



# **Traffic Analysis—Wireshark**

**CIS 6395, Incident Response Technologies**

**Fall 2016, Dr. Cliff Zou**

[czou@cs.ucf.edu](mailto:czou@cs.ucf.edu)



# Motivation for Network Monitoring

- Essential for Network Management
  - Router and Firewall policy
  - Detecting abnormal/error in networking
  - Access control
- Security Management
  - Detecting abnormal traffic
  - Traffic log for future forensic analysis



# Tools Overview

- **Tcpdump**

- Unix-based command-line tool used to intercept packets
    - Including **filtering** to just the packets of interest

- **Wireshark**

- GUI for displaying tcpdump/tshark packet traces

# Tcpdump example

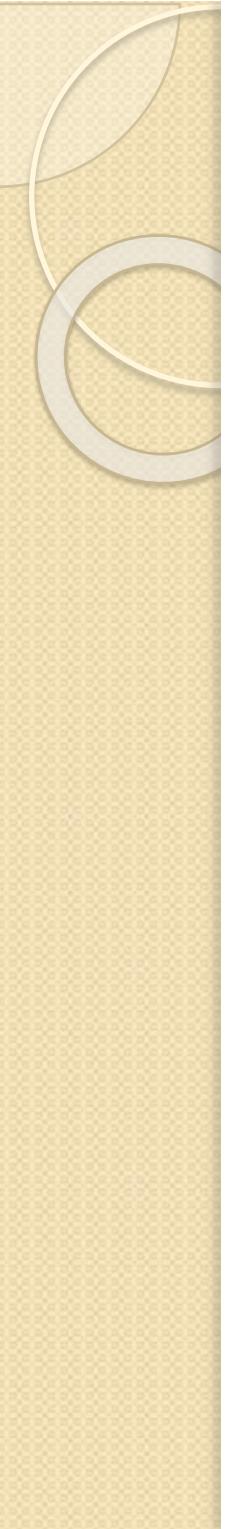
- Ran tcpdump on a Unix machine
- First few lines of the output:

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



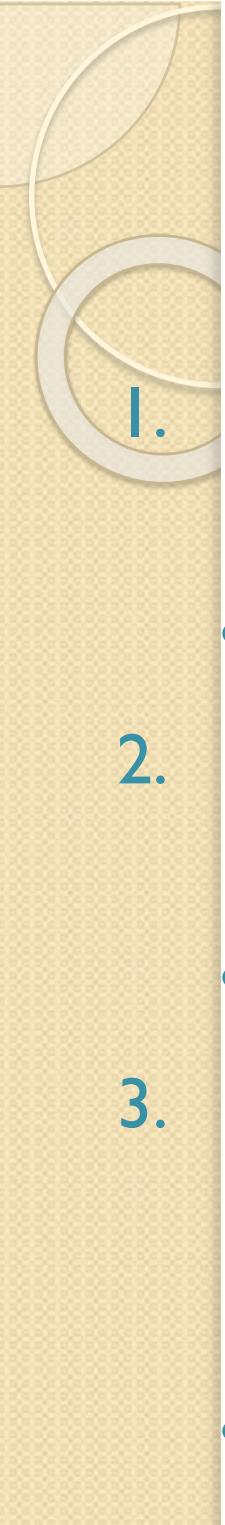
# Filters

- We are often not interested in all packets flowing through the network
- Use filters to capture only packets of interest to us
- How to write filters?
  - Refer the tcpdump/tshark man page
  - Many example webpages on the Internet



# Example

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



## Example (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"`



# Example (contd.)

- I. Capture only packets destined to longwood.eecs.ucf.edu
  - `tcpdump "dst host longwood.eecs.ucf.edu"`
2. Capture both DNS packets and TCP packets to/from longwood.eecs.ucf.edu
  - `tcpdump "(tcp and host longwood.eecs.ucf.edu) or udp port 53"`

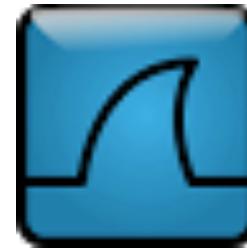


# Running tcpdump

- 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 in Vmware on your windows machine
- Tcpdump for Windows
  - WinDump: <http://www.winpcap.org/windump/>
    - Free software

# So What is Wireshark?

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



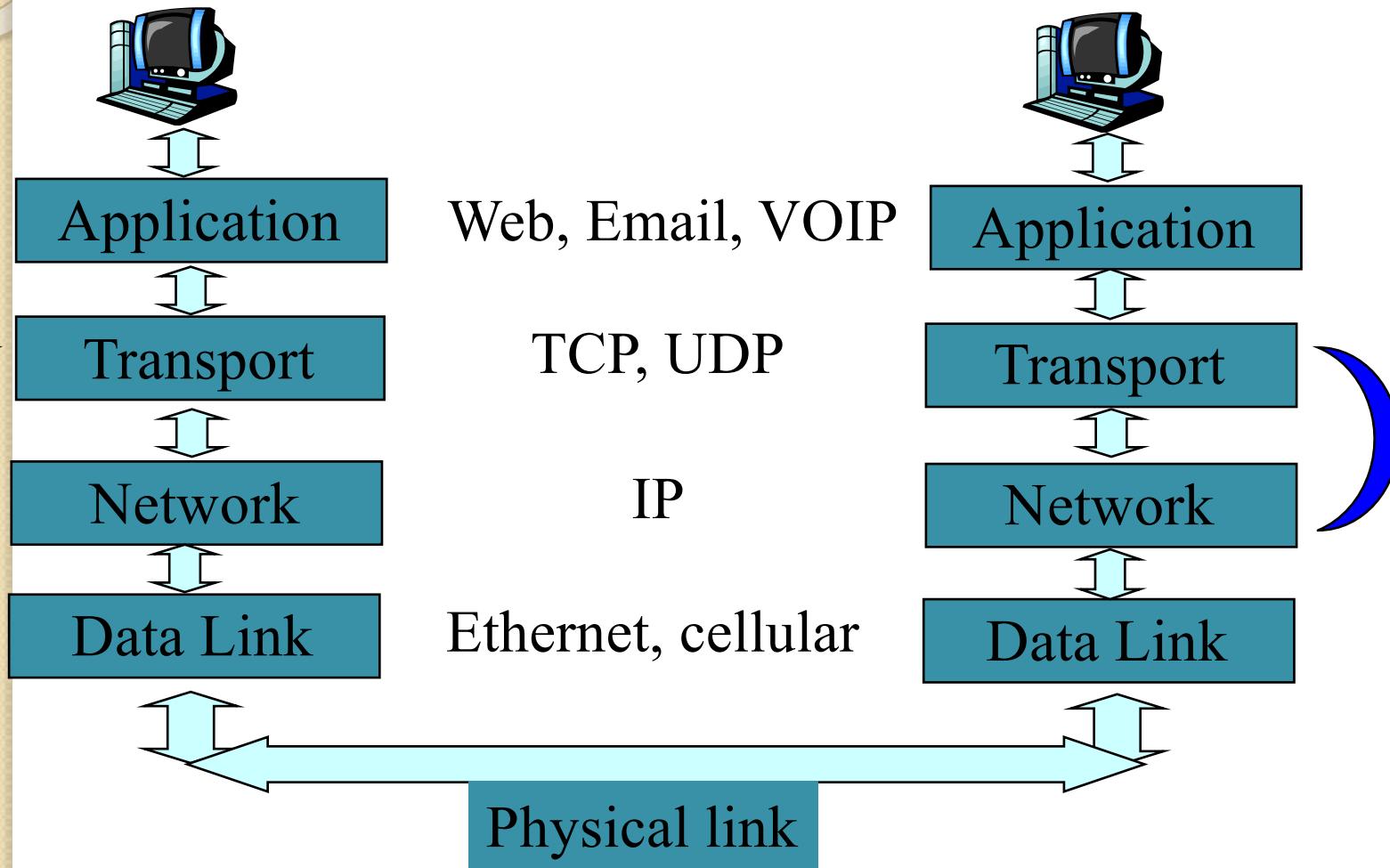
# What is tShark?

- The command-line based packet capture tool
- Equivalent to Wireshark



# Network Layered Structure

- What is the Internet?



