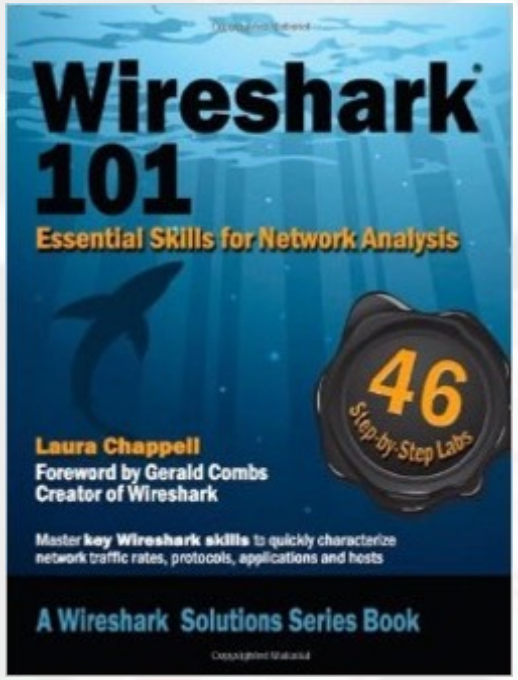
Josh Brunty
Assistant Professor
Marshall University

MARSHALL
DIGITAL FORENSICS
INFORMATION ASSURANCE

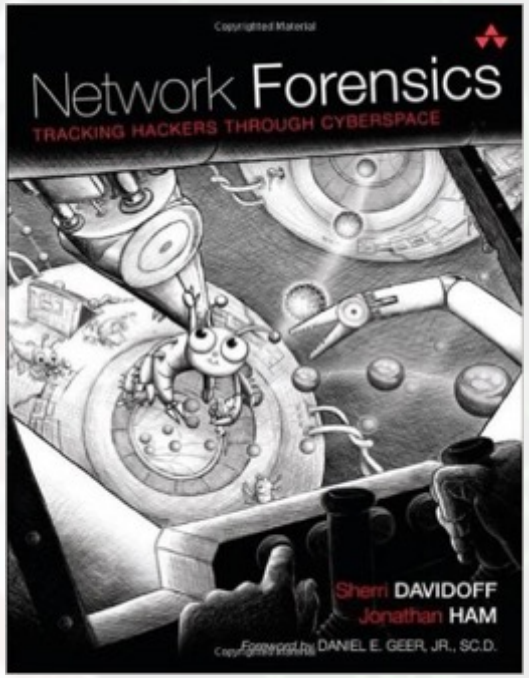
Hack3rCon 8 THE OCHO


joshbrunty@hack3rcon8: ~
joshbrunty@hack3rcon8:~$ whoami
joshbrunty
joshbrunty@hack3rcon8:~$ nano tellthemaboutyou.txt
GNU nano 2.2.6 New Buffer
* Asst. Professor of Digital Forensics
@MarshallU in Huntington, WV
* Former Digital Forensics Lab Tech Leader/QA Mgr/Examiner
* I teach: Network Forensics,
Intro to Linux,
Intro to Digital Forensics courses
* #1 Troll to @oncee
* Once stood in line with Bill Gardner at a Burger King

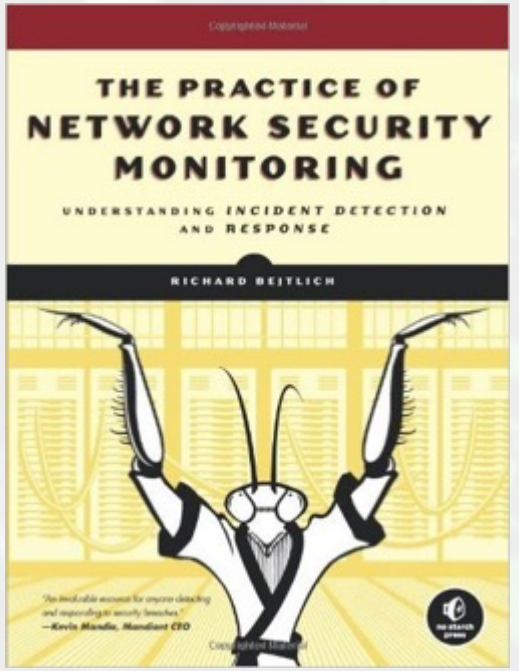
Resources I'd Recommend…



http://a.co/0Zg87AK



http://a.co/dtRtKuA



http://a.co/agGV6Ht

http://a.co/iavdsQw

Hack3rCon 8 THE OCHO

What we will need for this session…

- Kali and/or SIFT Workstation
  - http://www.kali.org
  - https://digital-forensics.sans.org/community/downloads
- Bash terminal
- A stable internet browser
- Download course materials PowerPoint, PCAP Files
  - www.github.com/joshbrunty/Hack3rCon8_Wireshark
- Wireshark
  - www.wireshark.org
  - Version 2 (preferably)
  - Running in Linux (preferably SANS SIFT or Kali)
  - Linux Users: `apt-get install wireshark-qt`

What do we use Wireshark for?

- Wireshark IS used for decoding and analyzing captured traffic
- Wireshark IS used to extract contents of packets/packet files (pcaps)

What do we NOT use Wireshark for?

- Wireshark IS NOT made to capture large amounts of traffic
- Wireshark IS NOT a SIEM or Firewall
- Wireshark IS NOT a real-time network monitoring tool (that's what IDS's are for)

Tools Overview

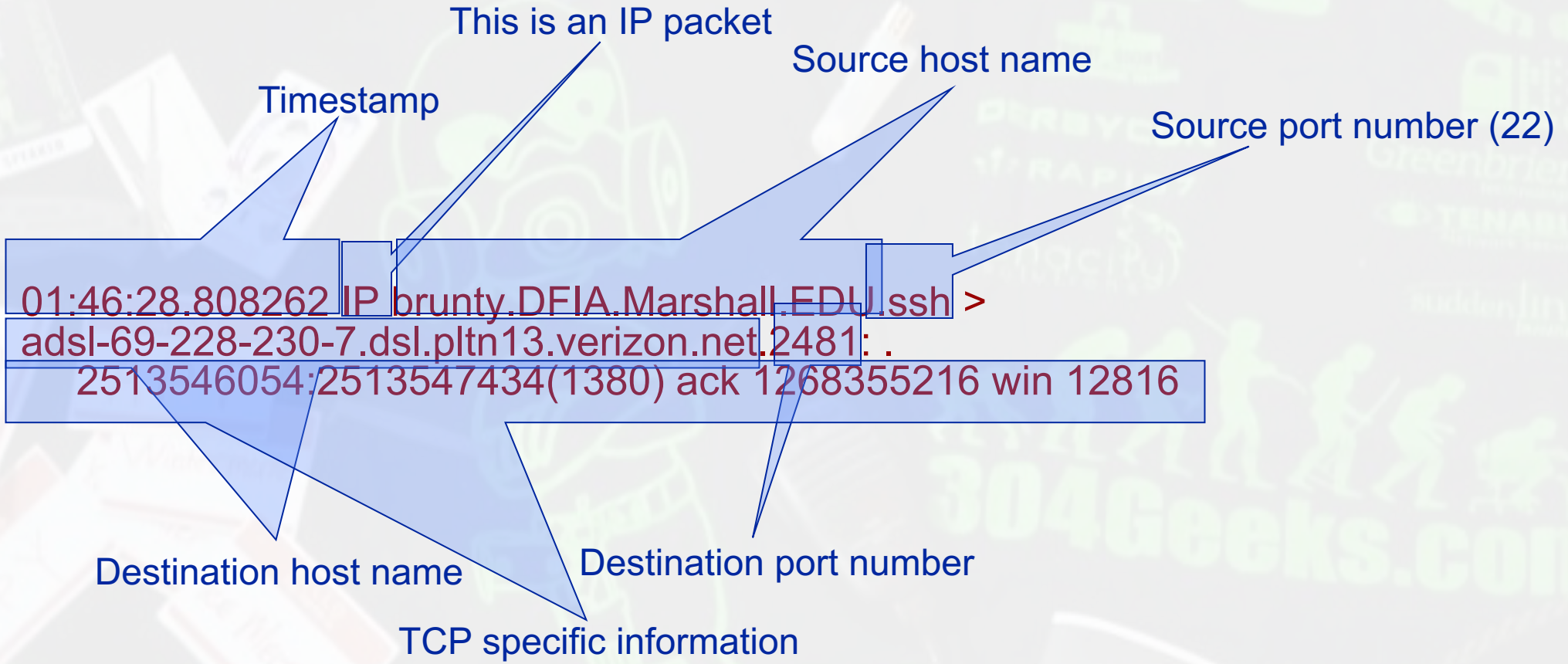
- Tcpdump
  - Unix-based command-line tool used to intercept packets
    - Including filtering to just the packets of interest
  - Reads “live traffic” from interface specified using **`-i`** option …
  - … or from a previously recorded *trace file* specified using **`-r`** option
    - You create files when capturing live traffic using **`-w`** option
- Tshark
  - Tcpdump-like capture program that comes w/ Wireshark
  - Very similar behavior & flags to tcpdump
- Wireshark
  - GUI for displaying tcpdump/tshark packet traces

Tcpdump example

- Example run of tcpdump on a Linux machine
- First few lines of the output:

```
01:46:28.808262 IP brunty.DFIA.Marshall.EDU.ssh > adsl-69-228-230-
  7.dsl.pltn13.verizon.net.2481: . 2513546054:2513547434(1380) ack
  1268355216 win 12816
01:46:28.808271 IP brunty.DFIA.Marshall.EDU.ssh > adsl-69-228-230-
  7.dsl.pltn13.verizon.net.2481: P 1380:2128(748) ack 1 win 12816
01:46:28.808276 IP brunty.DFIA.Marshall.EDU.ssh > adsl-69-228-230-
  7.dsl.pltn13.verizon.net.2481: . 2128:3508(1380) ack 1 win 12816
01:46:28.890021 IP adsl-69-228-230-7.dsl.pltn13.verizon.net.2481 >
  brunty.DFIA.Marshall.EDU.ssh: P 1:49(48) ack 1380 win 16560
```

What does a packet convey?



- Different output formats for different packet types

Similar Output from Tshark

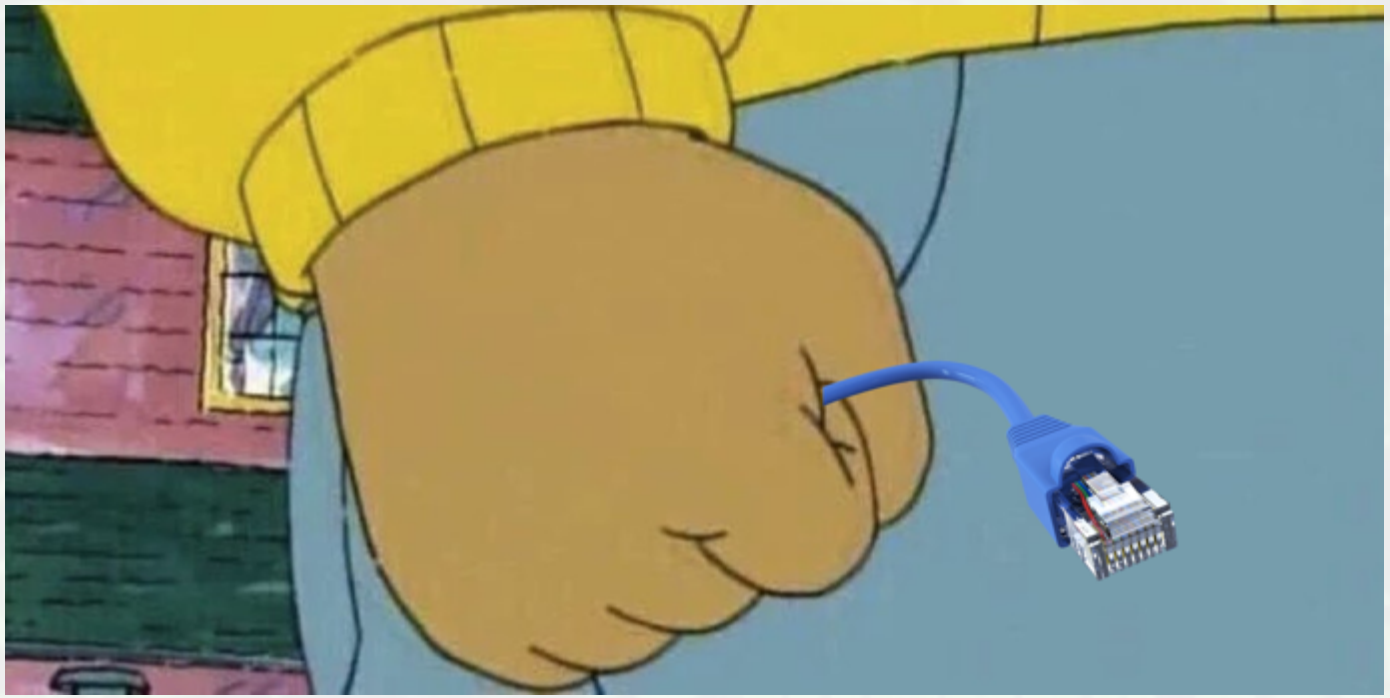
```
1190003744.940437 61.184.241.230 -> 128.32.48.169
   SSH Encrypted request packet len=48
1190003744.940916 128.32.48.169 -> 61.184.241.230
   SSH Encrypted response packet len=48
1190003744.955764 61.184.241.230 -> 128.32.48.169
   TCP 6943 > ssh [ACK] Seq=48 Ack=48 Win=65514 Len=0
   TSV=445871583 TSER=632535493
1190003745.035678 61.184.241.230 -> 128.32.48.169
   SSH Encrypted request packet len=48
1190003745.036004 128.32.48.169 -> 61.184.241.230
   SSH Encrypted response packet len=48
1190003745.050970 61.184.241.230 -> 128.32.48.169
   TCP 6943 > ssh [ACK] Seq=96 Ack=96 Win=65514 Len=0
   TSV=445871583 TSER=632535502
```

Running tcpdump

- Requires superuser/administrator privileges on Unix
  - http://www.tcpdump.org/
  - You can do it on your own Unix machine
  - You can install a Linux OS (SIFT/Kali) in Vmware on your machine
- Tcpdump for Windows
  - WinDump: http://www.winpcap.org/windump/
    - Free software
    - Part of Wireshark for Windows install

First Things First….

- We need to connect to the network we're sniffing
- Method 1: "on the wire"
- Method 2: "in the air"



http://www.newegg.com/Product/Product.aspx?Item=9SIA8UD37Y8211&cm_re=Alfa_wireless_adapter-_-9SIA8UD37Y8211-_-Product

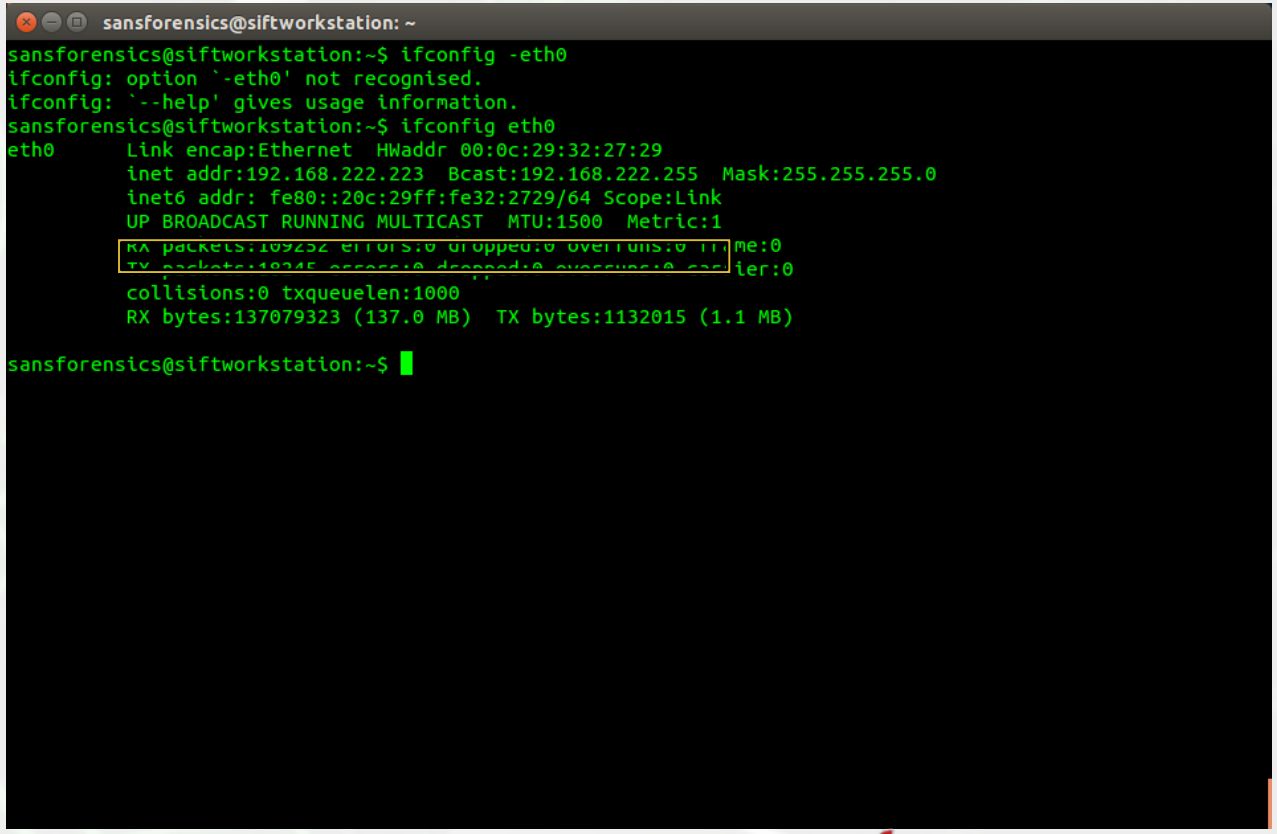
https://www.hak5.org/gear/packet-squirrel

http://hackerwarehouse.com/product/lan-tap-pro/

Hack3rCon 8 THE OCHO

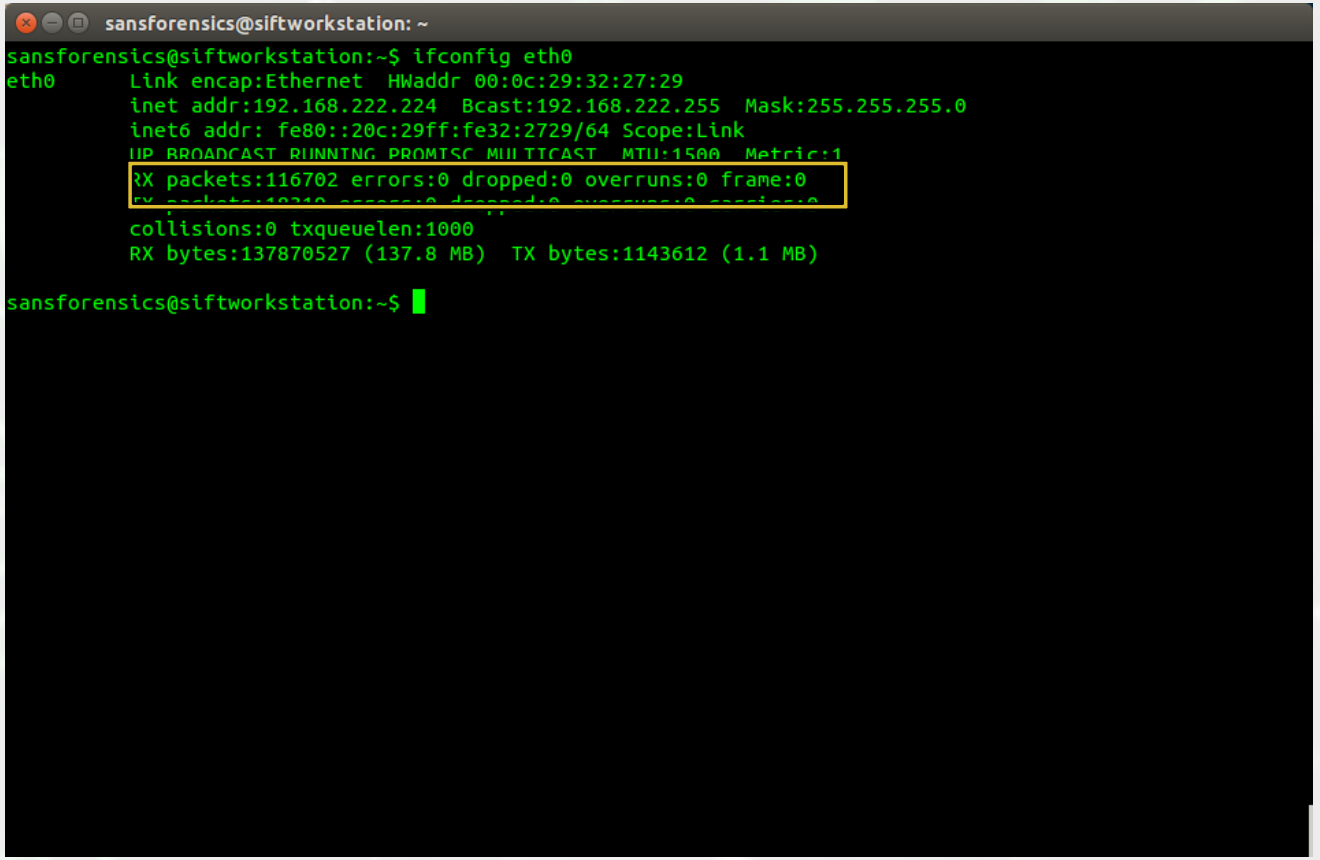
First Things First….