# Wireshark Interface

Tucker Ellis & West aaa.pcap - Wireshark

No. Time Source Destination Protocol Info

1	0.000000	192.168.1.2	192.168.1.255	NBNS	Name query NB ECI_DOMAIN<1c>
2	0.746308	192.168.1.2	192.168.1.255	NBNS	Name query NB ECI_DOMAIN<1c>
3	0.751270	192.168.1.2	192.168.1.255	NBNS	Name query NB ECI_DOMAIN<1c>
4	9.318731	silicom_01:6e:bd	Broadcast	ARP	Who has 192.168.1.1? Tell 192.168.1.1 is at 00:30:54:00
5	0.000664	Castlrene_00:34:56	silicom_01:6e:bd	ARP	192.168.1.1 is at 00:30:54:00
6	0.000026	192.168.1.2	192.168.1.1	DNS	Standard query A sip.cybercit
7	0.995383	192.168.1.2	192.168.1.1	DNS	Standard query A sip.cybercit
8	2.003039	192.168.1.2	192.168.1.1	DNS	Standard query A sip.cybercit
9	0.169652	192.168.1.1	192.168.1.2	DNS	Standard query response A 212
10	1.006246	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
11	0.996899	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
12	2.003024	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
13	0.992343	Castlrene_00:34:56	silicom_01:6e:bd	ARP	Who has 192.168.1.2? Tell 192.168.1.2 is at 00:e0:ed:01
14	0.000049	silicom_01:6e:bd	Castlrene_00:34:56	ARP	192.168.1.2 is at 00:e0:ed:01
15	1.010378	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
16	4.005777	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
17	8.002019	192.168.1.2	192.168.1.1	DNS	Standard query PTR 1.0.0.127.
18	0.001489	192.168.1.1	192.168.1.2	DNS	Standard query response PTR 1.0.0.127.
19	0.001640	192.168.1.2	212.242.33.35	SIP	Request: REGISTER sip:sip.cybercit SIP/2.0

Header Length: 20 bytes  
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)  
Total Length: 78  
Identification: 0x698c (27020) Highlighted packets here  
Flags: 0x00  
Fragment offset: 0

0000	ff ff ff ff ff ff	00 e0 ed 01 6e bd	08 00 45 00	..... .n...E.
0010	00 4e 69 8c 00 30	80 11 4c c1 c0 a8	01 02 c0 a8	.N. L.....
0020	01 ff 00 89 00 39	00 39 00 3a 5b b4	84 e7 01 10 00 01	..... [.....
0030	00 00 00 00 00 20	45 46 45 44 45 4a	46 50 45	..... E FEDEJFPE
0040	45 45 50 45 4e 45	42 45 4a 45 4f 43	41 43 41 43 41 43	FEPENEDE JEOCACAC
0050	41 43 41 43 41 42	4d 00 00 20 00 01		ACACABM. . .

Identification (ip.id), 2 bytes  
Packets: 691 Displayed: 691 Marked: 0  
Profile: Default

# Wireshark Interface

command menus

display filter specification

listing of captured packets

details of selected packet header

packet content in hexadecimal and ASCII

The screenshot shows the Wireshark application window with several panels:

- Top Panel:** A menu bar with File, Edit, View, Go, Capture, Analyze, Statistics, Help.
- Second Panel:** A toolbar with various icons for file operations, selection, and analysis.
- Third Panel:** A "Filter:" input field with an arrow pointing to it, followed by Expression..., Clear, and Apply buttons.
- Fourth Panel:** A table titled "No. | Time | Source | Destination | Protocol | Info" listing captured packets. The first few rows show TCP and HTTP traffic between 192.168.1.46 and 128.121.50.122.
- Fifth Panel:** A detailed view of the selected packet (Frame 4). It shows the packet structure with expanded fields like Ethernet II, Internet Protocol, Transmission Control Protocol, and Hypertext Transfer Protocol, along with the raw GET request message.
- Sixth Panel:** A hex/ASCII dump at the bottom of the interface, showing the raw bytes of the selected packet.

# Status Bar

Tucker Ellis & West aaa.pcap - Wireshark

No. Time Source Destination Protocol Info

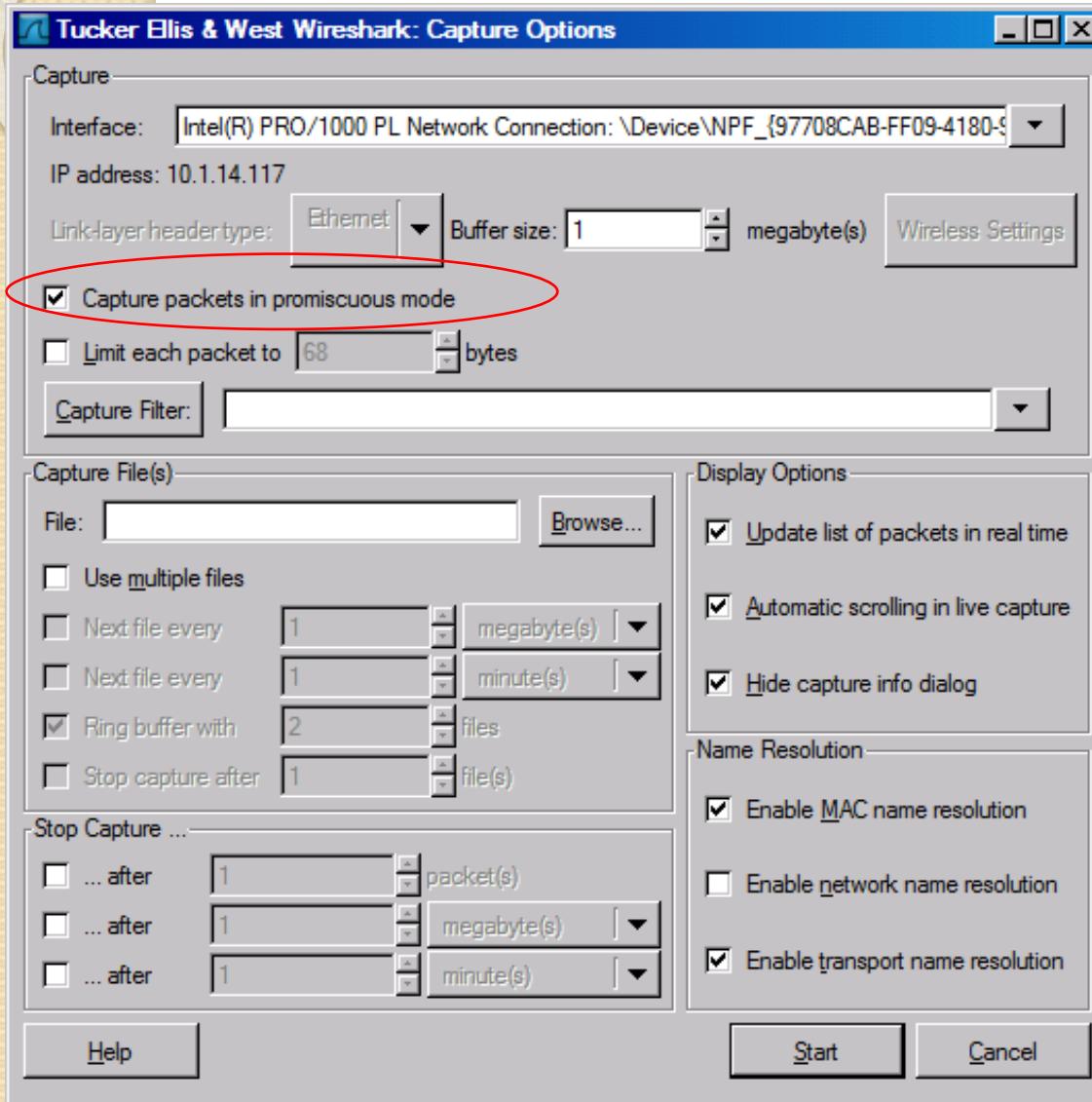
1	0.000000	192.168.1.2	192.168.1.255	NBNS	Name query NB ECI_DOMAIN<1c>
2	0.746308	192.168.1.2	192.168.1.255	NBNS	Name query NB ECI_DOMAIN<1c>
3	0.751270	192.168.1.2	192.168.1.255	NBNS	Name query NB ECI_DOMAIN<1c>
4	9.318731	Silicom_01:6e:bd	Broadcast	ARP	who has 192.168.1.1? Tell 192.168.1.1 is at 00:30:54:00
5	0.000664	Castlene_00:34:56	Silicom_01:6e:bd	ARP	192.168.1.1 is at 00:30:54:00
6	0.000026	192.168.1.2	192.168.1.1	DNS	Standard query A sip.cybercit
7	0.995383	192.168.1.2	192.168.1.1	DNS	Standard query A sip.cybercit
8	2.003039	192.168.1.2	192.168.1.1	DNS	Standard query A sip.cybercit
9	0.169652	192.168.1.1	192.168.1.2	DNS	Standard query response A 212
10	1.006246	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
11	0.996899	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
12	2.003024	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
13	0.992343	Castlene_00:34:56	Silicom_01:6e:bd	ARP	who has 192.168.1.2? Tell 192.168.1.2 is at 00:e0:ed:01
14	0.000049	Silicom_01:6e:bd	Castlene_00:34:56	ARP	192.168.1.2 is at 00:e0:ed:01
15	1.010378	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
16	4.005777	192.168.1.2	192.168.1.1	DNS	Standard query SRV _sip._udp.
17	8.002019	192.168.1.2	192.168.1.1	DNS	Standard query PTR 1.0.0.127.
18	0.001489	192.168.1.1	192.168.1.2	DNS	Standard query response PTR 1.0.0.127.
19	0.001640	192.168.1.2	212.242.33.35	SIP	Request: REGISTER sip:sip.cybercit

Header length: 20 bytes  
+ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)  
Total Length: 78  
Identification: 0x698c (27020)  
+ Flags: 0x00  
Fragment offset: 0

0000 ff ff ff ff ff ff 00 e0 ed 01 6e bd 08 00 45 00 ..n...E.
0010 00 4e 69 8c 00 00 80 11 4c c1 c0 a8 01 02 c0 a8 .Ni.....L.....
0020 01 ff 00 89 00 89 00 3a 5b b4 84 e7 01 10 00 01 .....[.....
0030 00 00 00 00 00 20 45 46 45 44 45 4a 46 50 45 .....E FEDEJFPE
0040 45 45 50 45 4e 45 42 45 4a 45 4f 43 41 43 41 43 EEPENEBE JEOCACAC
0050 41 43 41 43 41 42 4d 00 00 20 00 01 ACACABM. . .

Identification (ip.id), 2 bytes  
Packets: 691 Displayed: 691 Marked: 0  
Profile: Default

# Capture Options

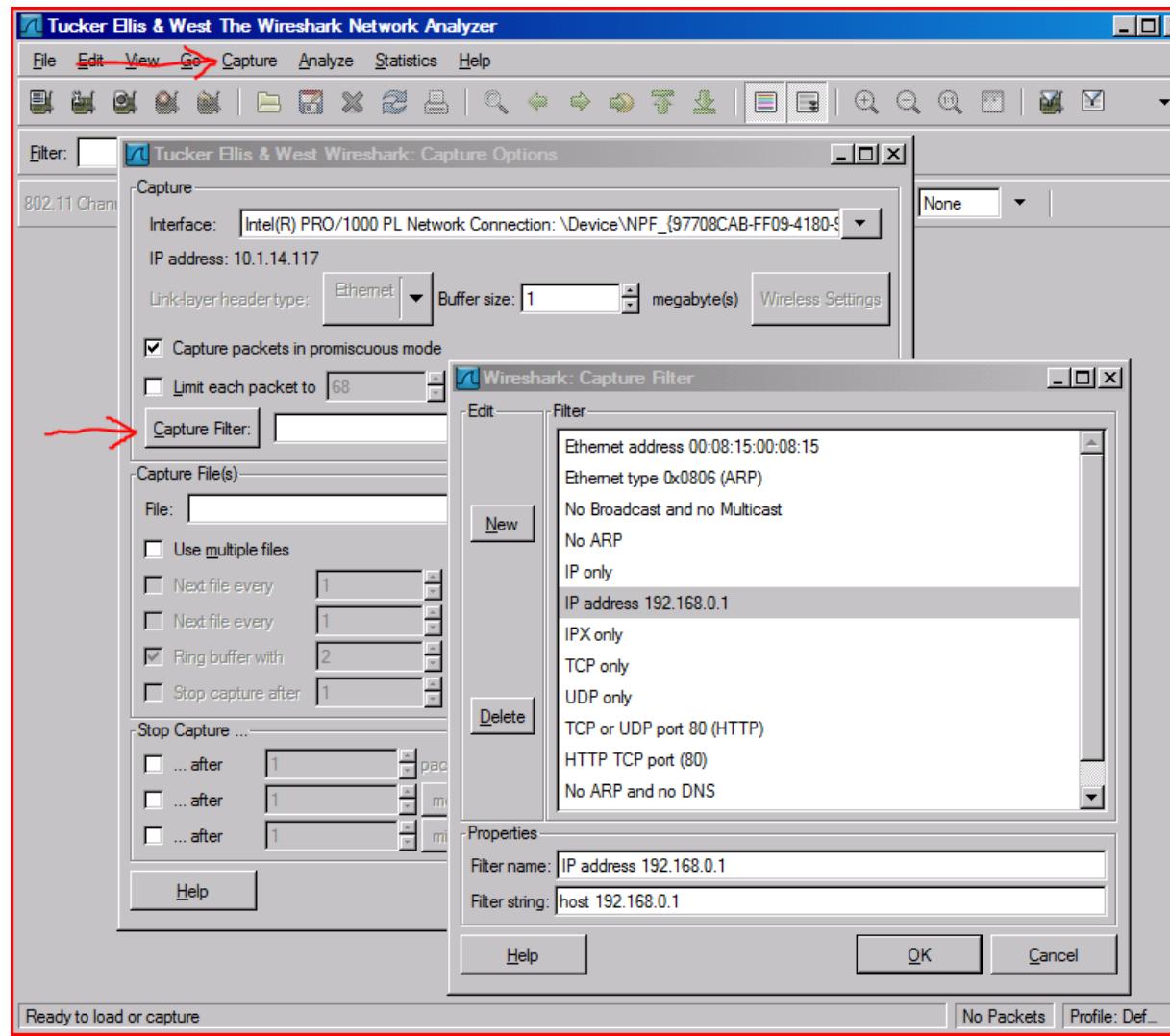


Promiscuous mode is used to  
Capture all traffic

In many cases this does not work:  
• Network driver does not support  
• You are on a switch LAN

# Capture Filter

There are some pre-built capture filters that you can use:





# **Capture Filter examples**

**host 10.1.11.24**

**host 192.168.0.1 and host 10.1.11.1**

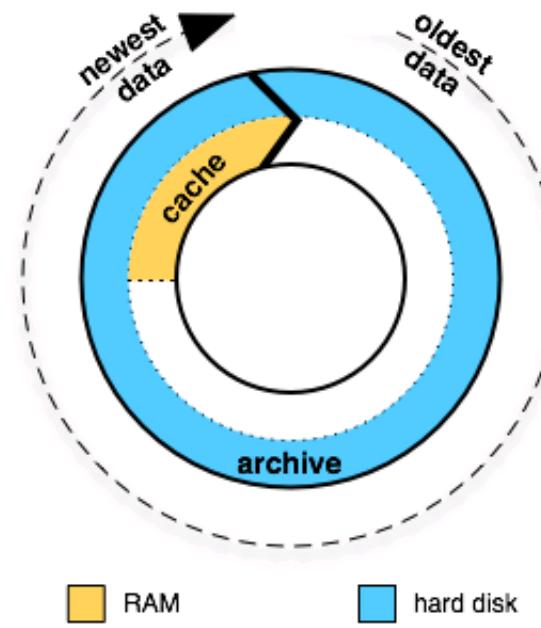
**tcp port http**

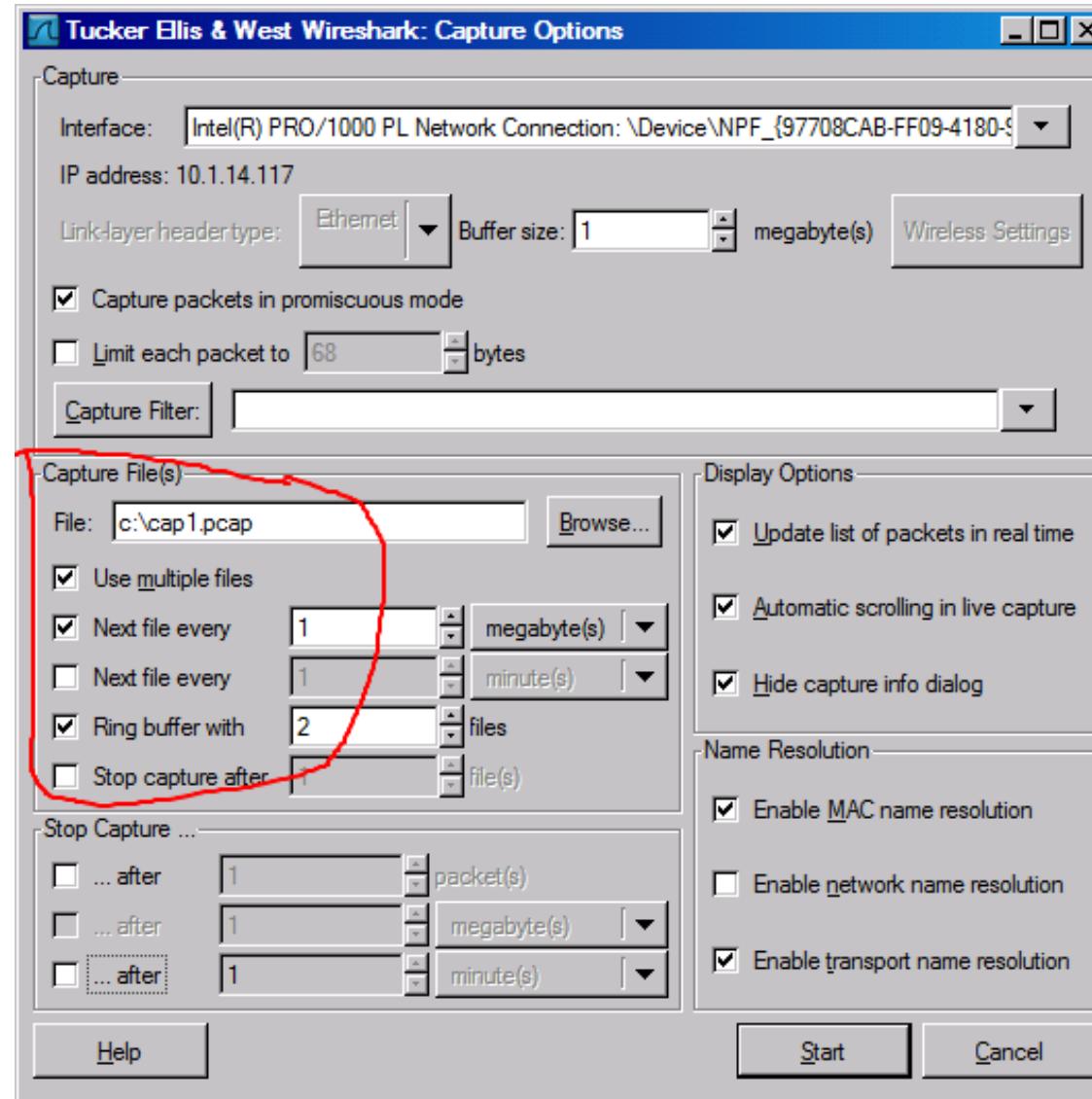
**ip**

**not broadcast not multicast**

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

# Capture Buffer Usage





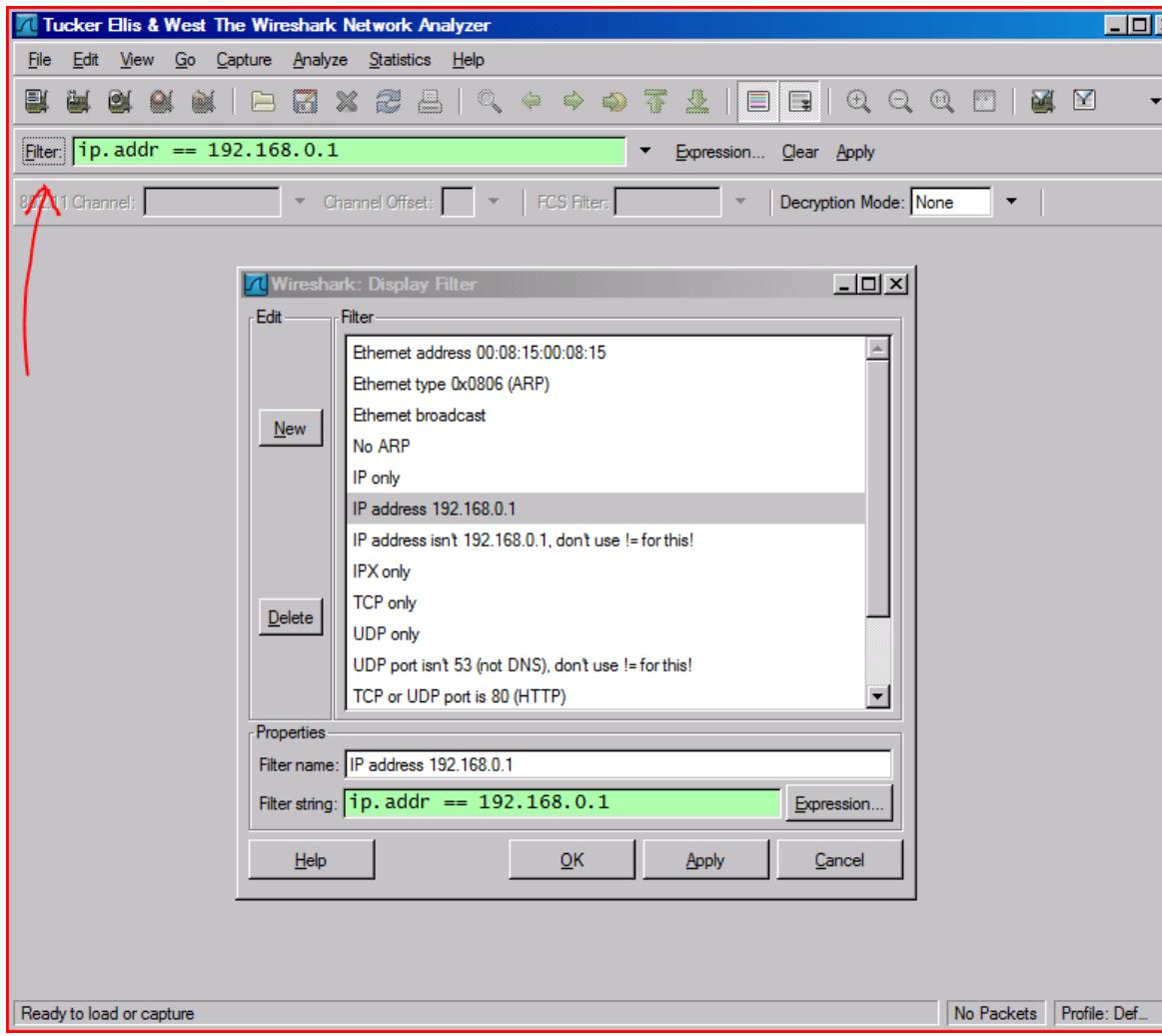


# Display Filters (Post-Filters)

- Display filters (also called post-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 than capture filters