- In order to capture packets we need to check and see if our capture interface is in promiscuous mode:
  - `$ sudo ifconfig eth0`



First Things First….

- If it is not, then we can set it to promiscuous mode
  - `$ sudo ifconfig eth0 promisc`



Demo 1 – Our 1st Run

- Syntax: *tcpdump [options] [filter expression]*
- I will demonstrate it using our domain here
  - `$ sudo tcpdump -i eth0`
  - On your own Unix machine, you can run it using “sudo” or directly run “tcpdump”
- Observe the output

Filters

- We are often not interested in all packets flowing through the network
- Use filters to capture only packets of interest to us

Demo 2- Setting a Capture Filter

1. Capture only udp packets
   - `tcpdump "udp"`
2. Capture only tcp packets
   - `tcpdump "tcp"`

Demo 2 (contd.)

1. Capture only UDP packets with destination port 53 (DNS requests)
   - `tcpdump "udp dst port 53"`
2. Capture only UDP packets with source port 53 (DNS replies)
   - `tcpdump "udp src port 53"`
3. Capture only UDP packets with source or destination port 53 (DNS requests and replies)
   - `tcpdump "udp port 53"`

Demo 2 (contd.)

1. Capture only packets destined to forensics.marshall.edu
   - `tcpdump "dst host forensics.marshall.edu"`
2. Capture both DNS packets and TCP packets to/from forensics.marshall.edu
   - `tcpdump "(tcp and host forensics.marshall.edu) or udp port 53"`

How to write filters

- Refer the tcpdump/tshark man page
  - `man pcap-filter` (tcpdump)
  - `Man wireshark-filter` (Tshark/Wireshark)
- Many example webpages on the Internet

So What is WireShark?

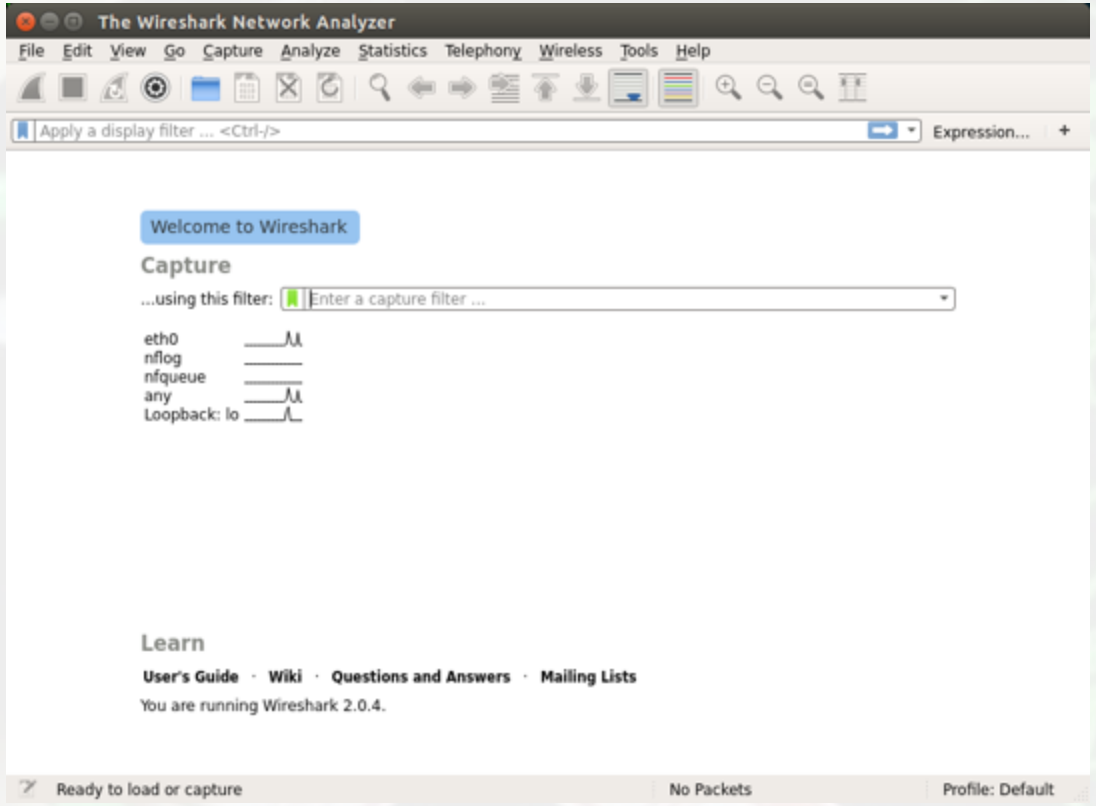
- Packet sniffer/protocol analyzer
- Open Source Network Tool
- Latest version of the ethereal tool

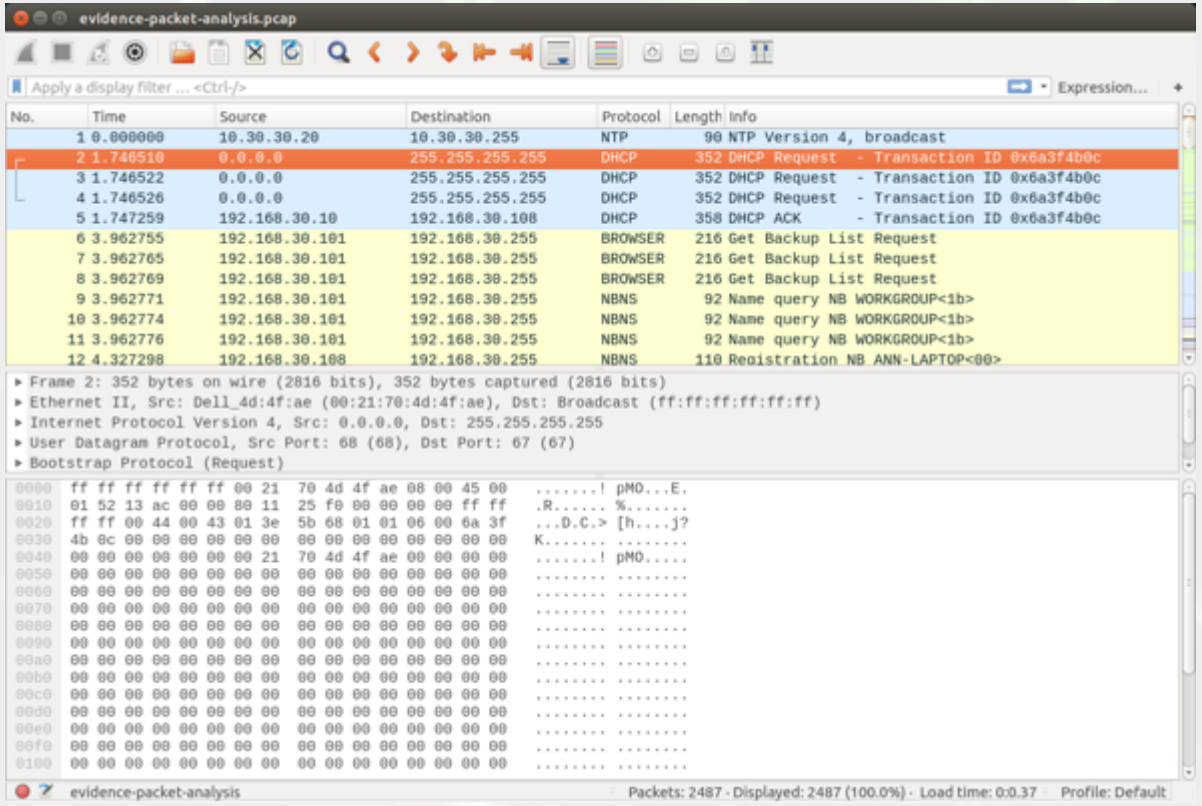
What is tShark?

- The command-line based packet capture tool
- Equivalent to Wireshark except command line (CLI) based

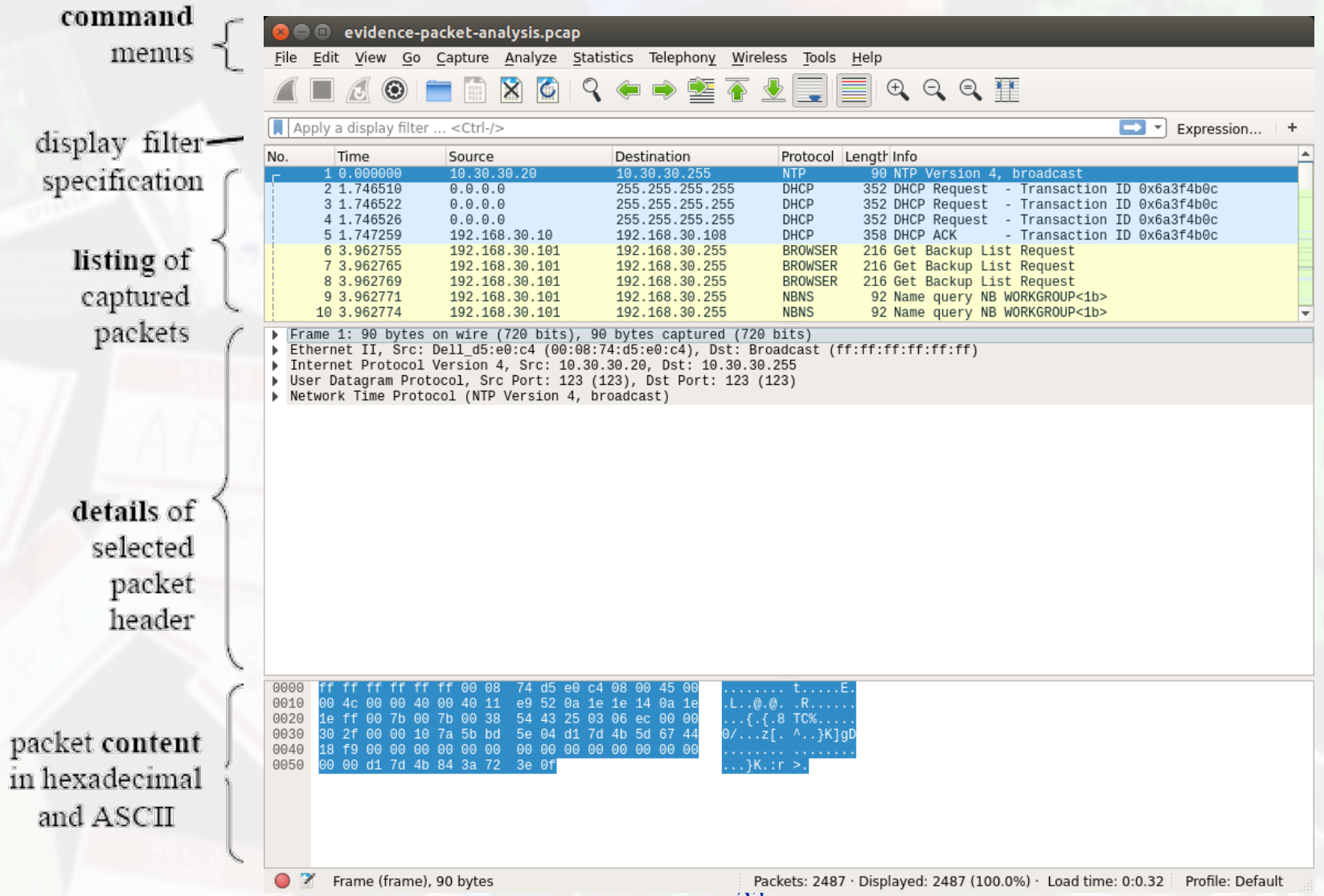
Hack3rCon 8 THE OCHO

Wireshark Interface

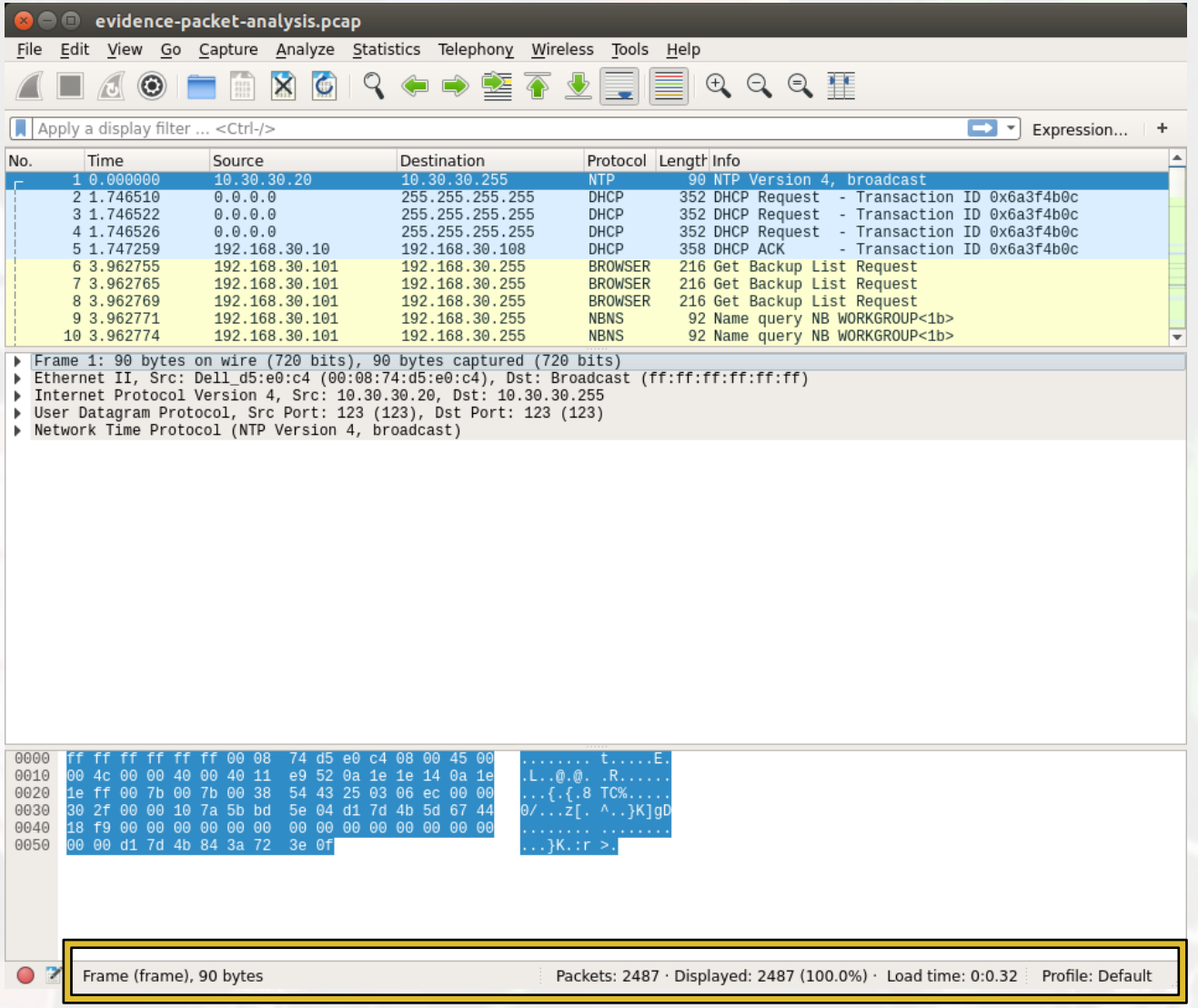




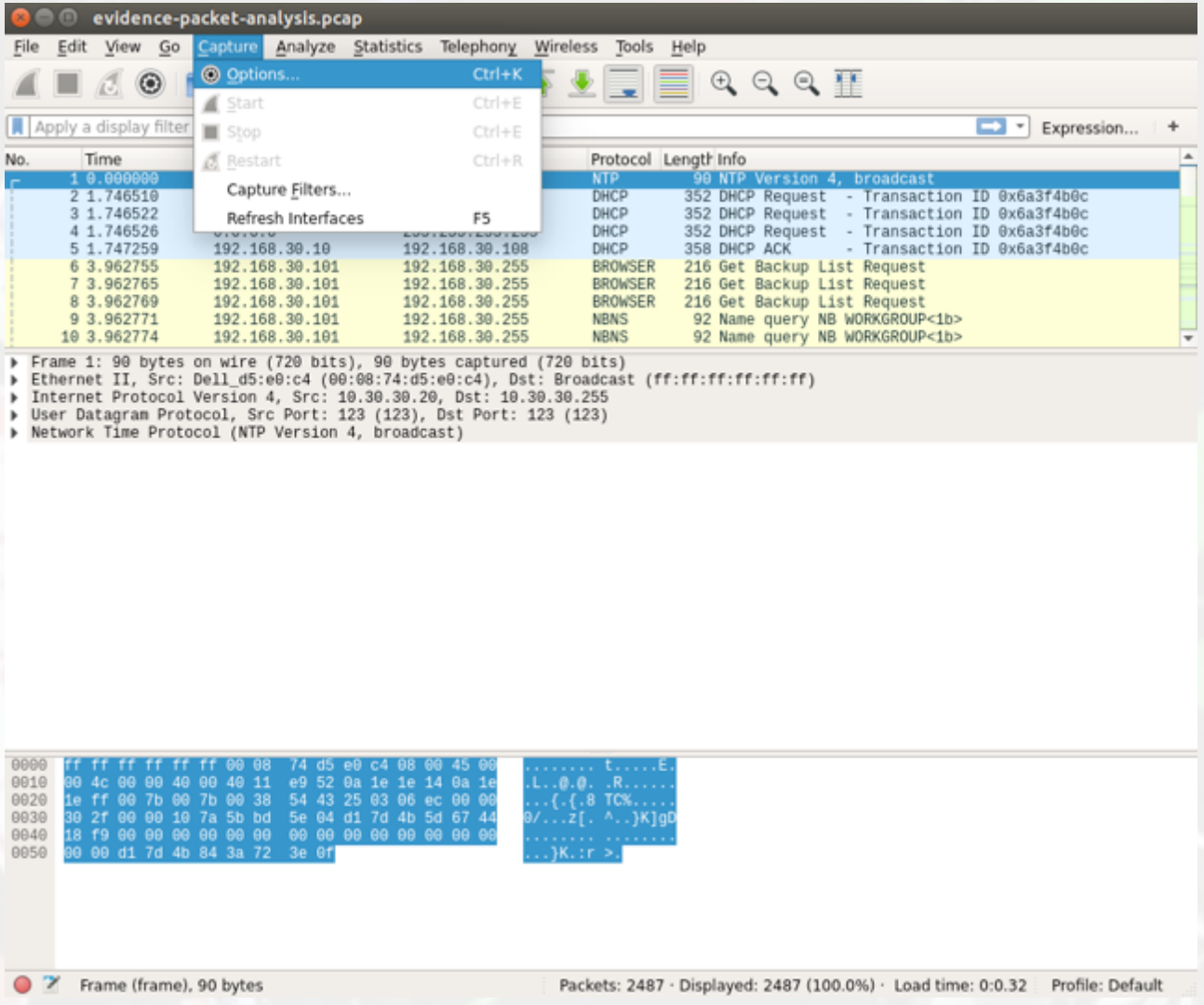
Wireshark Interface



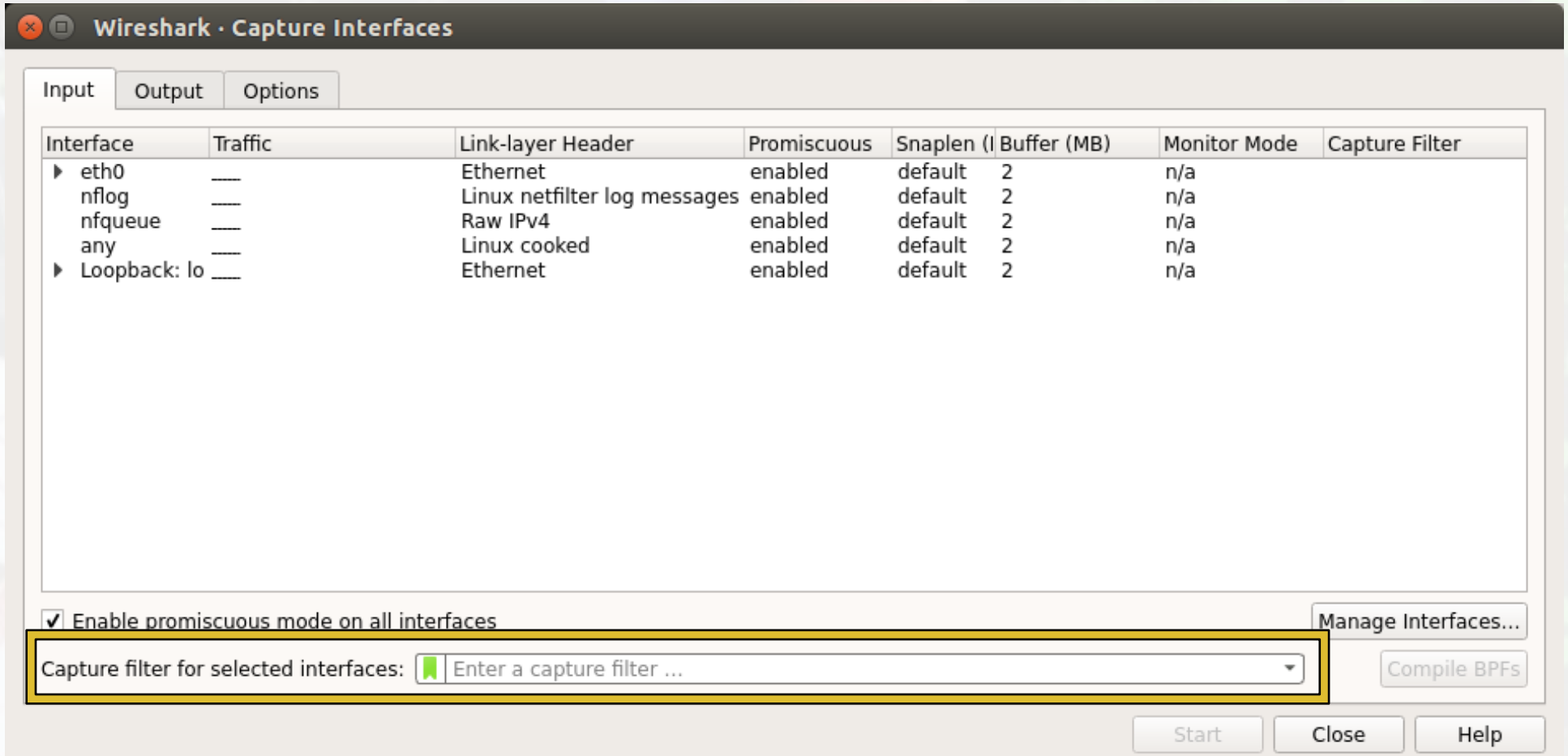
Status Bar


evidence-packet-analysis.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
No. Time Source Destination Protocol Length Info
1 0.000000 10.30.30.20 10.30.30.255 NTP 90 NTP Version 4, broadcast
2 1.746510 0.0.0.0 255.255.255.255 DHCP 352 DHCP Request - Transaction ID 0x6a3f4b0c
3 1.746522 0.0.0.0 255.255.255.255 DHCP 352 DHCP Request - Transaction ID 0x6a3f4b0c
4 1.746526 0.0.0.0 255.255.255.255 DHCP 352 DHCP Request - Transaction ID 0x6a3f4b0c
5 1.747259 192.168.30.10 192.168.30.108 DHCP 358 DHCP ACK - Transaction ID 0x6a3f4b0c
6 3.962755 192.168.30.101 192.168.30.255 BROWSER 216 Get Backup List Request
7 3.962765 192.168.30.101 192.168.30.255 BROWSER 216 Get Backup List Request
8 3.962769 192.168.30.101 192.168.30.255 BROWSER 216 Get Backup List Request
9 3.962771 192.168.30.101 192.168.30.255 NBNS 92 Name query NB WORKGROUP<1b>
10 3.962774 192.168.30.101 192.168.30.255 NBNS 92 Name query NB WORKGROUP<1b>
Frame 1: 90 bytes on wire (720 bits), 90 bytes captured (720 bits)
Ethernet II, Src: Dell_d5:e0:c4 (00:08:74:d5:e0:c4), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 10.30.30.20, Dst: 10.30.30.255
User Datagram Protocol, Src Port: 123 (123), Dst Port: 123 (123)
Network Time Protocol (NTP Version 4, broadcast)
0000 ff ff ff ff ff ff 00 08 74 d5 e0 c4 08 00 45 00 .... t.....E.
0010 00 4c 00 00 40 00 40 11 e9 52 0a 1e 1e 14 0a 1e .L..@.@. .R......
0020 1e ff 00 7b 00 7b 00 38 54 43 25 03 06 ec 00 00 ...{.{.8 TC%.....
0030 30 2f 00 00 10 7a 5b bd 5e 04 d1 7d 4b 5d 67 44 0/...z[. ^..}K]gD
0040 18 f9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
0050 00 00 d1 7d 4b 84 3a 72 3e 0f ...}K.:r >.
Frame (frame), 90 bytes
Packets: 2487 · Displayed: 2487 (100.0%) · Load time: 0:0.32
Profile: Default

Capture Options



Capture Filter

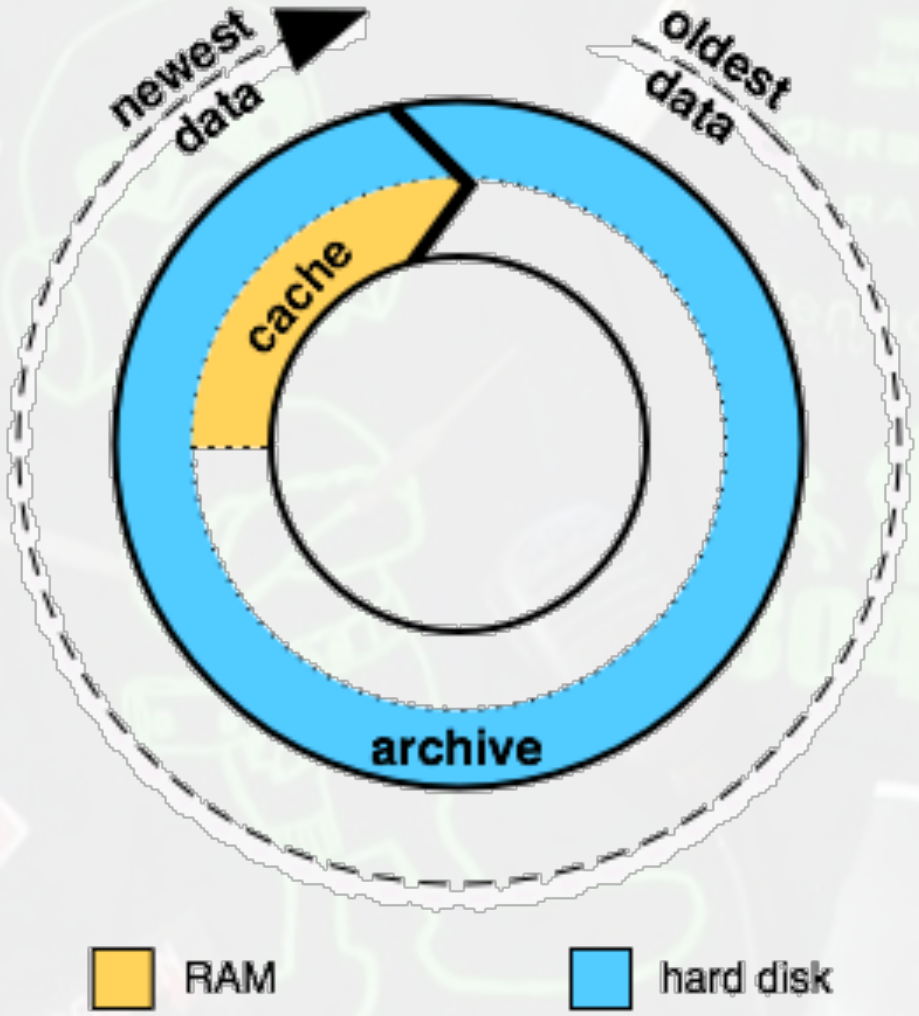


Capture Filter examples

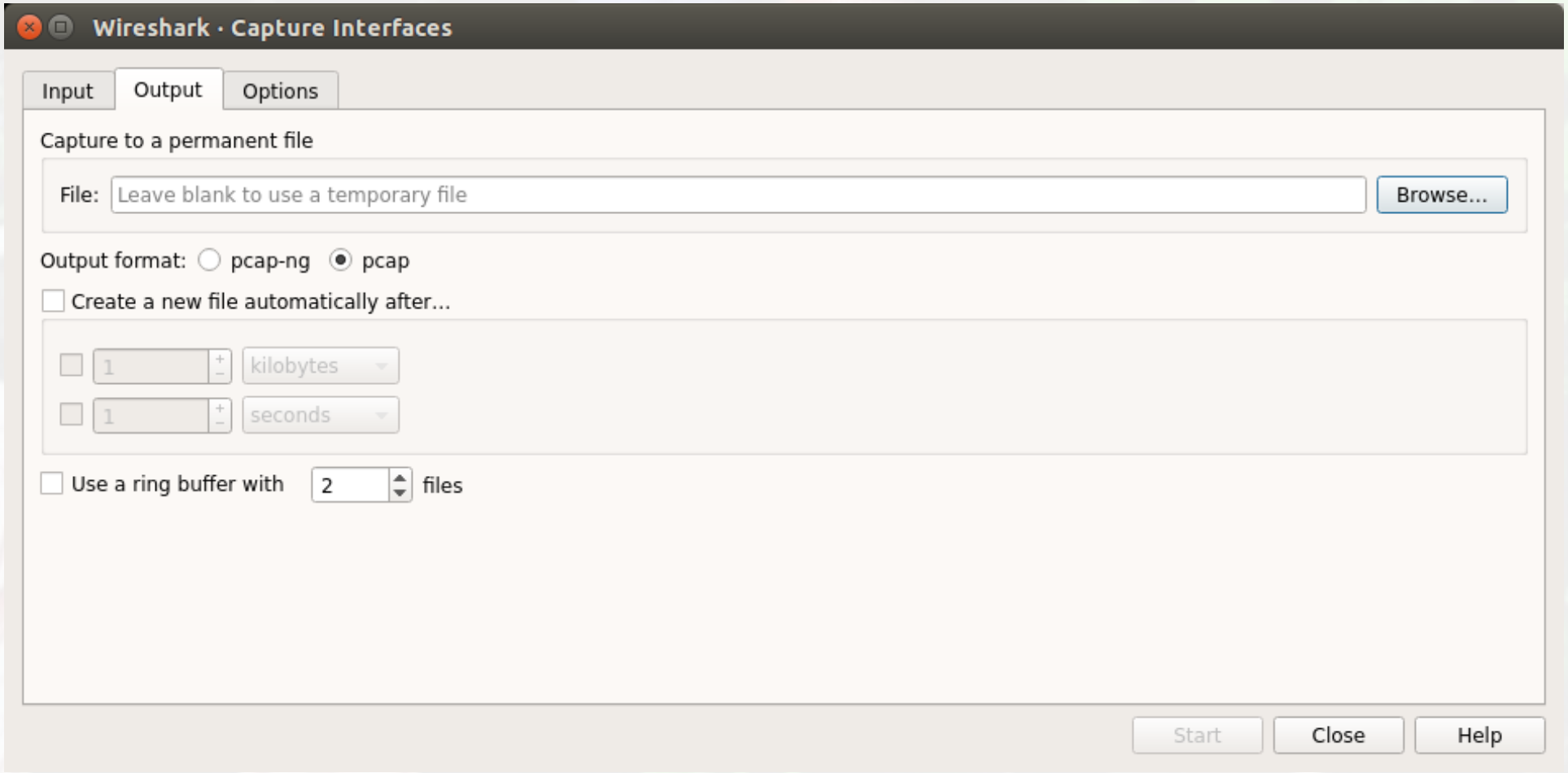
```
host 10.1.11.24

host securewv.com

host 192.168.0.1 and host 10.1.11.1

tcp port http

ip

not broadcast not multicast

ether host 00:04:13:00:09:a3
```

http://packetlife.net/media/library/13/Wireshark_Display_Filters.pdf

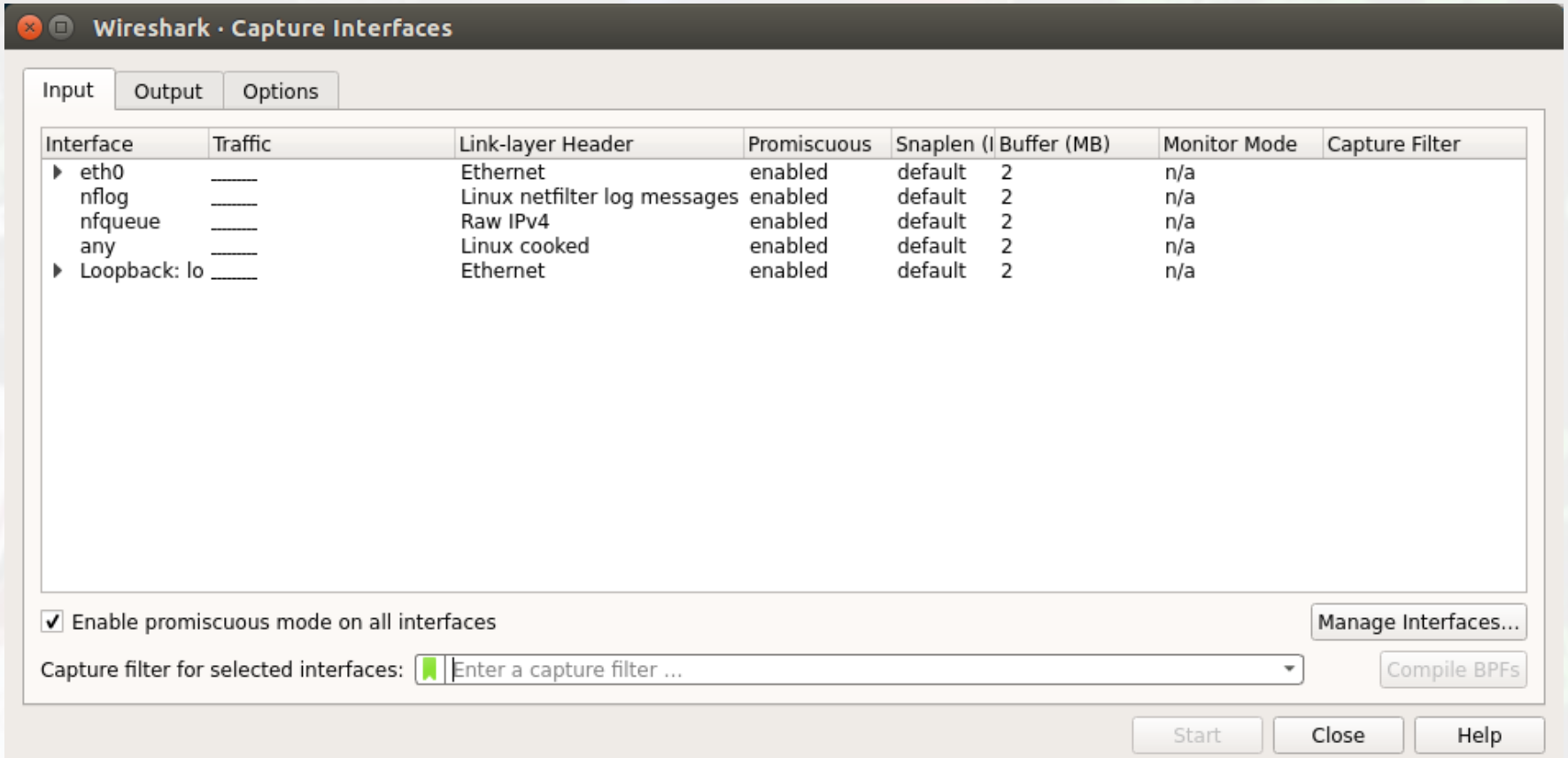
Capture Buffer



Capture Interfaces (Capture Buffer)



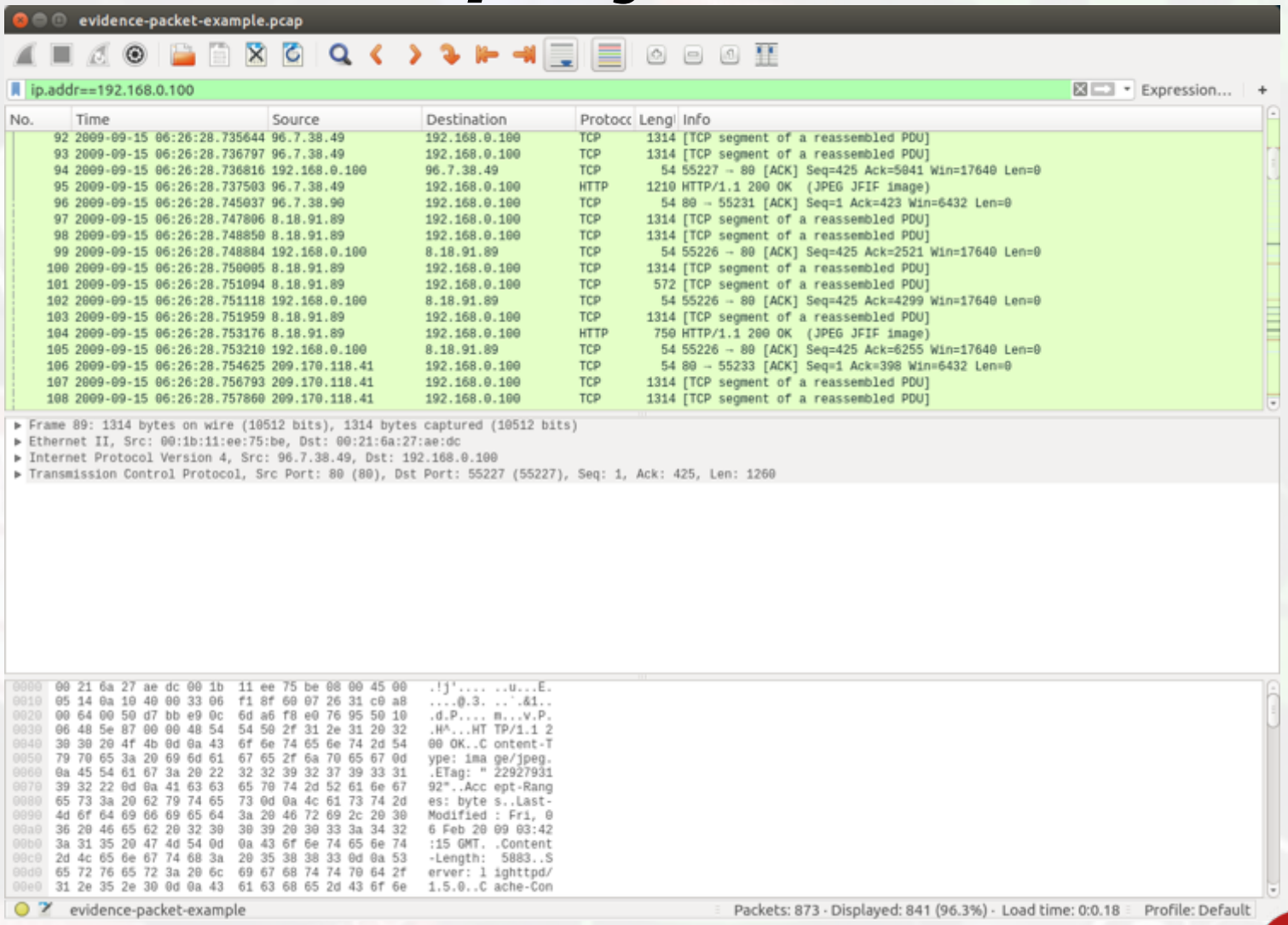
Capture Interfaces



Display Filters (Post-Filters)

- Display filters (also called post or post-capture filters) only filter the view of what you are seeing. All packets in the capture still exist in the trace
- Display filters use their own format and are much more powerful then capture filters