# Display Filter

There are some basic pre-built display filters, too



# Display Filter Examples

`ip.src==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)`

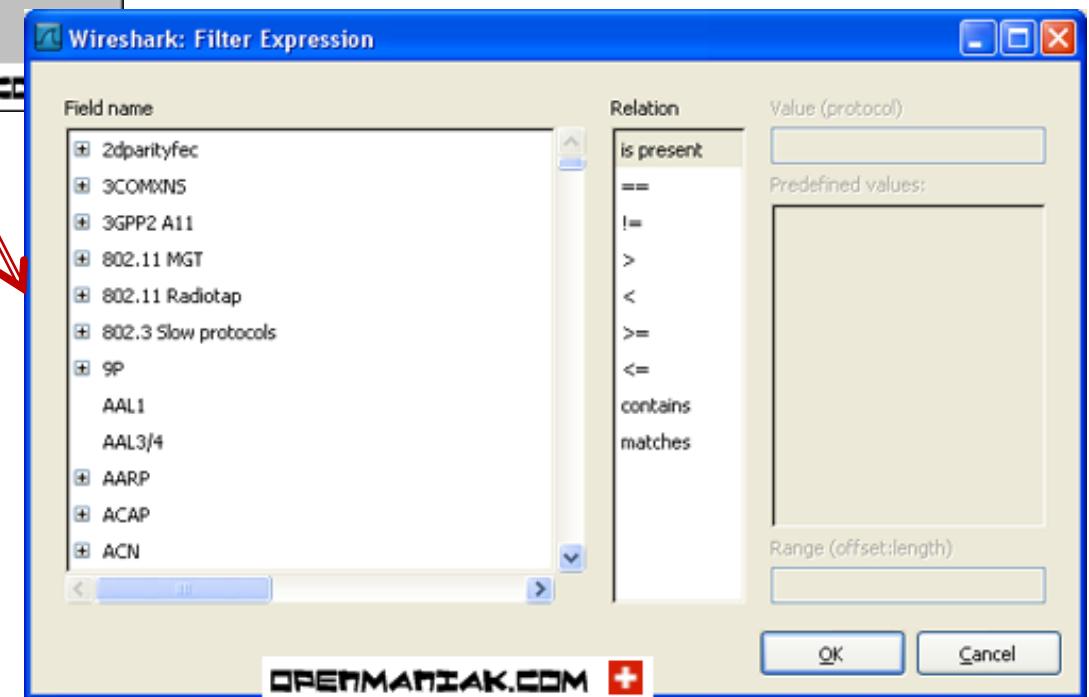
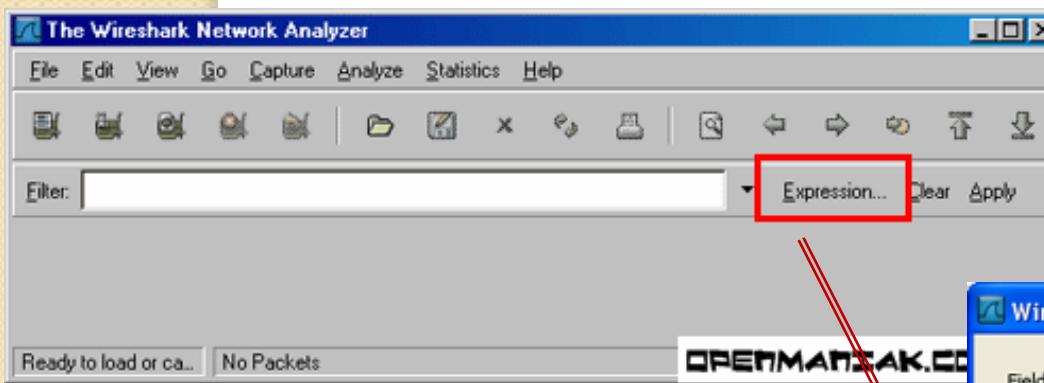
`(ip.addr==192.168.1.10 && ip.addr==192.168.1.20) && (tcp.port==445 ||  
tcp.port==139)`

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

`tcp.dstport == 80`

# 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



There are thousands of pre-defined protocol fields that You can use in the display filter!

# TCP segment structure

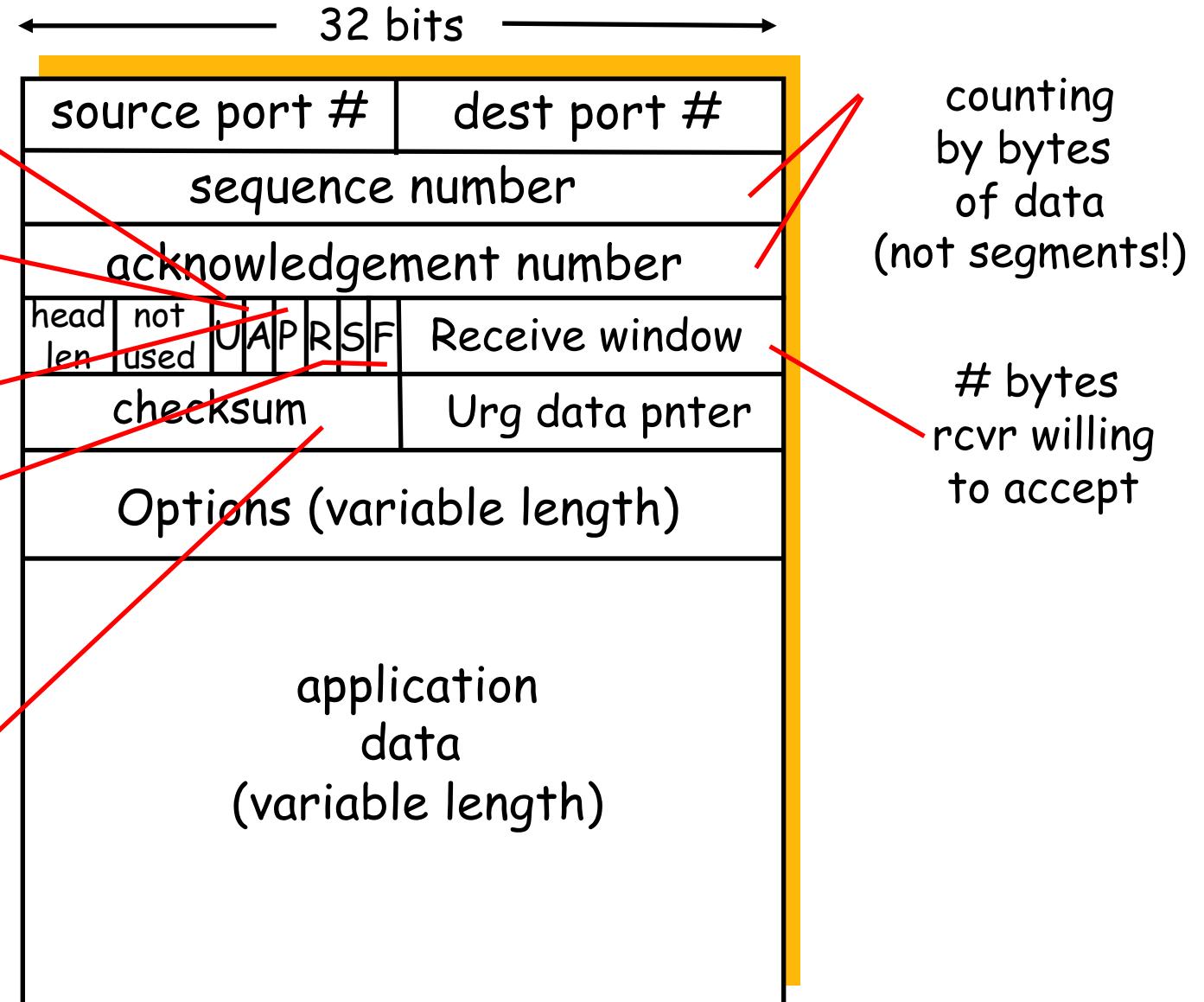
URG: urgent data  
(generally not used)

ACK: ACK #  
valid

PSH: push data now

RST, SYN, FIN:  
connection estab  
(setup, teardown  
commands)

Internet  
checksum  
(as in UDP)



# 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

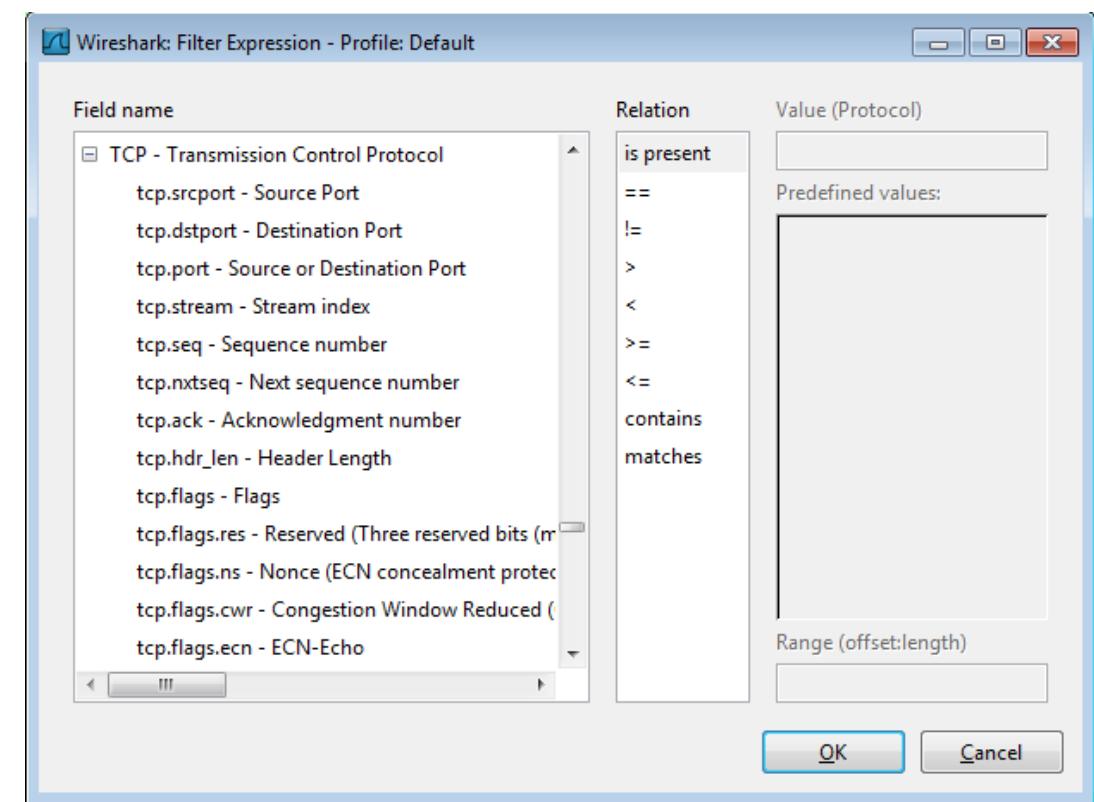
- Or use “**Tcp.flags.syn==1**”