Display Filter

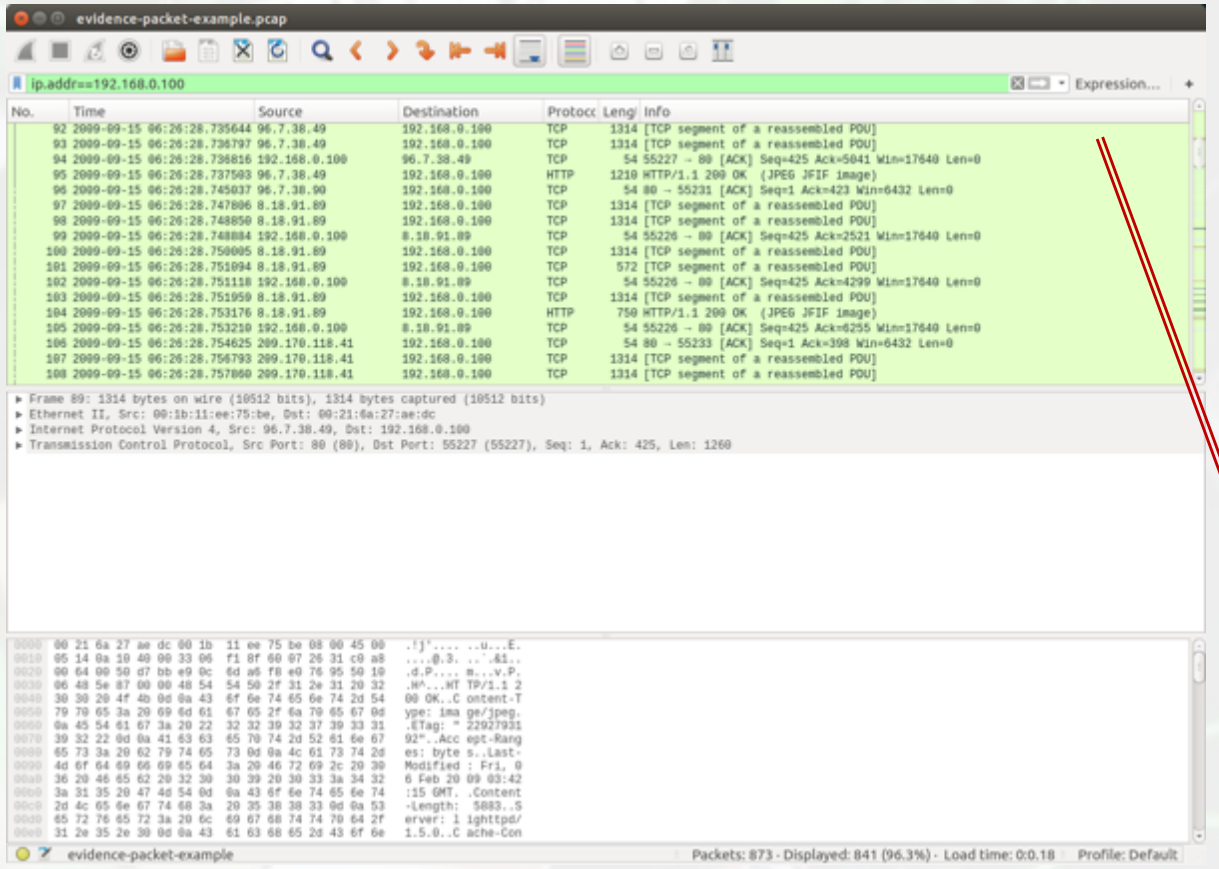


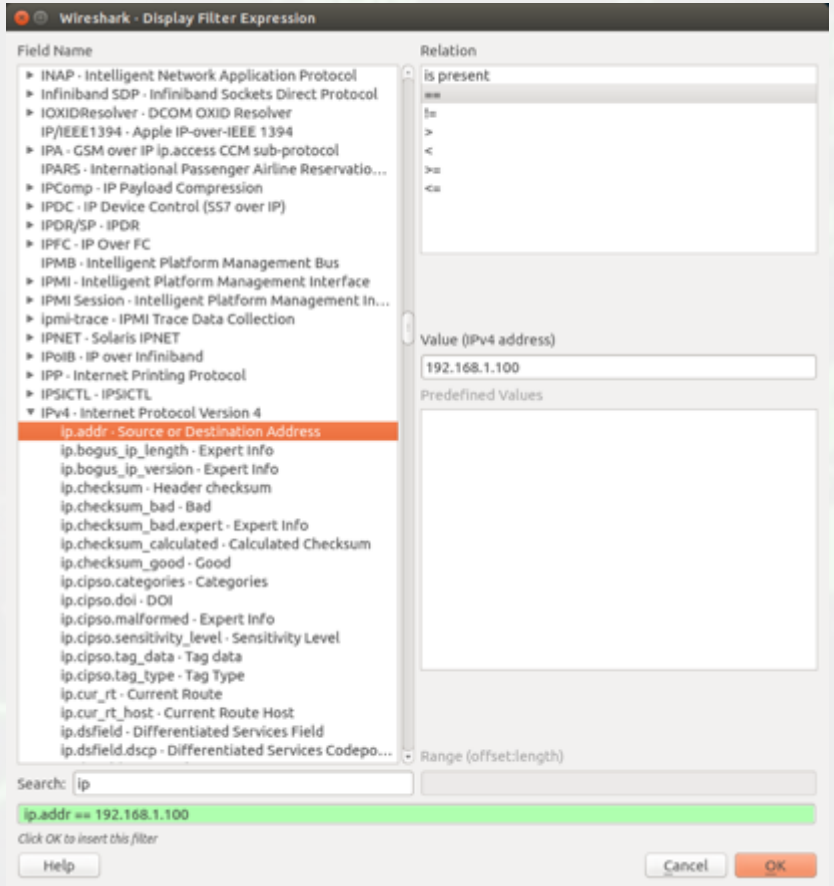
Display Filter Examples

```
ip.src==10.1.11.00/24

ip.dst==10.1.11.00/24

ip.addr==192.168.1.10 && ip.addr==192.168.1.20

tcp.port==80 || tcp.port==3389

!(ip.addr==192.168.1.10 && ip.addr==192.168.1.20)

(ip.addr==192.168.1.10 && ip.addr==192.168.1.20) &&
   (tcp.port==445 || tcp.port==139)

(ip.addr==192.168.1.10 && ip.addr==192.168.1.20) &&
   (udp.port==67 || udp.port==68)

tcp.dstport == 80

Frame contains searchtermyouaresearchingfor   e.g. brunty
```

Display Filter

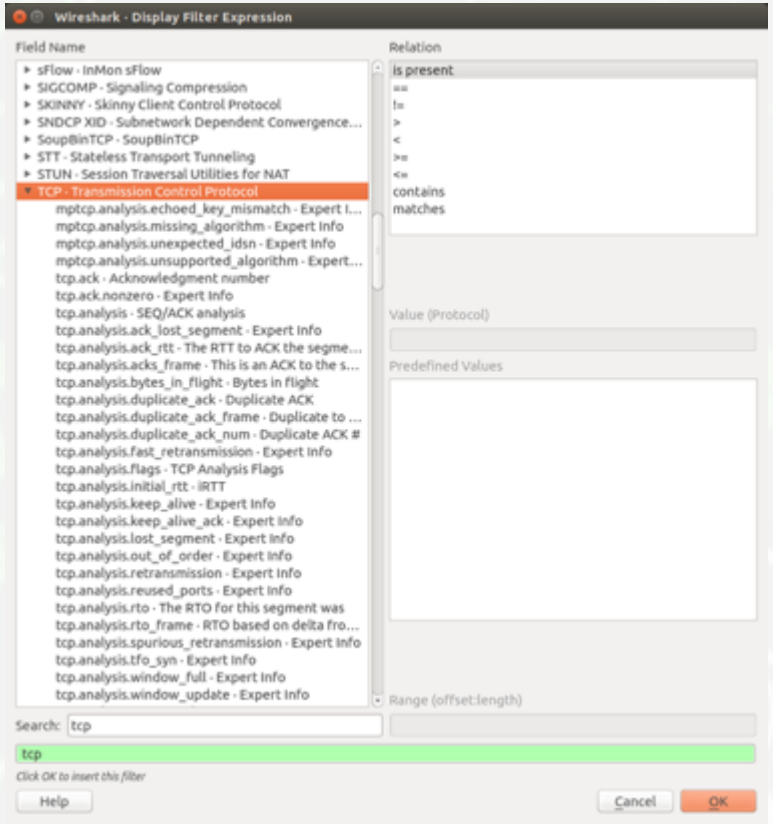
Syntax:	Protocol	.	String 1	.	String 2	Comparison operator	Value	Logical Operations	Other expression
Example:	ftp		passive		ip	==	10.2.3.4	xor	icmp.type





Display Filter

- String1, String2 (Optional settings):
  - Sub protocol categories inside the protocol.
  - Look for a protocol and then click on the "+" character.
  - Example:
  - **tcp.srcport == 80**
  - **tcp.flags == 2**
    - **SYN packet**
    - **Tcp.flags.syn==1**
  - **tcp.flags == 18**
    - **SYN/ACK**



Display Filter Expressions

- snmp || dns || icmp
  - Display the SNMP or DNS or ICMP traffics.
- tcp.port == 80
  - Display packets with TCP source or destination port 80.
- tcp.flags
  - Display packets having a TCP flags
- tcp.flags.syn == 0x02
  - Display packets with a TCP SYN flag.

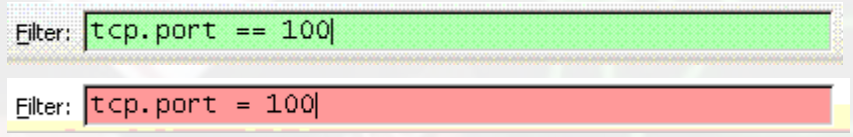
Six comparison operators are available:

English format:	C like format:	Meaning:
eq	==	Equal
ne	!=	Not equal
gt	>	Greater than
lt	<	Less than
ge	>=	Greater or equal
le	<=	Less or equal

Logical expressions:

English format:	C like format:	Meaning:
and	&&	Logical AND
or	||	Logical OR
xor	^^	Logical XOR
not	!	Logical NOT

If the filter syntax is correct, it will be highlighted in green, otherwise if there is a syntax mistake it will be highlighted in red.



Correct syntax

Wrong syntax

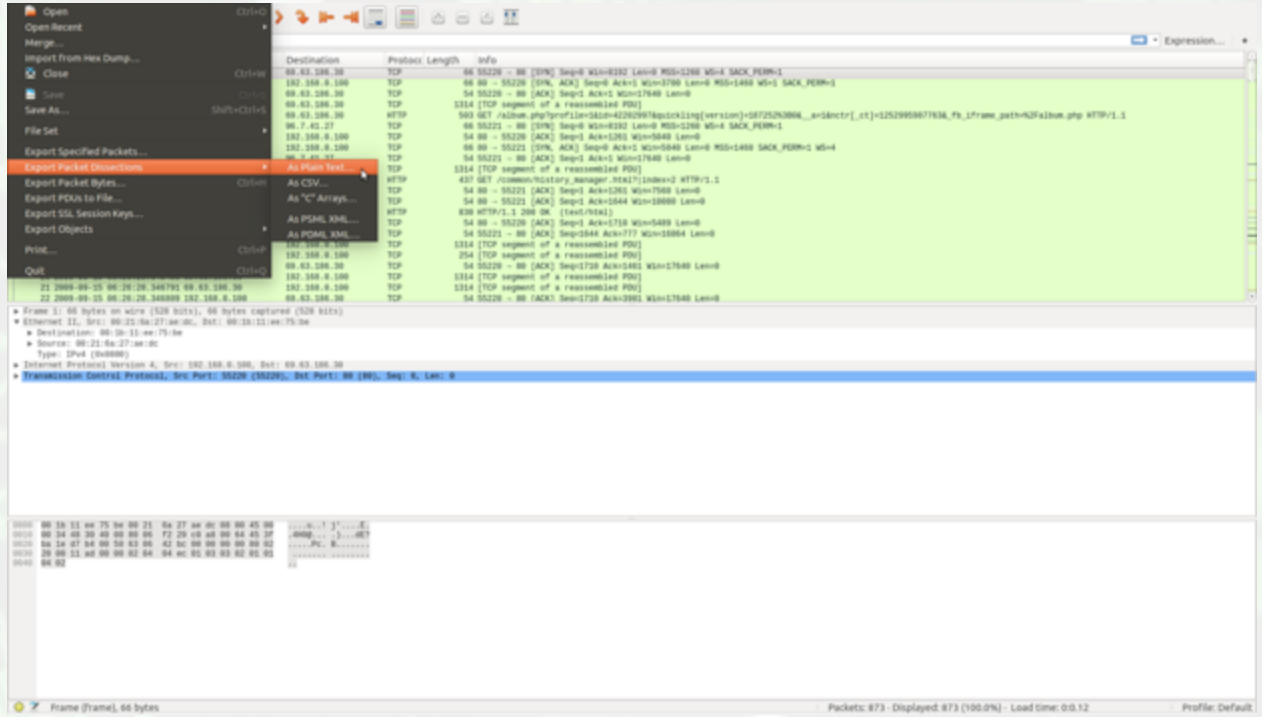
Save Filtered Packets After Using Display Filter

- We can also save all filtered packets in text file for further analysis
- Operation:

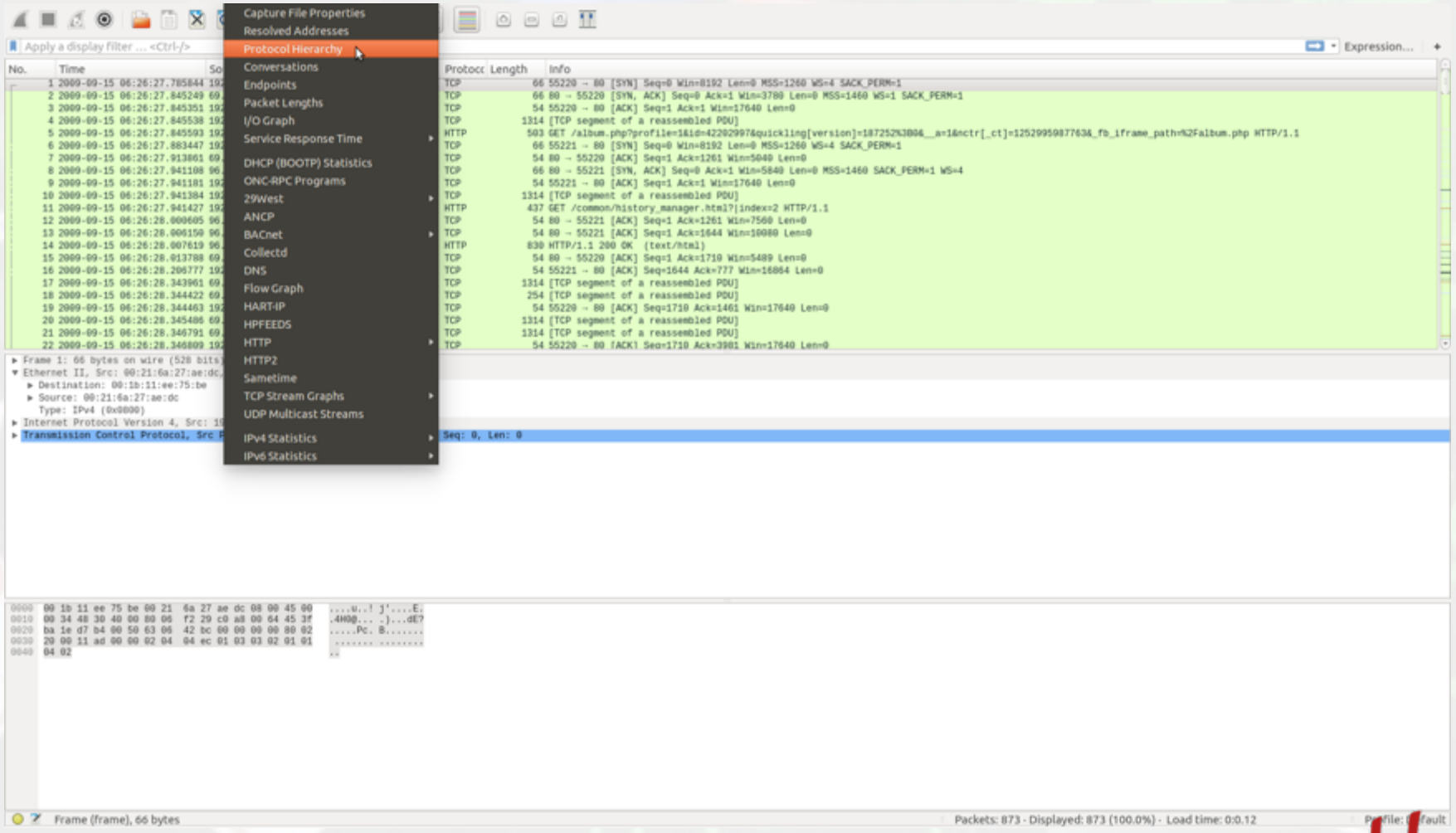
File→Export packet dissections →as “plain text” file

1). In “packet range” option, select “Displayed”

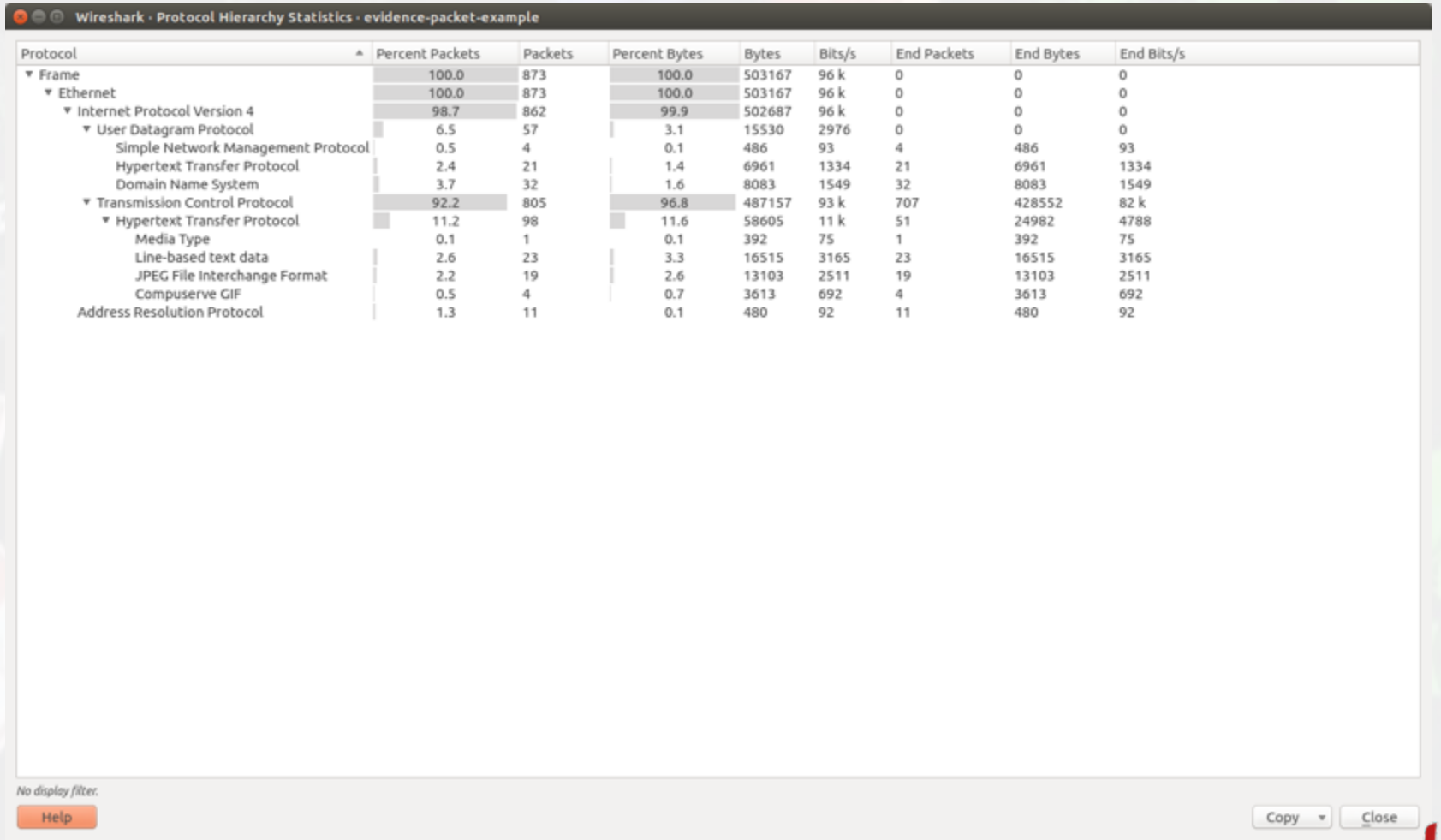
2). In choose “summary line” or “detail”