◦ **tcp.flags == 18**

- SYN/ACK

U	A	P	R	S	F
R	C	S	S	Y	I
G	K	H	T	N	N

◦ Note of TCP Flag field:



# Display Filter Expressions

- `snmp || dns || icmp`
  - Display the SNMP or DNS or ICMP traffics.
- `tcp.port == 25`
  - Display packets with TCP source or destination port 25.
- `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.

```
Filter: tcp.port == 100|
```

```
Filter: tcp.port = 100|
```

Correct syntax

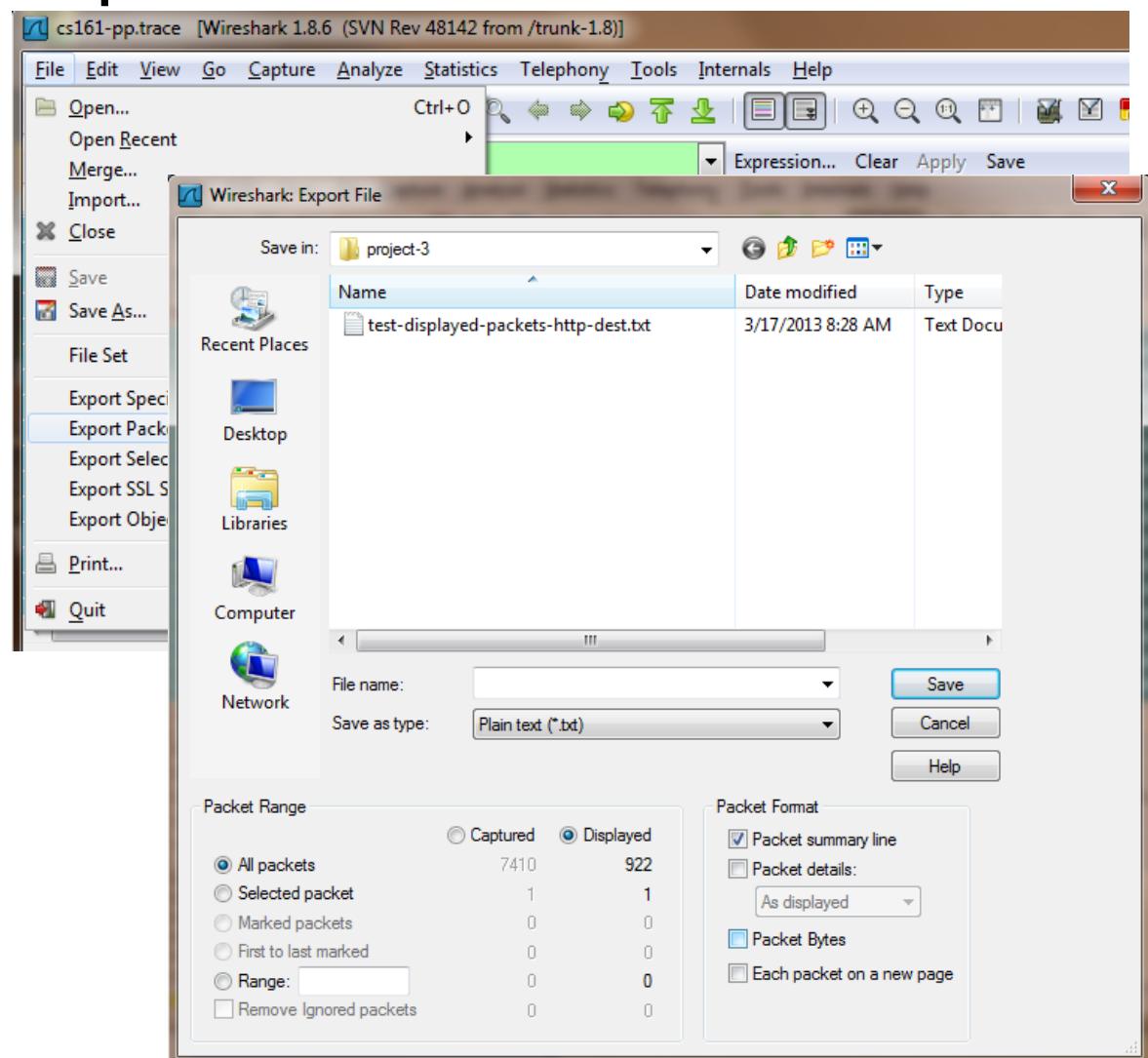
Wrong syntax

# Save Filtered Packets as Text After Using Display Filter

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

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

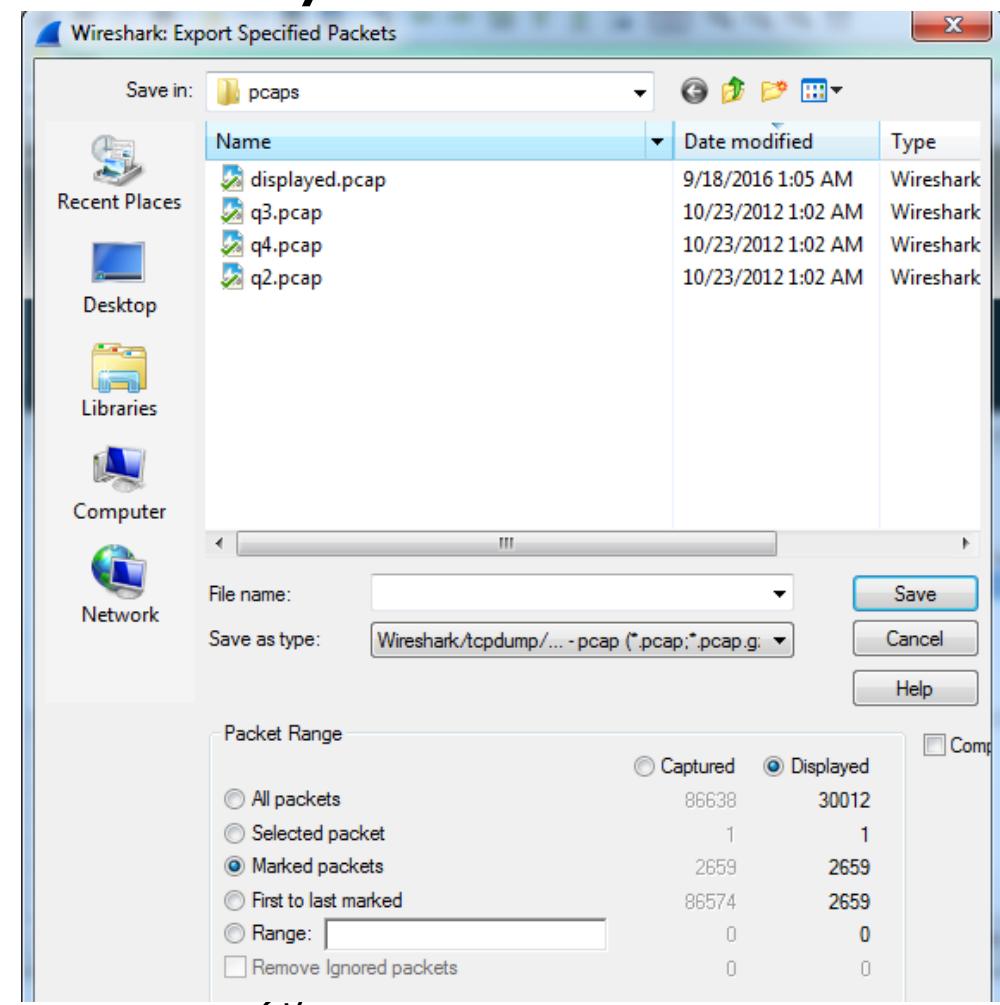
- 1). In “packet range” option, select “Displayed”
- 2). In choose “summary line” or “detail”