Protocol Hierarchy

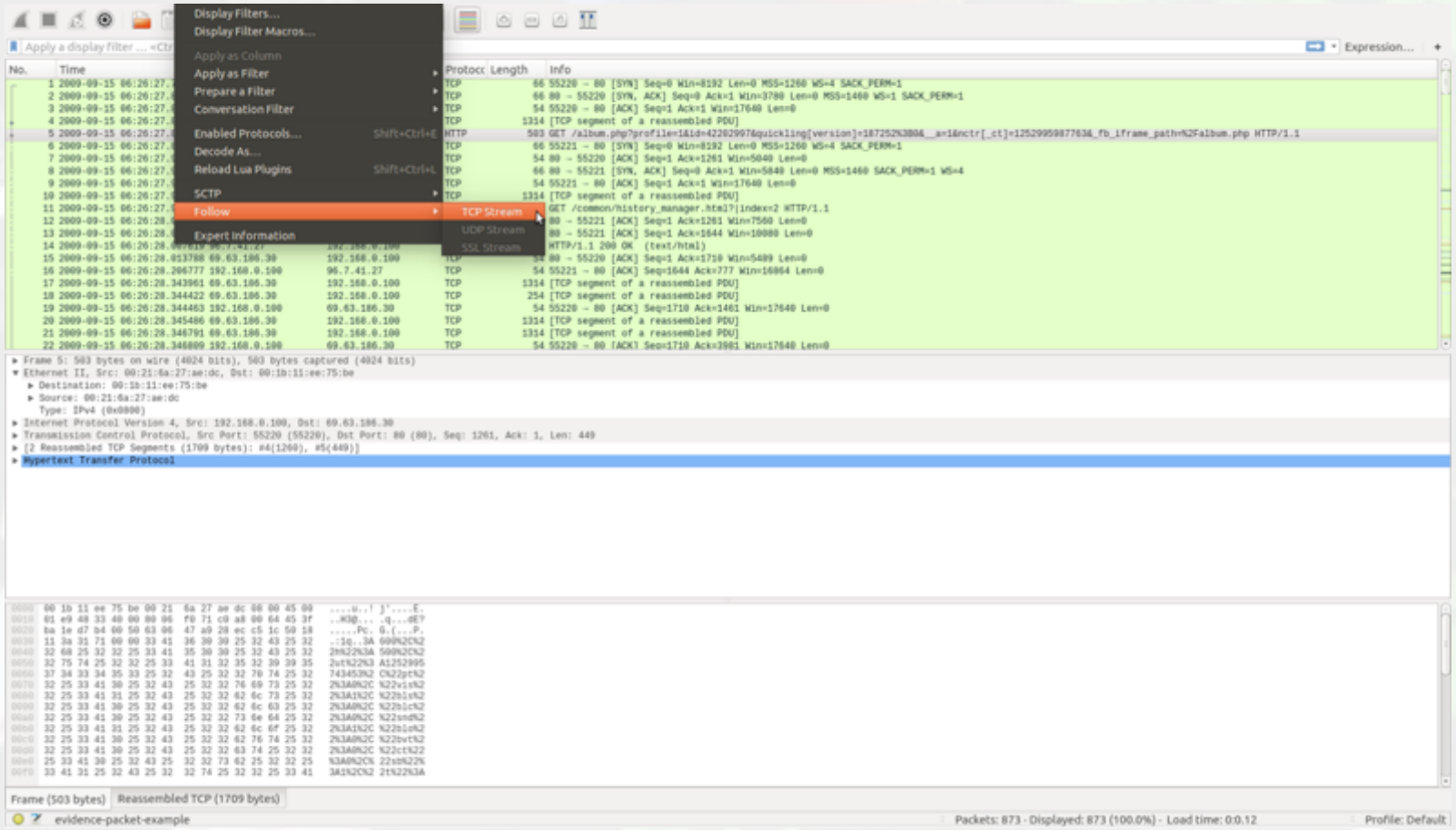


Protocol Hierarchy



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
▼ Frame	100.0	873	100.0	503167	96 k	0	0	0
▼ Ethernet	100.0	873	100.0	503167	96 k	0	0	0
▼ Internet Protocol Version 4	98.7	862	99.9	502687	96 k	0	0	0
▼ User Datagram Protocol	6.5	57	3.1	15530	2976	0	0	0
Simple Network Management Protocol	0.5	4	0.1	486	93	4	486	93
Hypertext Transfer Protocol	2.4	21	1.4	6961	1334	21	6961	1334
Domain Name System	3.7	32	1.6	8083	1549	32	8083	1549
▼ Transmission Control Protocol	92.2	805	96.8	487157	93 k	707	428552	82 k
▼ Hypertext Transfer Protocol	11.2	98	11.6	58605	11 k	51	24982	4788
Media Type	0.1	1	0.1	392	75	1	392	75
Line-based text data	2.6	23	3.3	16515	3165	23	16515	3165
JPEG File Interchange Format	2.2	19	2.6	13103	2511	19	13103	2511
Compuserve GIF	0.5	4	0.7	3613	692	4	3613	692
Address Resolution Protocol	1.3	11	0.1	480	92	11	480	92

Follow TCP Stream



Follow TCP Stream

red - stuff you sent **blue - stuff you get**

Wireshark · Follow TCP Stream (tcp.stream eq 0) · evidence-packet-example

```
GET /album.php?profile=1&id=42202997&quickling[version]=187252%3B0&__a=1&nctr[_ct]=1252995987763&_fb_iframe_path=%2Falbum.php HTTP/1.1
Accept: image/gif, image/jpeg, image/pjpeg, application/x-ms-application, application/vnd.ms-xpsdocument, application/xaml+xml,
application/x-ms-xbap, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, application/x-shockwave-flash, */*
Referer: http://www.facebook.com/home.php?
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; Trident/4.0; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0; InfoPath.2;
.NET CLR 3.5.30729; .NET CLR 3.0.30618)
Accept-Encoding: gzip, deflate
Host: www.facebook.com
Connection: Keep-Alive
Cookie: datr=1252995732-edbc452bccd93a8be5ec16b67ad7704c2d7a525061cc971020a4f; lsd=Au26-;
ABT=2dd086ca2a46e9e50fff44e0ec48cb811st0%3A1253600533%3AB; lxe=brunty11%40marshall.edu; xs=0571f052cf5e4611891c35a695f4d4f5;
c_user=42202997; h_user=dd89ae8ab715; x-referer=http%3A%2F%2Fwww.facebook.com%2Fjosh.brunty%3Fref%3Dprofile; presence=%7B%22v%22%3A2%2C
%22time%22%3A1252995746%2C%22ch%22%3A%7B%22h%22%3A%22channel19%22%2C%22p%22%3A80%2C%22sub%22%3A%5B1%5D%2C%22p_42202997%22%3A0%2C%22ri
%22%3A0%7D%2C%22state%22%3A%7B%22p%22%3A0%2C%22w%22%3A600%2C%22h%22%3A500%2C%22ut%22%3A1252995743453%2C%22pt%22%3A0%2C%22vis%22%3A1%2C
%22bls%22%3A0%2C%22blc%22%3A0%2C%22snd%22%3A1%2C%22blo%22%3A0%2C%22bvt%22%3A0%2C%22ct%22%3A0%2C%22sb%22%3A1%2C%22t%22%3A%7B%7D%2C%22f
%22%3Anull%2C%22uct%22%3A0%2C%22s%22%3A0%7D%2C%22bl%22%3A%7B%22ac%22%3A3%2C%22al%22%3A%7B%7D%2C%22ut%22%3A1252995743%2C%22ud%22%3A400%2C
%22lc%22%3A1%2C%22uo%22%3A%7B%7D%2C%22cvr%22%3A%7B%22r%22%3A1%2C%22ts%22%3A1252995743%7D%7D%7D

HTTP/1.1 200 OK
Date: Tue, 15 Sep 2009 06:26:29 GMT
Server: Apache/1.3.41.fb2
Expires: Mon, 26 Jul 1997 05:00:00 GMT
Cache-Control: private, no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-Cnection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=utf-8
Content-Encoding: gzip

a
.........
12ed
.\{W.:....B7w..v....BO..)o..@.>.-'N.....z..}Gr..@ {.{O...m..G..h4..H..I...`..p.ClEn.l...+..?Y...v....-b.Aw.
z..w.\.-.8.E*.....mJ.d.}}....H..~L.:.Z"Q.D.Un).........Ri.d.d..._xZ..8.].UZu....R..^.....R..P..$.j..,....".2........a.8.G.....
.'..
U'n.. .3j...].i.N.A..h..d....JRh.|c.~.m....E. HHdXA8..V...>.!.....,...4K...>4..5....v..(q{.Y..!q.|...\...I.<........KK..$..vA.%..i.N..@
+....M
...ka*w#&I......q.p}3.....}..q.V.;..J....l. R?1..g....'C&...C..W...d
^X..cx.....(.......~....sR.yEP..6.l....F...d... ...U..........P..Q;"....Md......{.....=. 1P....]x.....{....z...t....
8..DOw.S_..g.A.?.?.....M.(...H.....6^Mt`t. &sKoc+...=.
```

2 client pkt(s), 5 server pkt(s), 1 turn.

Entire conversation (6939 bytes) | Show data as ASCII | Stream 0

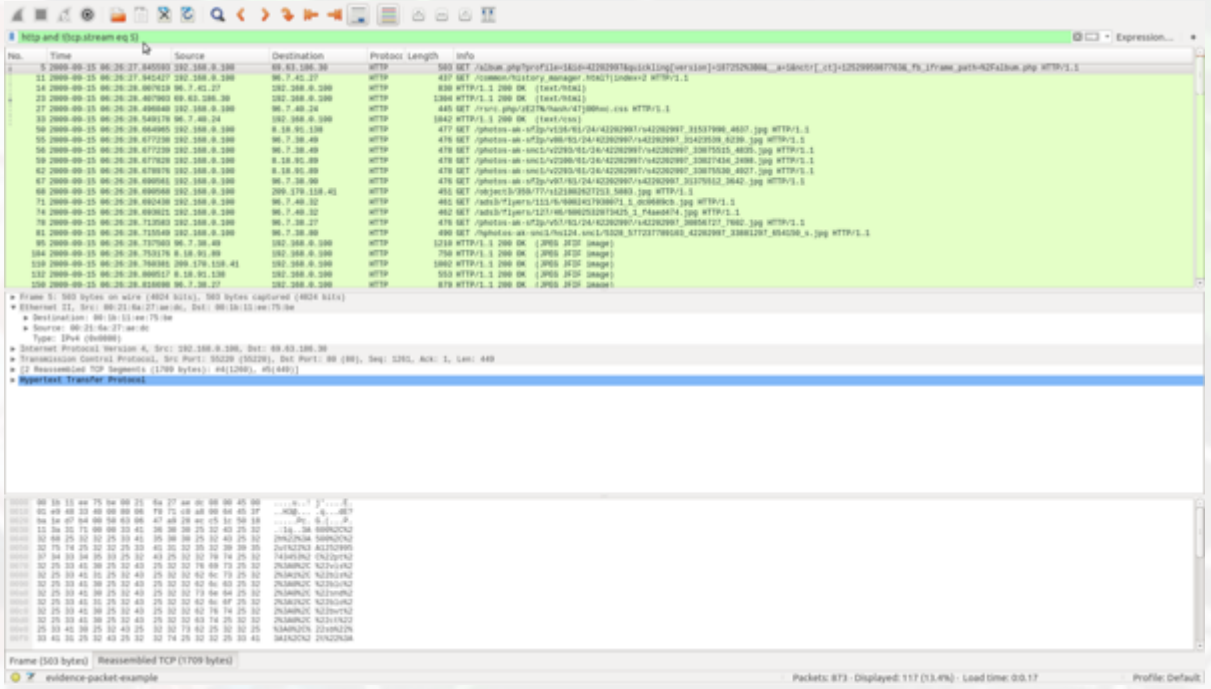
Find: | Find Next

Help | Hide this stream | Print | Save as... | Close

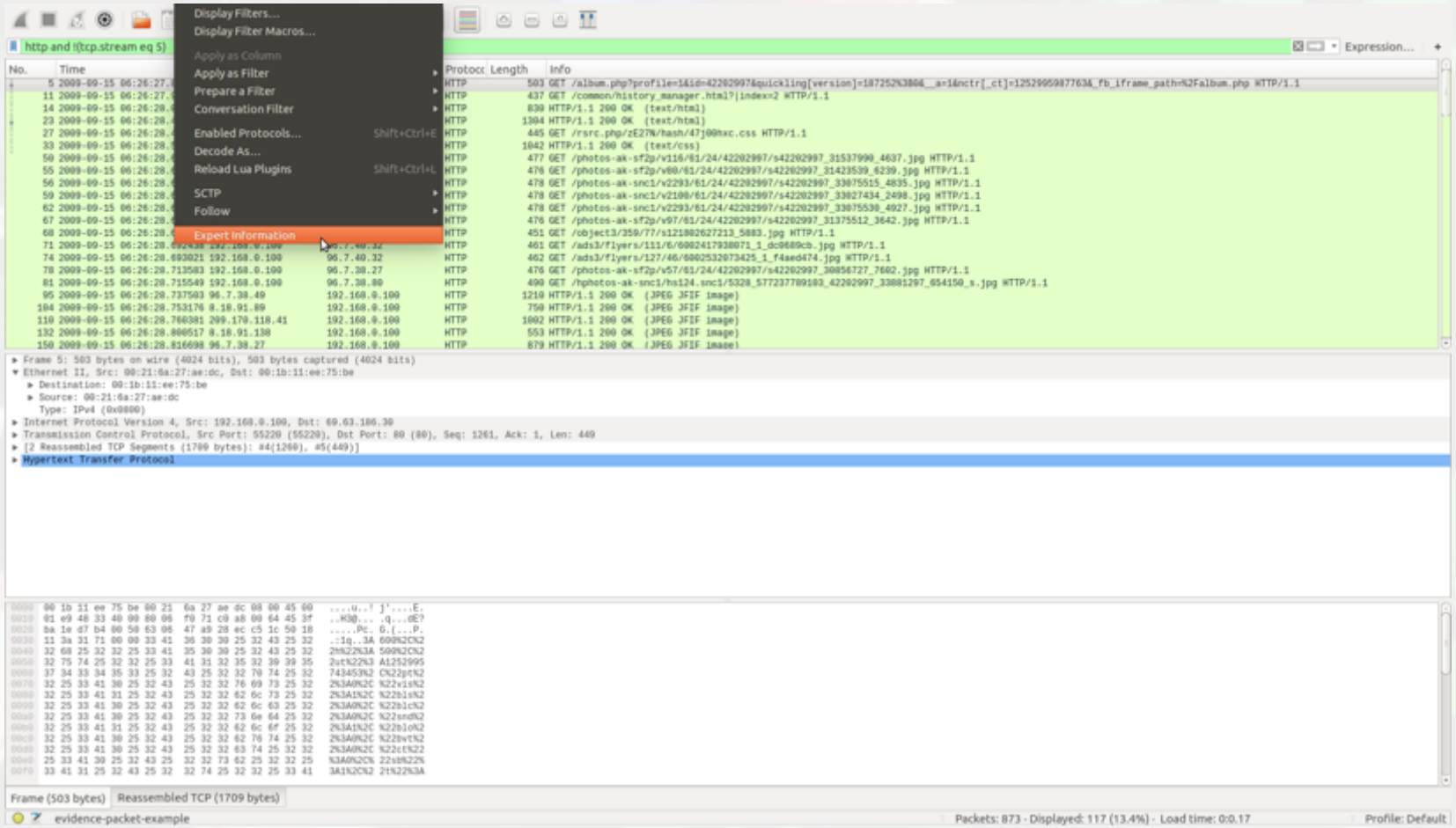
Hack3rCon 8 THE OCHO

Filter out/in Single TCP Stream

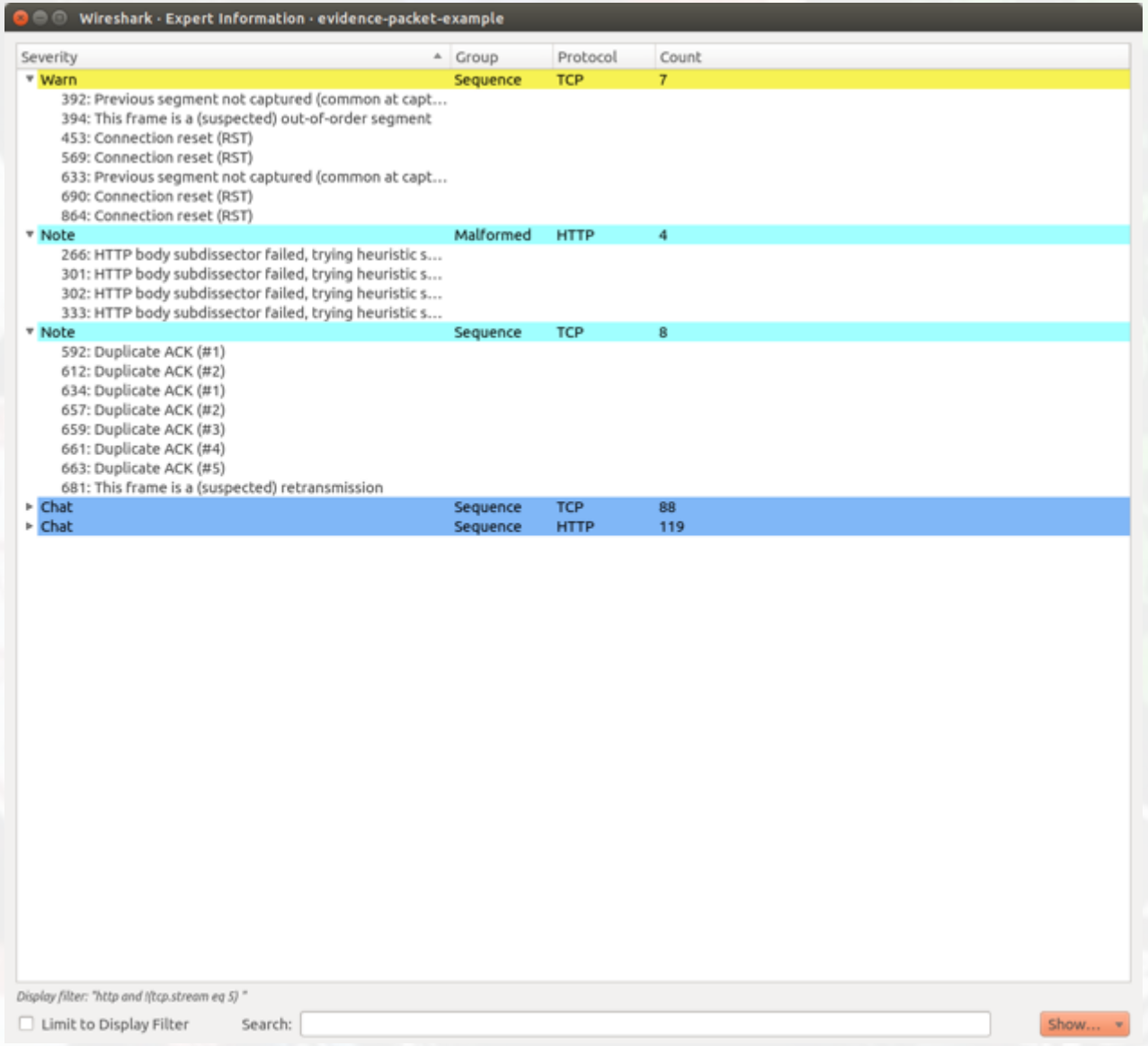
- When click “filter out this TCP stream” in previous page’s box, new filter string will contain like:
  - http and !(tcp.stream eq 5)
- So, if you use “tcp.stream eq 5” as filter string, you keep this HTTP session