# Save Filtered Packets in Wireshark format After Using Display Filter

- We can also save all filtered packets in the original wireshark format for further analysis
- Operation:

1. Enter Display filter to show packets you want
2. Go to "Edit>" and choose "Mark all displayed packets"
3. Go to "File" → Export specific packets...
4. Choose the option "Marked packets" to save the file



# Protocol Hierarchy

Tucker Ellis & West Obsolete\_Packets.cap - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: [ ]

802.11 Channel: [ ]

No.	Time	Source
1	0.000000	::
2	0.000010	::
3	2.179063	192.168.1.1
4	2.439522	192.168.1.1
5	2.715733	192.168.1.1
6	2.821401	192.168.1.1
7	2.821546	192.168.1.1
8	2.824683	192.168.1.1
9	2.990859	192.168.1.1
10	3.266913	192.168.1.1
11	3.495707	fe80::20c:9ff%1
12	3.495727	fe80::20c:9ff%1
13	3.542893	192.168.1.1
14	3.543088	192.168.1.1

Protocol Hierarchy

- Summary
- Protocol Hierarchy
- Conversations
- Endpoints
- I/O Graphs

Expression... Clear Apply

Decryption Mode: None

	Protocol	Info
0d:56e3	ICMPv6	Multicast listener report
0d:56e3	ICMPv6	Multicast listener report
255	NBNS	Name query NB LOCALHOST
255	NBNS	Name query NB LOCALHOST
255	NBNS	Name query NB LOCALHOST
254	DNS	Standard query PTR 66.1
254	DNS	Standard query PTR 255.
66	DNS	Standard query response
03.255	NBNS	Name query NB LOCALHOST
03.255	NBNS	Name query NB LOCALHOST
	ICMPv6	Router solicitation
	ICMPv6	Router solicitation
254	DNS	Standard query A DoCoMo
03.255	NBNS	Name query NB LOCALHOST

Frame 1 (88 bytes on wire, 88 bytes captured)  
+ Linux cooked capture  
+ Internet Protocol Version 4, Src: 192.168.1.1 (192.168.1.1), Dst: 255.255.255.255 (255.255.255.255)  
+ Internet Control Message Protocol (ICMPv6)

BOOTP-DHCP... Destinations... Flow Graph... HTTP IP address... ISUP Messages... Multicast Streams ONC-RPC Programs Packet Length... Port Type... SMPP Operations... TCP Stream Graph WLAN Traffic...

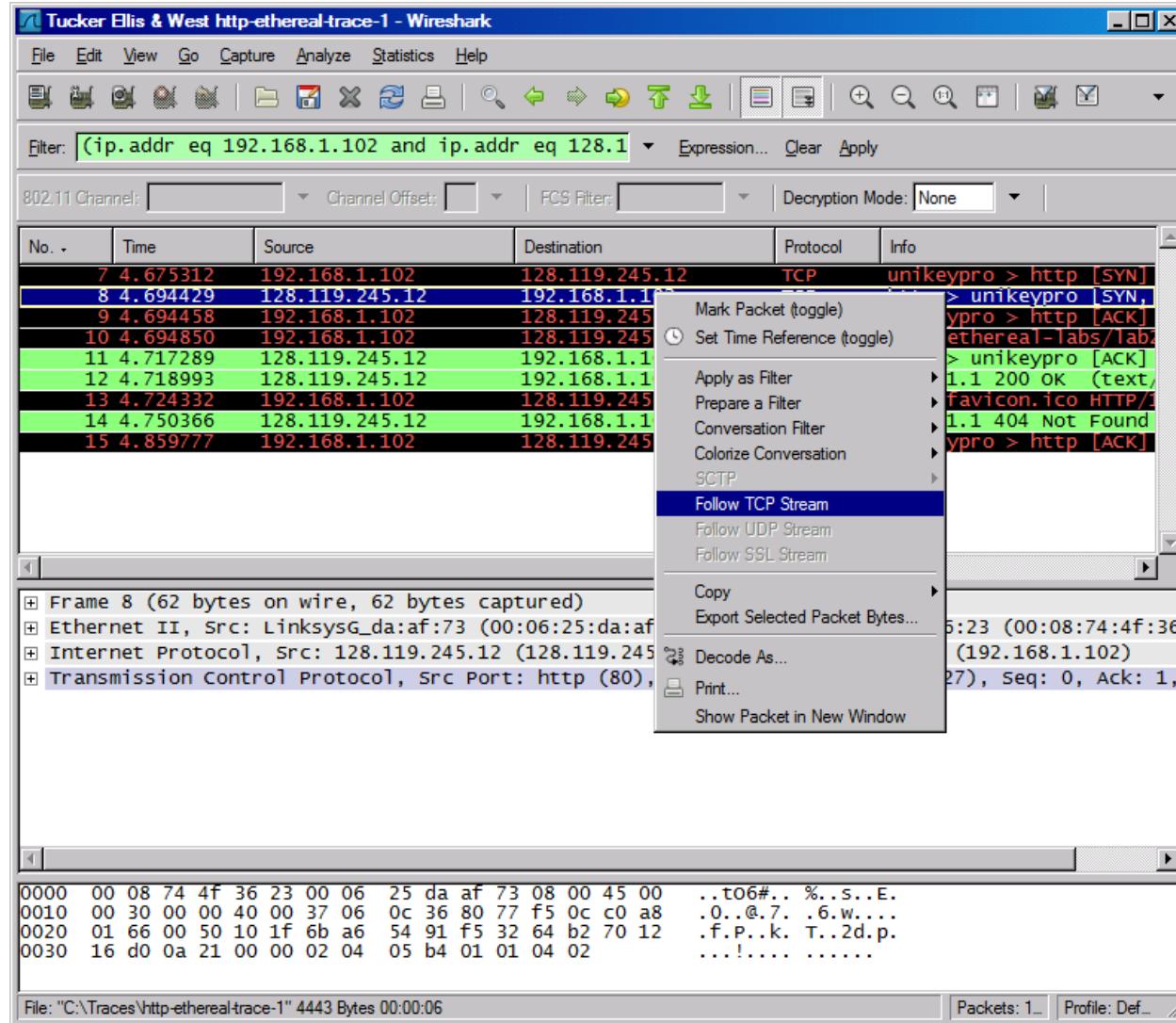
dd :.....).V.....  
00 ..... . ....  
00 .....V. ....  
00 .....V. ....

File: "C:\Users\vo2.TEW\Downloads\Obsolete\_Packets.cap..." Packets: 10949 Displayed: 10949 Marked: 0 Profile: Default

# Protocol Hierarchy

Wireshark: Protocol Hierarchy Statistics								
Display filter: none								
Protocol	% Packets	packets	bytes	Mbit/s	End Packets	End Bytes	End Mbit/s	
Frame	100.00%	10949	1433310	0.004	0	0	0.000	
Linux cooked-mode capture	100.00%	10949	1433310	0.004	0	0	0.000	
Internet Protocol Version 6	0.16%	18	1392	0.000	0	0	0.000	
Internet Control Message Protocol v6	0.16%	18	1392	0.000	18	1392	0.000	
Internet Protocol	82.62%	9046	1312691	0.004	0	0	0.000	
User Datagram Protocol	17.33%	1898	262866	0.001	0	0	0.000	
Transmission Control Protocol	64.69%	7083	1046121	0.003	2350	163598	0.000	
Internet Group Management Protocol	0.57%	62	3440	0.000	62	3440	0.000	
Internet Control Message Protocol	0.03%	3	264	0.000	3	264	0.000	
DEC DNA Routing Protocol	2.60%	285	14820	0.000	285	14820	0.000	
Address Resolution Protocol	7.63%	835	46928	0.000	835	46928	0.000	
MS Network Load Balancing	1.26%	138	8280	0.000	138	8280	0.000	
Data	2.75%	301	25143	0.000	301	25143	0.000	
Logical-Link Control	2.23%	244	20024	0.000	0	0	0.000	
Appletalk Address Resolution Protocol	0.37%	40	2480	0.000	40	2480	0.000	
Internet Protocol eXchange	1.46%	160	14328	0.000	0	0	0.000	
Datagram Delivery Protocol	0.40%	44	3216	0.000	0	0	0.000	
Internet Protocol eXchange	0.27%	30	1680	0.000	0	0	0.000	
Banyan Vines IP	0.47%	52	2352	0.000	0	0	0.000	