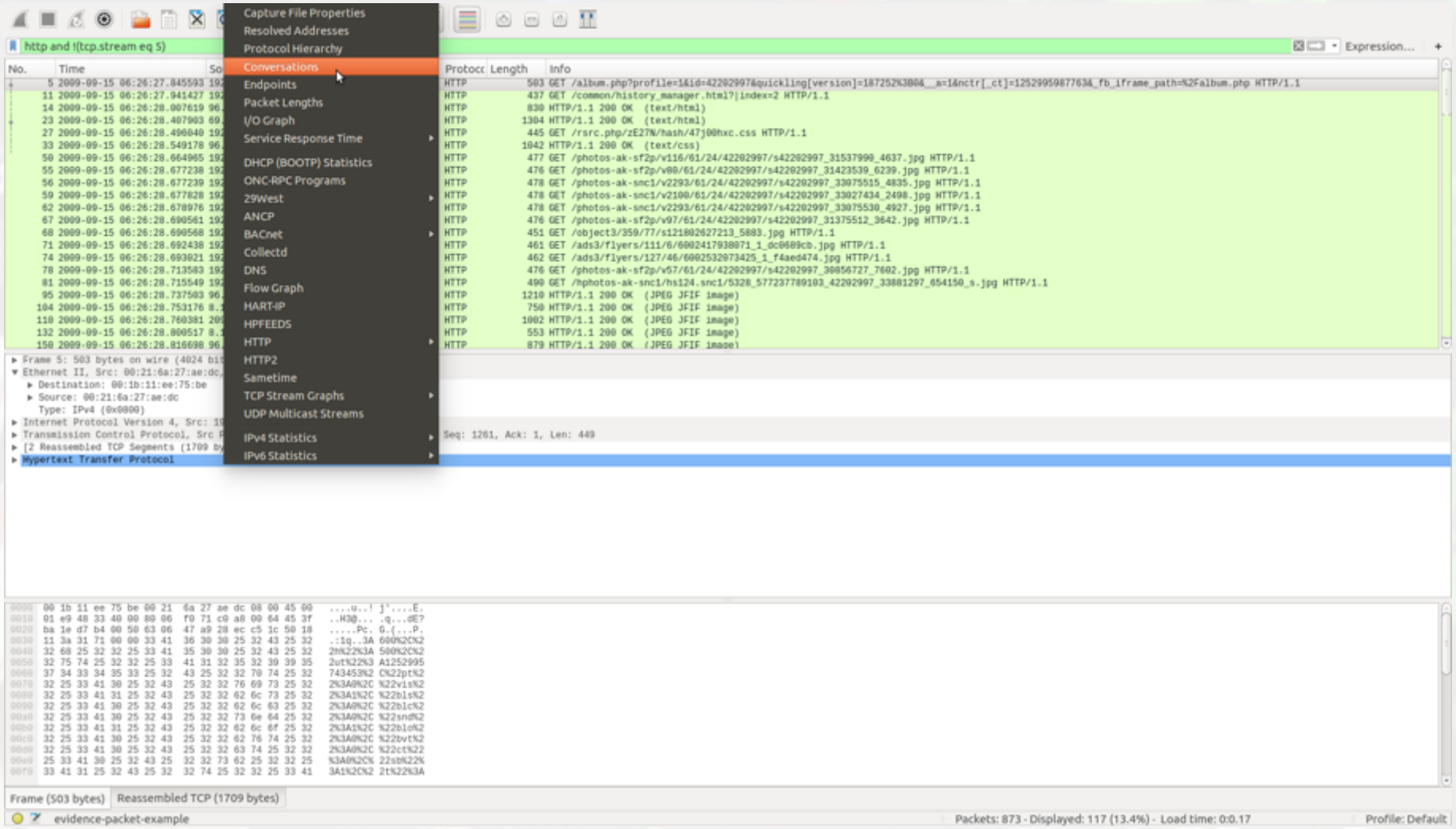
Expert Info


Display Filters...
Display Filter Macros...
Apply as Column
Apply as Filter
Prepare a Filter
Conversation Filter
Enabled Protocols... Shift+Ctrl+E
Decode As...
Reload Lua Plugins Shift+Ctrl+L
SCTP
Follow
Expert Information
http and !(tcp.stream eq 5)
Expression...
No. Time Protocc Length Info
Frame 5: 503 bytes on wire (4024 bits), 503 bytes captured (4024 bits)
Ethernet II, Src: 00:21:6a:27:ae:dc, Dst: 00:1b:11:ee:75:be
Destination: 00:1b:11:ee:75:be
Source: 00:21:6a:27:ae:dc
Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.0.109, Dst: 69.63.186.30
Transmission Control Protocol, Src Port: 55220 (55220), Dst Port: 80 (80), Seq: 1261, Ack: 1, Len: 449
[2 Reassembled TCP Segments (1709 bytes): #4(1260), #5(449)]
Hypertext Transfer Protocol
Frame (503 bytes)
Reassembled TCP (1709 bytes)
evidence-packet-example
Packets: 873 · Displayed: 117 (13.4%) · Load time: 0:0.17
Profile: Default

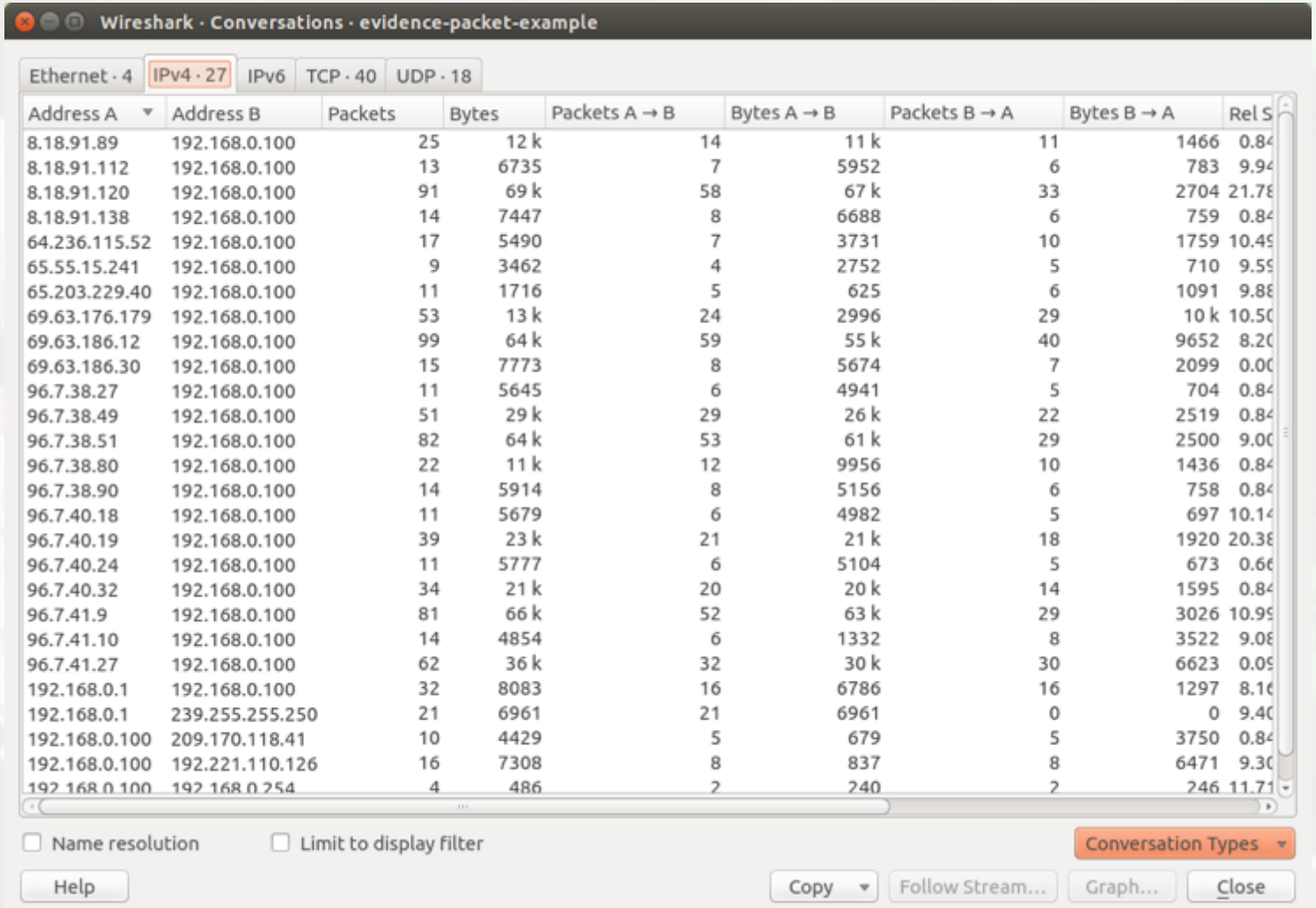
Expert Info


Wireshark · Expert Information · evidence-packet-example
Severity
Group
Protocol
Count
Warn
Sequence
TCP
7
392: Previous segment not captured (common at capt…
394: This frame is a (suspected) out-of-order segment
453: Connection reset (RST)
569: Connection reset (RST)
633: Previous segment not captured (common at capt…
690: Connection reset (RST)
864: Connection reset (RST)
Note
Malformed
HTTP
4
266: HTTP body subdissector failed, trying heuristic s…
301: HTTP body subdissector failed, trying heuristic s…
302: HTTP body subdissector failed, trying heuristic s…
333: HTTP body subdissector failed, trying heuristic s…
Note
Sequence
TCP
8
592: Duplicate ACK (#1)
612: Duplicate ACK (#2)
634: Duplicate ACK (#1)
657: Duplicate ACK (#2)
659: Duplicate ACK (#3)
661: Duplicate ACK (#4)
663: Duplicate ACK (#5)
681: This frame is a (suspected) retransmission
Chat
Sequence
TCP
88
Chat
Sequence
HTTP
119
Display filter: "http and !(tcp.stream eq 5) "
Limit to Display Filter
Search:
Show…

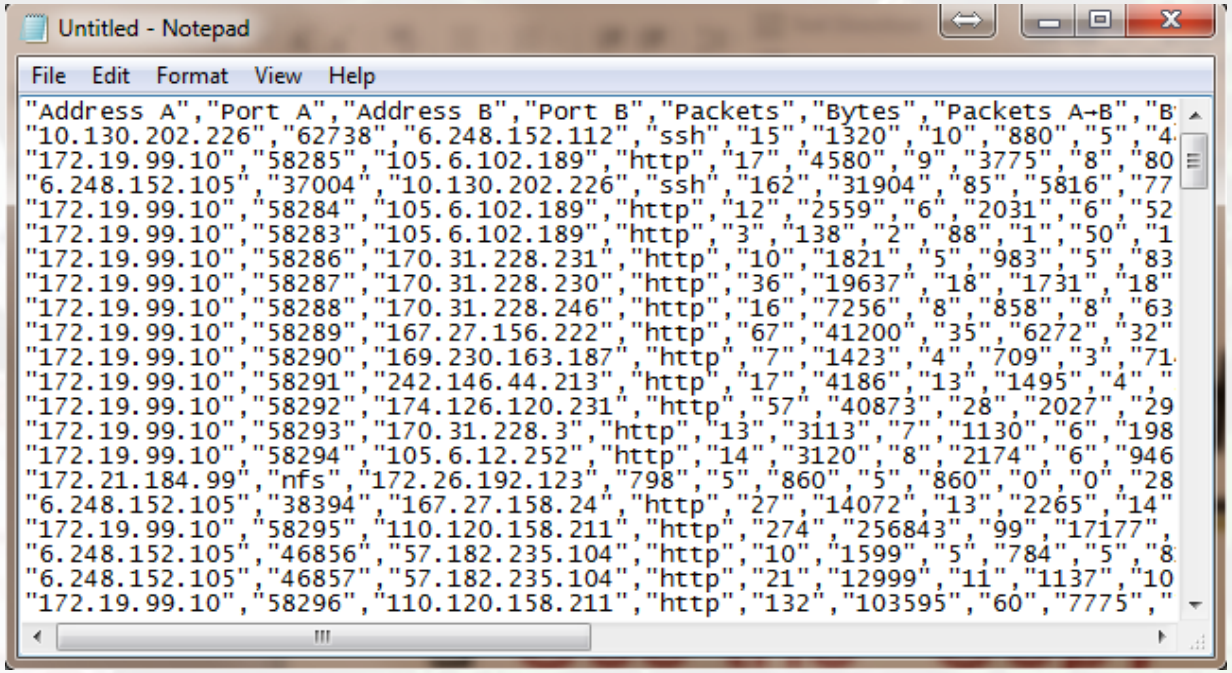
Conversations



Conversations



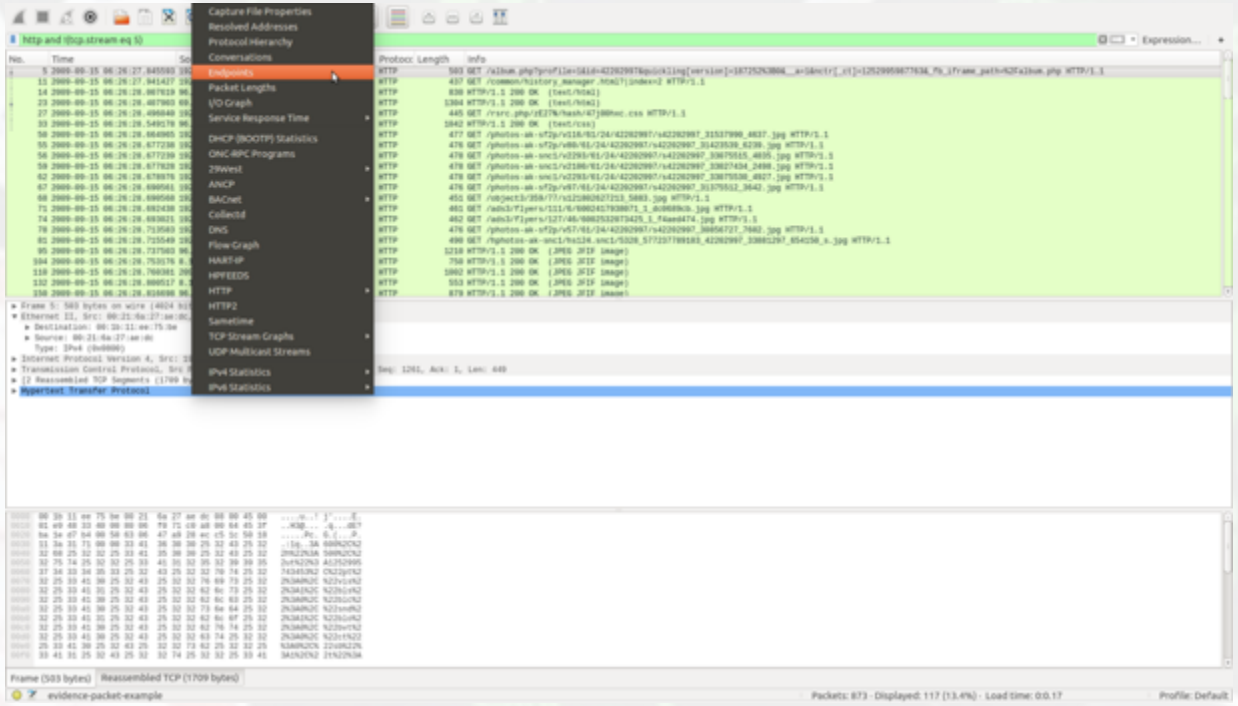
- Use the “Copy” button to copy all text into clipboard



- Then, you can analyze this text file to get what statistics you want (i.e. grep it)
- Or you can use a tool like “ngrep” to search against

Find EndPoint Statistics

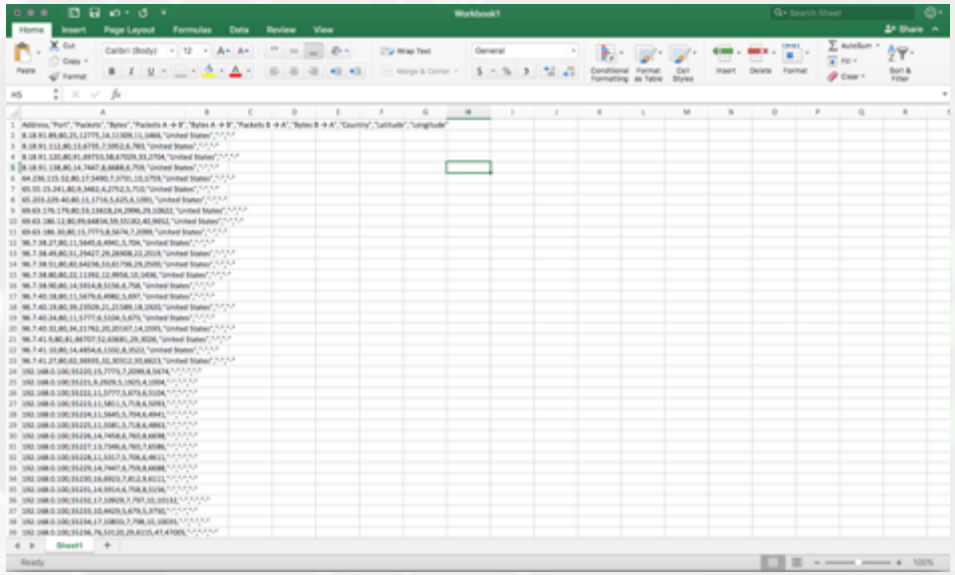
- Menu “statistics” → “endpoints” → “TCP”



- You can sort by field
- “Tx” : transmit “Rx” : receive

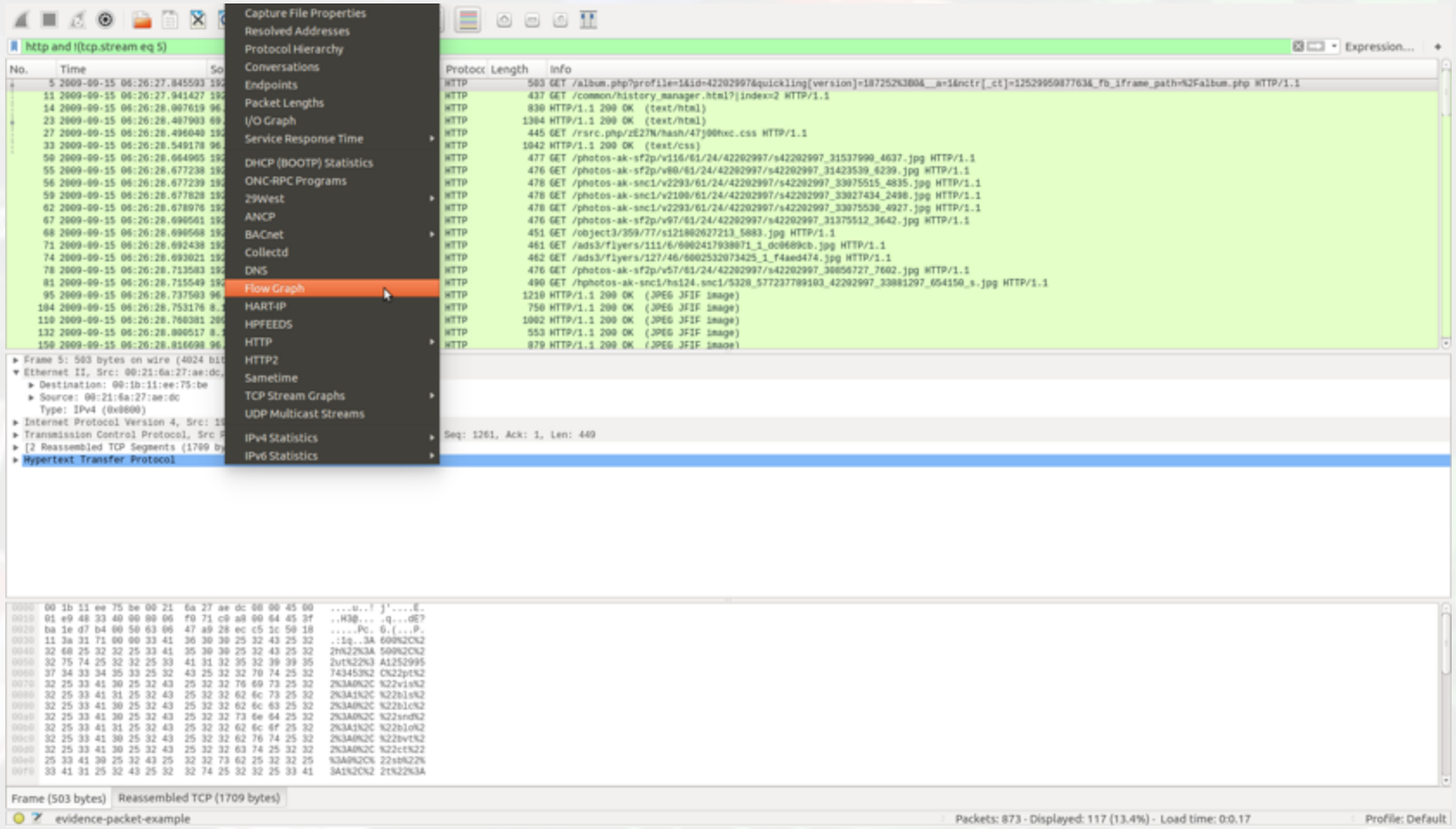
Find EndPoint Statistics

- Use the “Copy” button to copy all text into clipboard

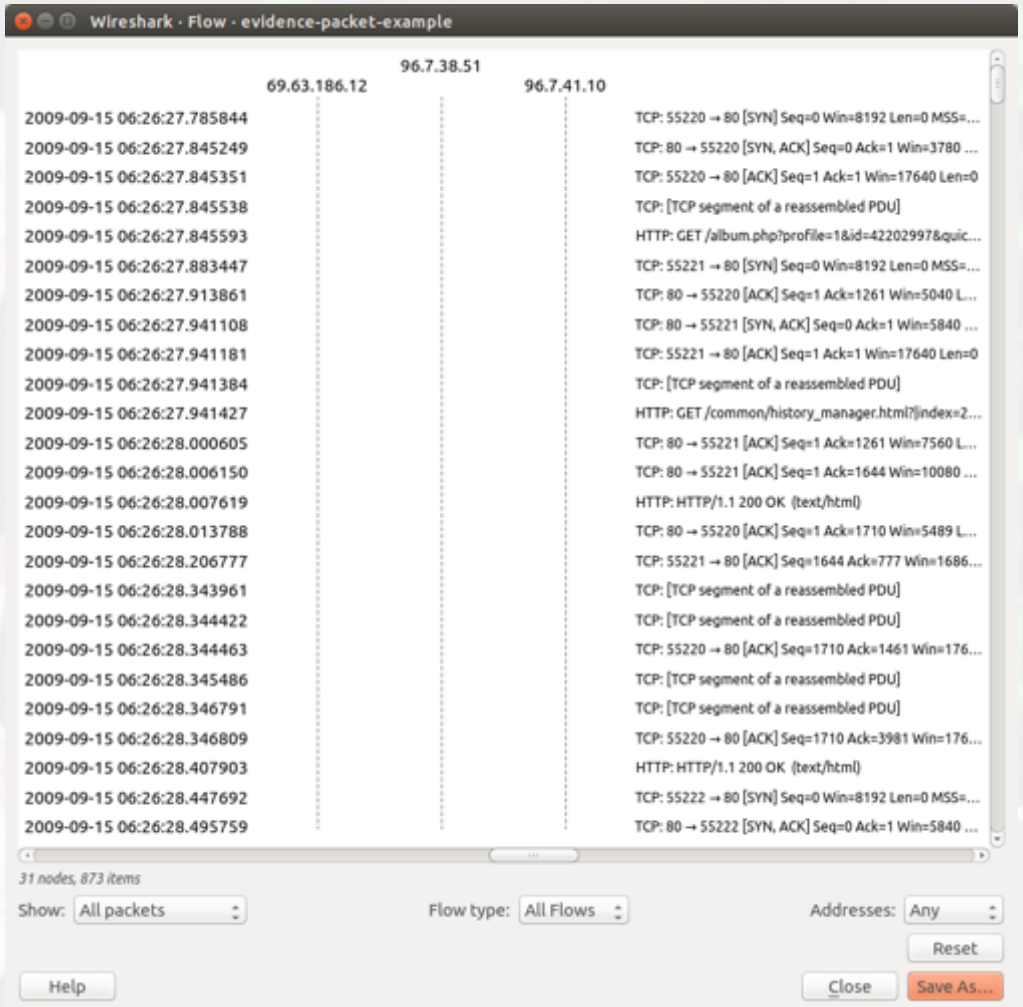


- Then, you can analyze this text file to get what statistics you want

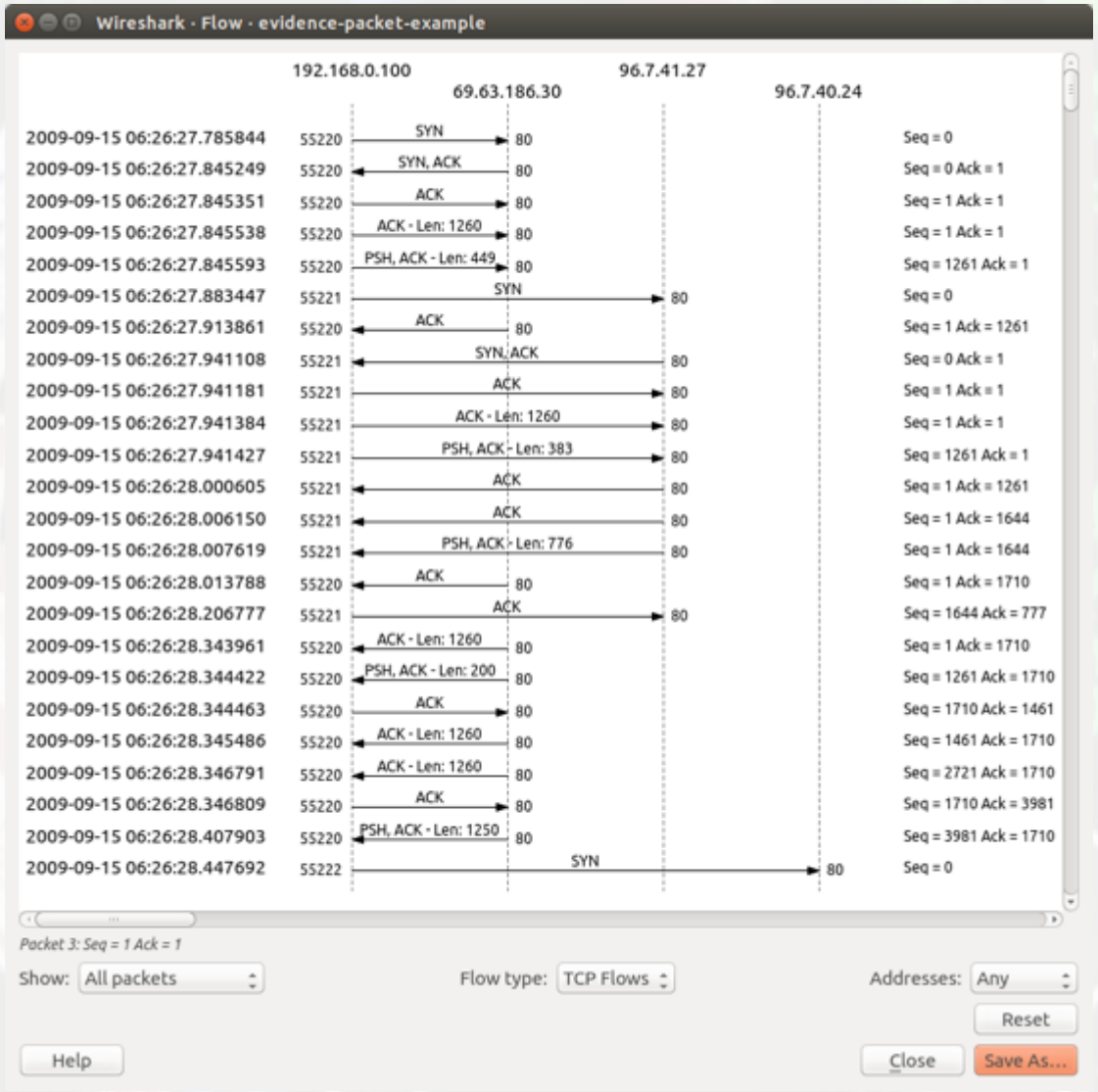
Flow Graphs


Capture File Properties
Resolved Addresses
Protocol Hierarchy
Conversations
Endpoints
Packet Lengths
I/O Graph
Service Response Time
DHCP (BOOTP) Statistics
ONC-RPC Programs
29West
ANCP
BACnet
Collectd
DNS
Flow Graph
HART-IP
HPFEEDS
HTTP
HTTP2
Sametime
TCP Stream Graphs
UDP Multicast Streams
IPv4 Statistics
IPv6 Statistics
http and !(tcp.stream eq 5)
Expression...
No.
Time
Protocc
Length
Info
Frame (503 bytes)
Reassembled TCP (1709 bytes)
evidence-packet-example
Packets: 873 · Displayed: 117 (13.4%) · Load time: 0:0.17
Profile: Default

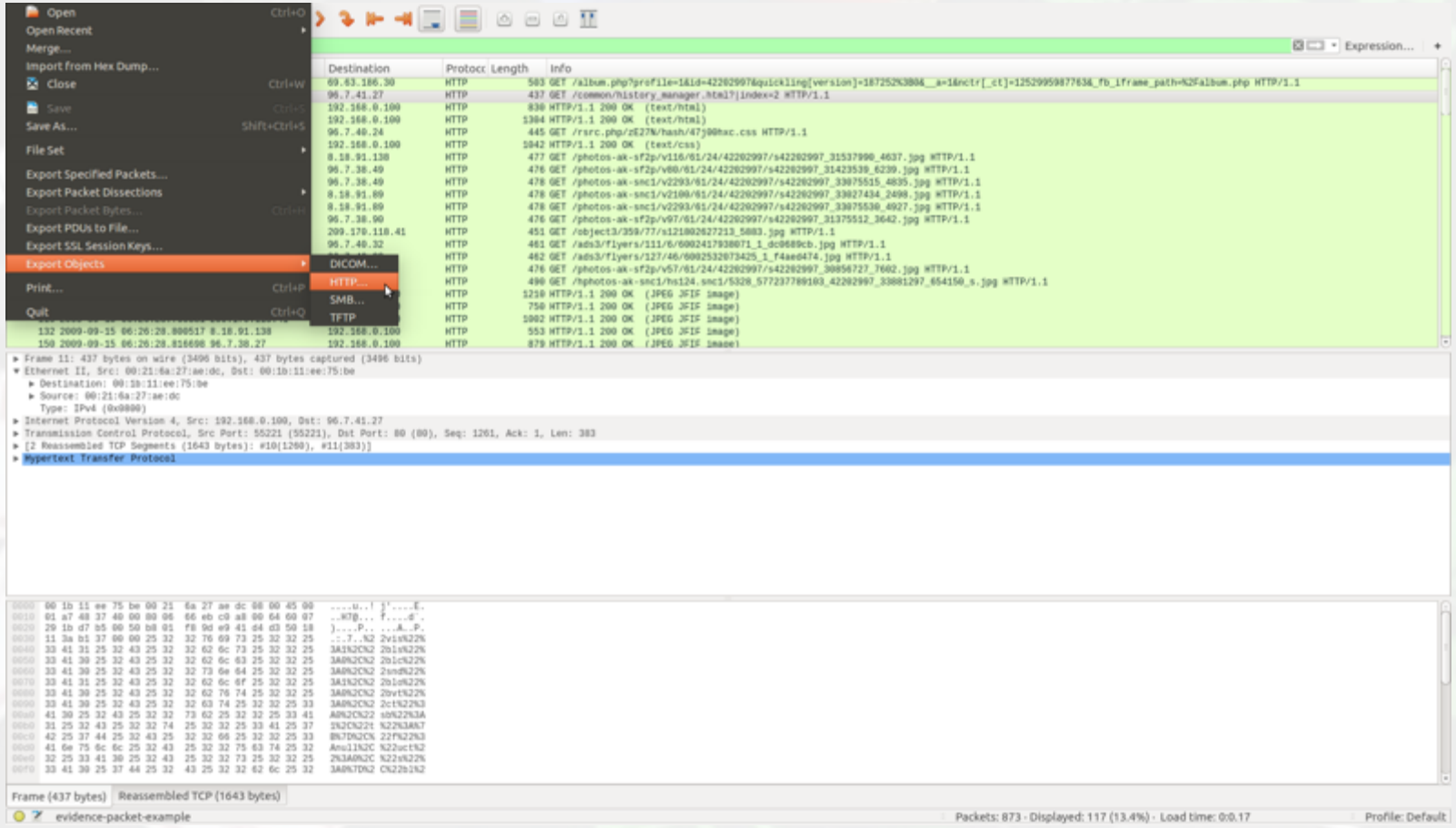
Flow Graphs


Wireshark · Flow · evidence-packet-example
96.7.38.51
69.63.186.12
96.7.41.10
2009-09-15 06:26:27.785844 TCP: 55220 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=...
2009-09-15 06:26:27.845249 TCP: 80 → 55220 [SYN, ACK] Seq=0 Ack=1 Win=3780 ...
2009-09-15 06:26:27.845351 TCP: 55220 → 80 [ACK] Seq=1 Ack=1 Win=17640 Len=0
2009-09-15 06:26:27.845538 TCP: [TCP segment of a reassembled PDU]
2009-09-15 06:26:27.845593 HTTP: GET /album.php?profile=1&id=42202997&quic...
2009-09-15 06:26:27.883447 TCP: 55221 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=...
2009-09-15 06:26:27.913861 TCP: 80 → 55220 [ACK] Seq=1 Ack=1261 Win=5040 L...
2009-09-15 06:26:27.941108 TCP: 80 → 55221 [SYN, ACK] Seq=0 Ack=1 Win=5840 ...
2009-09-15 06:26:27.941181 TCP: 55221 → 80 [ACK] Seq=1 Ack=1 Win=17640 Len=0
2009-09-15 06:26:27.941384 TCP: [TCP segment of a reassembled PDU]
2009-09-15 06:26:27.941427 HTTP: GET /common/history_manager.html?jindex=2...
2009-09-15 06:26:28.000605 TCP: 80 → 55221 [ACK] Seq=1 Ack=1261 Win=7560 L...
2009-09-15 06:26:28.006150 TCP: 80 → 55221 [ACK] Seq=1 Ack=1644 Win=10080 ...
2009-09-15 06:26:28.007619 HTTP: HTTP/1.1 200 OK (text/html)
2009-09-15 06:26:28.013788 TCP: 80 → 55220 [ACK] Seq=1 Ack=1710 Win=5489 L...
2009-09-15 06:26:28.206777 TCP: 55221 → 80 [ACK] Seq=1644 Ack=777 Win=1686...
2009-09-15 06:26:28.343961 TCP: [TCP segment of a reassembled PDU]
2009-09-15 06:26:28.344422 TCP: [TCP segment of a reassembled PDU]
2009-09-15 06:26:28.344463 TCP: 55220 → 80 [ACK] Seq=1710 Ack=1461 Win=176...
2009-09-15 06:26:28.345486 TCP: [TCP segment of a reassembled PDU]
2009-09-15 06:26:28.346791 TCP: [TCP segment of a reassembled PDU]
2009-09-15 06:26:28.346809 TCP: 55220 → 80 [ACK] Seq=1710 Ack=3981 Win=176...
2009-09-15 06:26:28.407903 HTTP: HTTP/1.1 200 OK (text/html)
2009-09-15 06:26:28.447692 TCP: 55222 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=...
2009-09-15 06:26:28.495759 TCP: 80 → 55222 [SYN, ACK] Seq=0 Ack=1 Win=5840 ...
31 nodes, 873 items
Show: All packets
Flow type: All Flows
Addresses: Any
Reset
Help
Close
Save As...

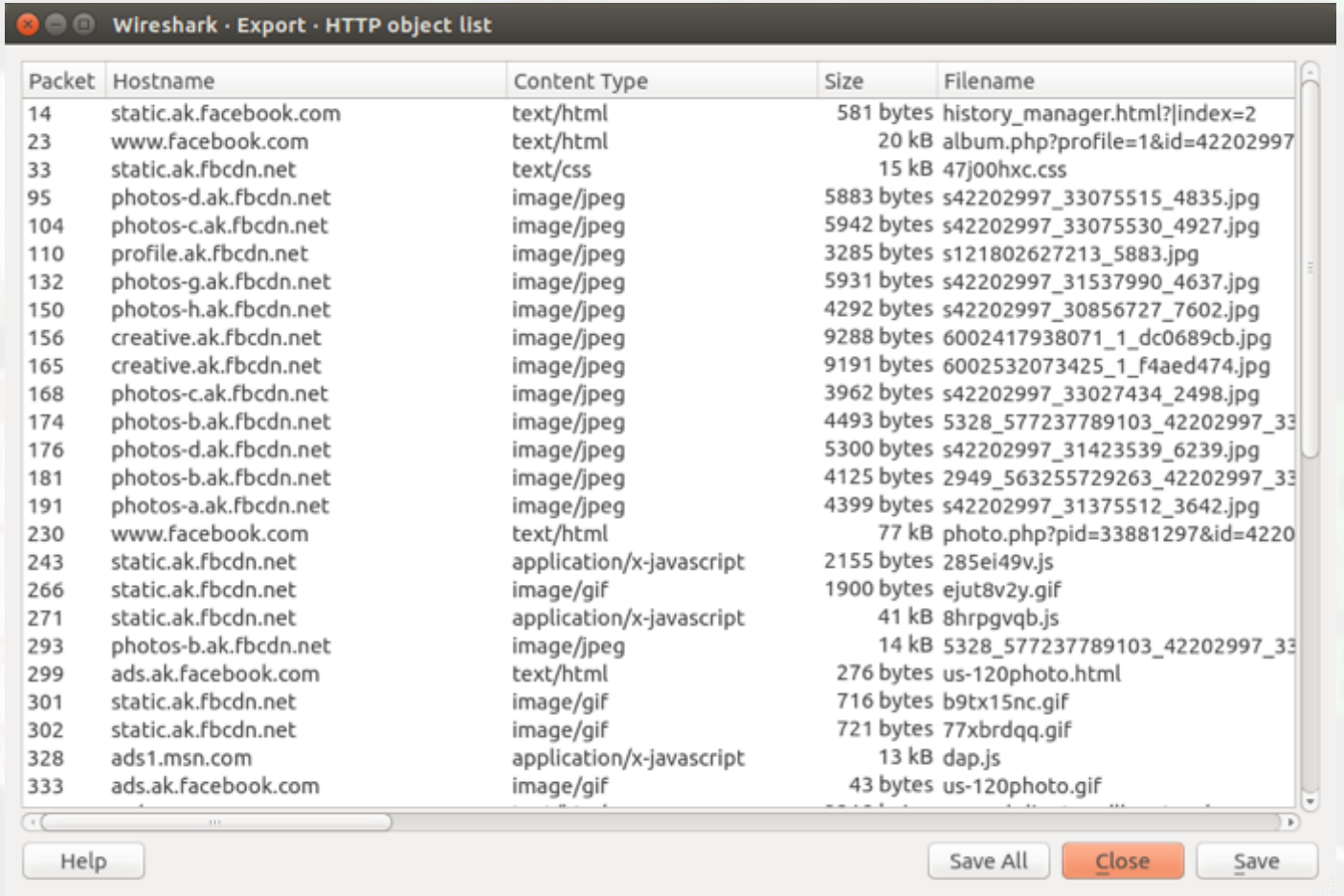
Flow Graphs


Wireshark · Flow · evidence-packet-example
192.168.0.100
69.63.186.30
96.7.41.27
96.7.40.24
2009-09-15 06:26:27.785844 55220 SYN 80 Seq = 0
2009-09-15 06:26:27.845249 55220 SYN, ACK 80 Seq = 0 Ack = 1
2009-09-15 06:26:27.845351 55220 ACK 80 Seq = 1 Ack = 1
2009-09-15 06:26:27.845538 55220 ACK - Len: 1260 80 Seq = 1 Ack = 1
2009-09-15 06:26:27.845593 55220 PSH, ACK - Len: 449 80 Seq = 1261 Ack = 1
2009-09-15 06:26:27.883447 55221 SYN 80 Seq = 0
2009-09-15 06:26:27.913861 55220 ACK 80 Seq = 1 Ack = 1261
2009-09-15 06:26:27.941108 55221 SYN, ACK 80 Seq = 0 Ack = 1
2009-09-15 06:26:27.941181 55221 ACK 80 Seq = 1 Ack = 1
2009-09-15 06:26:27.941384 55221 ACK - Len: 1260 80 Seq = 1 Ack = 1
2009-09-15 06:26:27.941427 55221 PSH, ACK - Len: 383 80 Seq = 1261 Ack = 1
2009-09-15 06:26:28.000605 55221 ACK 80 Seq = 1 Ack = 1261
2009-09-15 06:26:28.006150 55221 ACK 80 Seq = 1 Ack = 1644
2009-09-15 06:26:28.007619 55221 PSH, ACK - Len: 776 80 Seq = 1 Ack = 1644
2009-09-15 06:26:28.013788 55220 ACK 80 Seq = 1 Ack = 1710
2009-09-15 06:26:28.206777 55221 ACK 80 Seq = 1644 Ack = 777
2009-09-15 06:26:28.343961 55220 ACK - Len: 1260 80 Seq = 1 Ack = 1710
2009-09-15 06:26:28.344422 55220 PSH, ACK - Len: 200 80 Seq = 1261 Ack = 1710
2009-09-15 06:26:28.344463 55220 ACK 80 Seq = 1710 Ack = 1461
2009-09-15 06:26:28.345486 55220 ACK - Len: 1260 80 Seq = 1461 Ack = 1710
2009-09-15 06:26:28.346791 55220 ACK - Len: 1260 80 Seq = 2721 Ack = 1710
2009-09-15 06:26:28.346809 55220 ACK 80 Seq = 1710 Ack = 3981
2009-09-15 06:26:28.407903 55220 PSH, ACK - Len: 1250 80 Seq = 3981 Ack = 1710
2009-09-15 06:26:28.447692 55222 SYN 80 Seq = 0
Packet 3: Seq = 1 Ack = 1
Show: All packets
Flow type: TCP Flows
Addresses: Any
Reset
Help
Close
Save As...

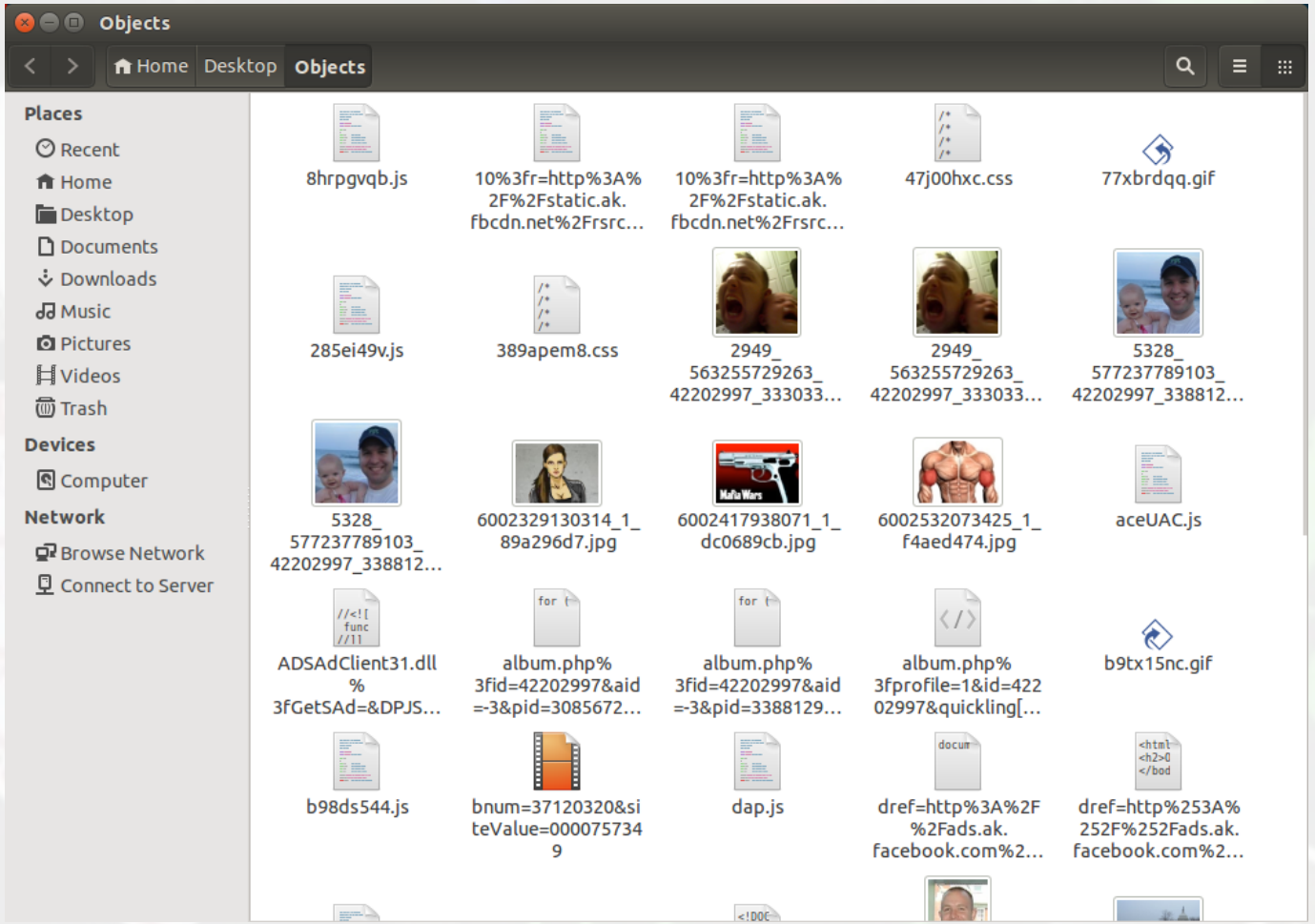
Export HTTP


Open Ctrl+O
Open Recent
Merge...
Import from Hex Dump...
Close Ctrl+W
Save Ctrl+S
Save As... Shift+Ctrl+S
File Set
Export Specified Packets...
Export Packet Dissections
Export Packet Bytes... Ctrl+H
Export PDUs to File...
Export SSL Session Keys...
Export Objects
Print... Ctrl+P
Quit Ctrl+Q
DICOM...
HTTP...
SMB...
TFTP
Expression...
Destination Protocc Length Info
Frame 11: 437 bytes on wire (3496 bits), 437 bytes captured (3496 bits)
Ethernet II, Src: 00:21:6a:27:ae:dc, Dst: 00:1b:11:ee:75:be
Internet Protocol Version 4, Src: 192.168.0.100, Dst: 96.7.41.27
Transmission Control Protocol, Src Port: 55221 (55221), Dst Port: 80 (80), Seq: 1261, Ack: 1, Len: 383
[2 Reassembled TCP Segments (1643 bytes): #10(1260), #11(383)]
Hypertext Transfer Protocol
Frame (437 bytes)
Reassembled TCP (1643 bytes)
evidence-packet-example
Packets: 873 · Displayed: 117 (13.4%) · Load time: 0:0.17
Profile: Default

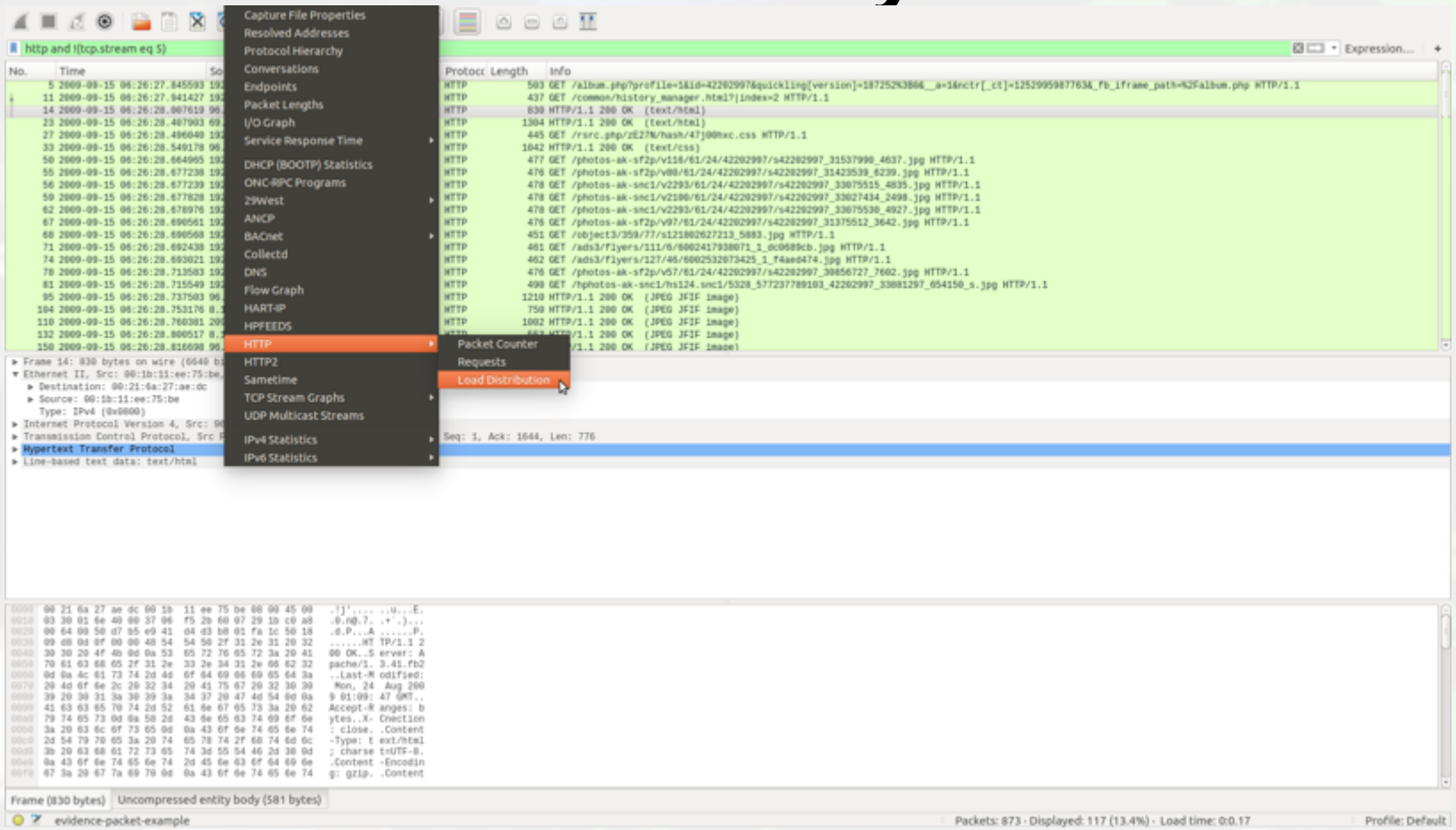
Export HTTP Objects



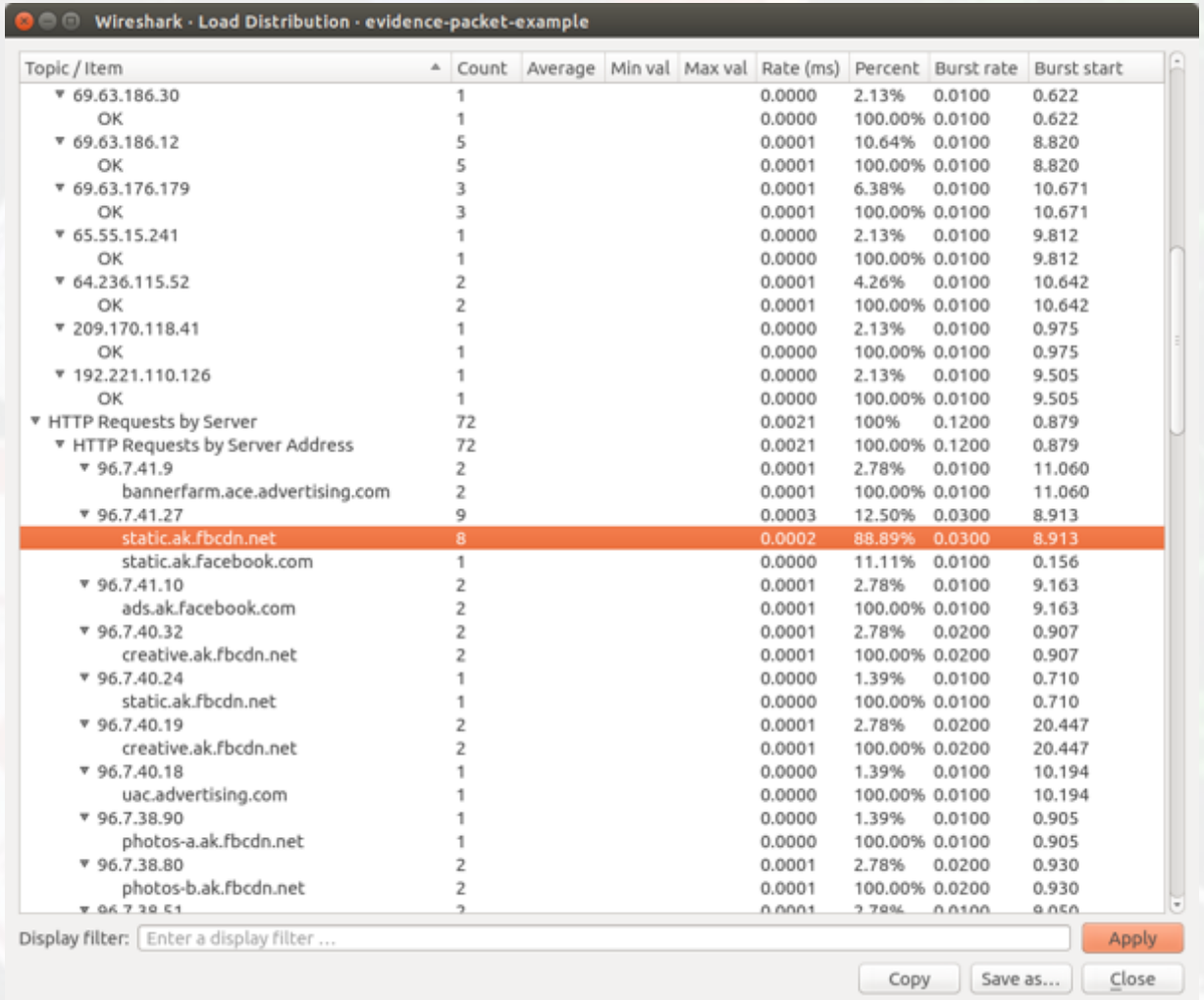
Export HTTP Objects



HTTP Analysis

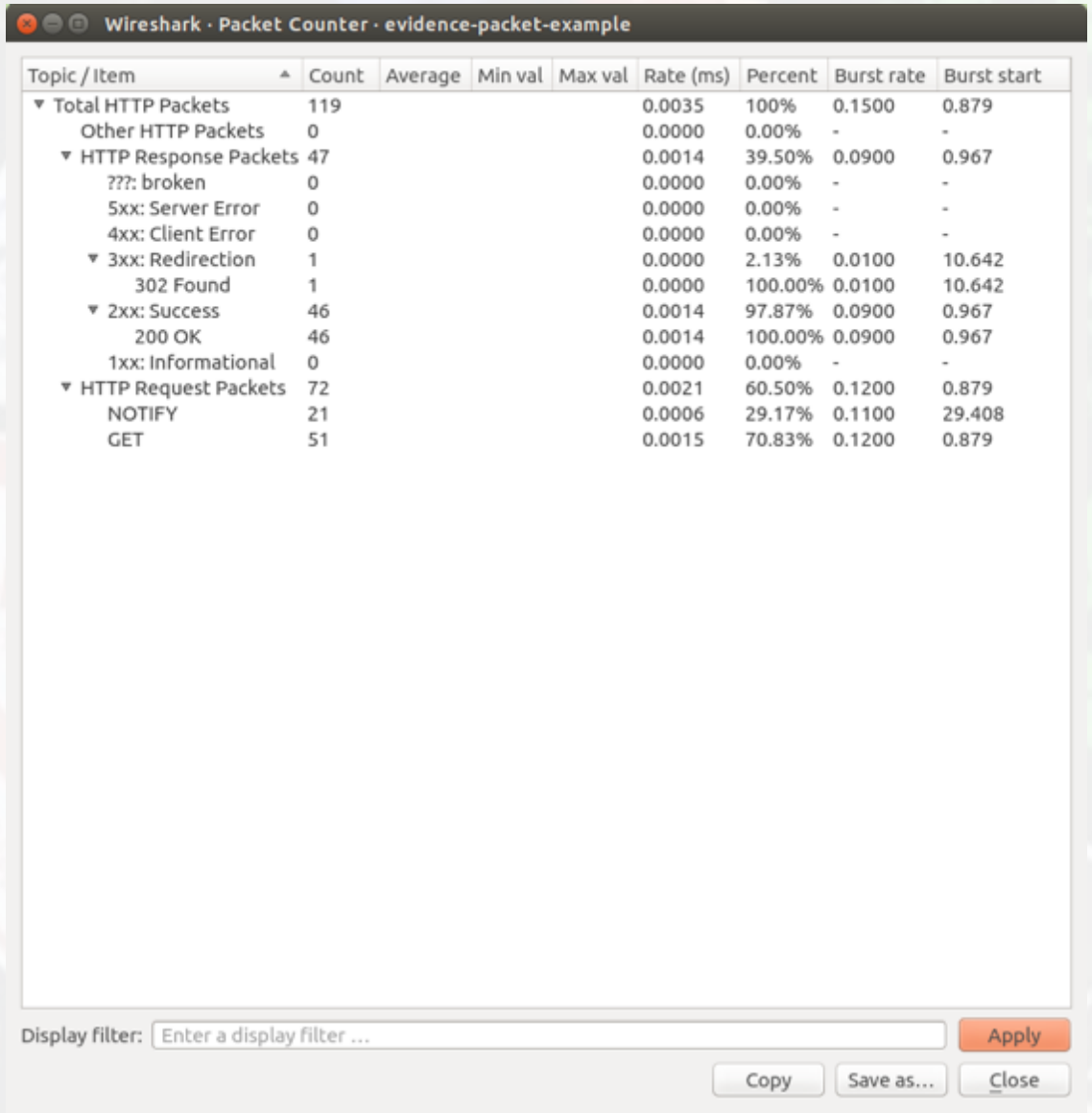


HTTP Analysis – Load Distribution

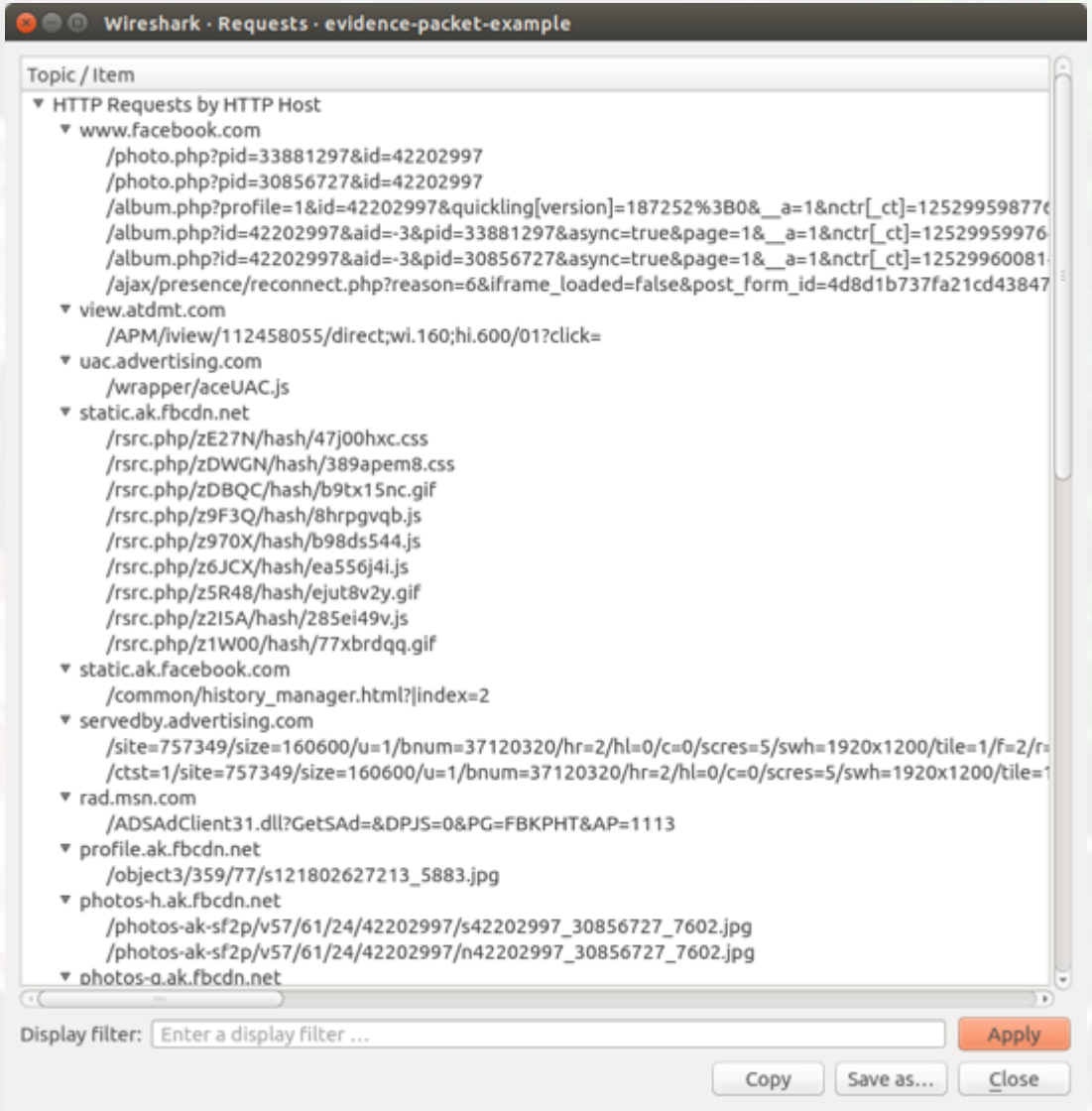


Use the display filter to only Show selected traffic

HTTP Analysis – Packet Counter



HTTP Analysis – Requests



Improving WireShark Performance

- Don’t use capture filters (use post-capture display filters)
- Increase your read buffer size (if possible)
- Don’t update the screen dynamically
- Get a faster computer (do I have to say this!!)
- Run in a linux distro (less overhead)
- Use a TAP (Hak5)
- Don’t resolve names (MAC’s and IP’s) **false positives

Post-Processing Text File

- For saved text-format packet files, further analysis needs coding or special tools (tcpxtract works well here)
- One useful Linux tools for packets: Grep & ngrep
  - On Windows: PowerGrep http://www.powergrep.com/
  - Command-line based utility for searching plain-text data sets for lines matching a regular expression.

Basic usage of Grep

- Command-line text-search program in Linux
- Some useful usage:
  - Grep ‘word’ filename # find lines with ‘word’
  - Grep –v ‘word’ filename # find lines without ‘word’
  - Grep ‘^word’ filename # find lines beginning with ‘word’
  - Grep ‘word’ filename > file2 # output lines with ‘word’ to file2
  - ls -l | grep rwxrwxrwx # list files that have ‘rwxrwxrwx’ feature
  - grep '^[0-4]‘ filename # find lines beginning with any of the numbers from 0-4
  - Grep –c ‘word’ filename # find lines with ‘word’ and print out the number of these lines
  - Grep –i ‘word’ filename # find lines with ‘word’ regardless of case
- NGREP also exists for Grep’ing of packet capture files
- Many tutorials on grep online
  - http://www.cyberciti.biz/faq/howto-use-grep-command-in-linux-unix/
  - http://www.thegeekstuff.com/2009/03/15-practical-unix-grep-command-examples/

Lab Exercise

File: evidence1.pcap

What we should be looking for:

- Determine PC's used
- IP addresses of devices
- Mac Addresses of devices
- Protocols Found in packets
- User-related data
- Exported Files
- Endpoints

Disclaimer: The "investigation" part of this exercise will be covered in Saturday's "Network Forensics" session

Questions?