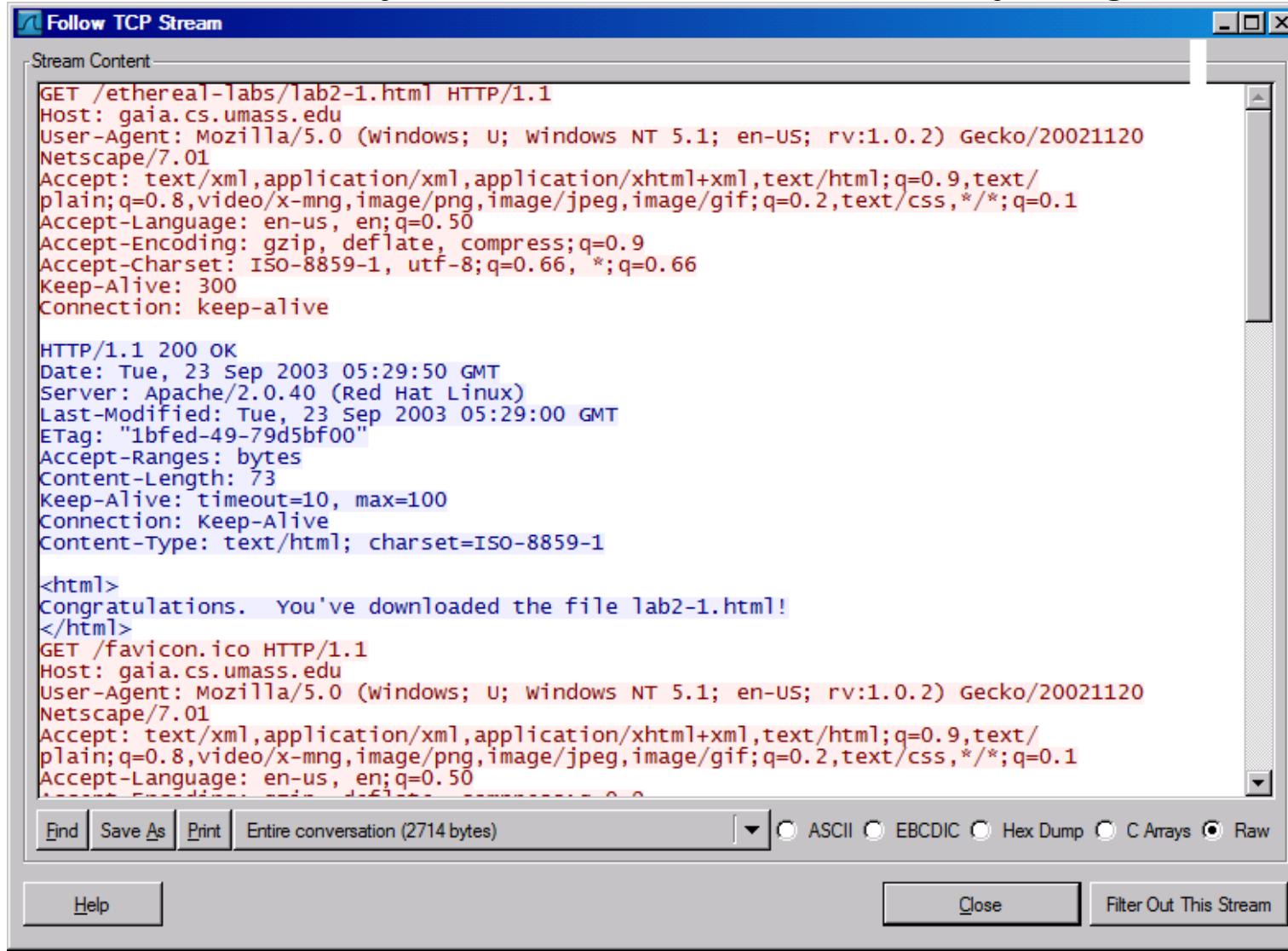
# Follow TCP Stream



# Follow TCP Stream

red - stuff you sent

blue - stuff you get



The screenshot shows a window titled "Follow TCP Stream" displaying a network conversation. The "Stream Content" pane shows the following text:

```
GET /ethereal-tabs/Tab2-1.html HTTP/1.1
Host: gaia.cs.umass.edu
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.2) Gecko/20021120
Netscape/7.01
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,video/x-mng,image/png,image/jpeg,image/gif;q=0.2,text/css,*/*;q=0.1
Accept-Language: en-us, en;q=0.50
Accept-Encoding: gzip, deflate, compress;q=0.9
Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66
Keep-Alive: 300
Connection: keep-alive

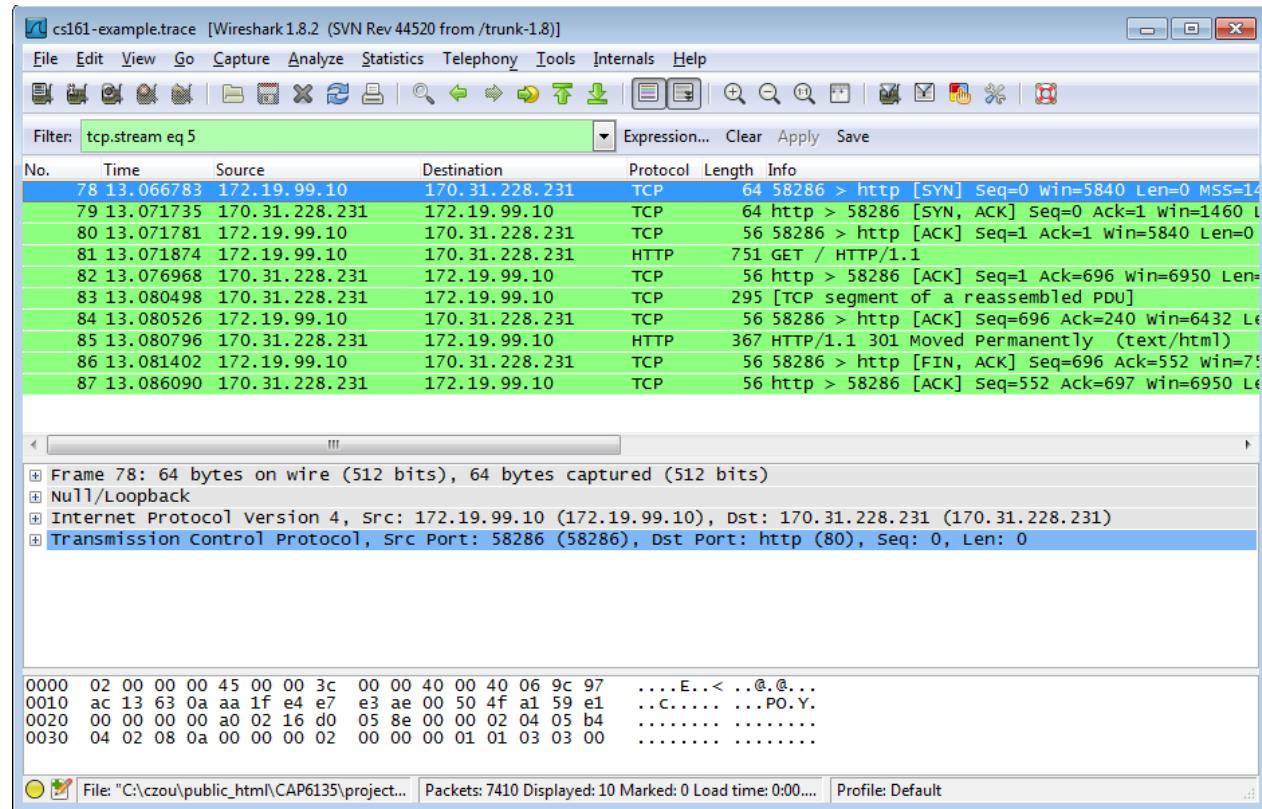
HTTP/1.1 200 OK
Date: Tue, 23 Sep 2003 05:29:50 GMT
Server: Apache/2.0.40 (Red Hat Linux)
Last-Modified: Tue, 23 Sep 2003 05:29:00 GMT
ETag: "1bfed-49-79d5bf00"
Accept-Ranges: bytes
Content-Length: 73
Keep-Alive: timeout=10, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=ISO-8859-1

<html>
Congratulations. You've downloaded the file tab2-1.html!
</html>
GET /favicon.ico HTTP/1.1
Host: gaia.cs.umass.edu
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.2) Gecko/20021120
Netscape/7.01
Accept: text/xml,application/xml,application/xhtml+xml;text/html;q=0.9;text/
plain;q=0.8,video/x-mng,image/png,image/jpeg,image/gif;q=0.2;text/css,*/*;q=0.1
Accept-Language: en-us, en;q=0.50
```

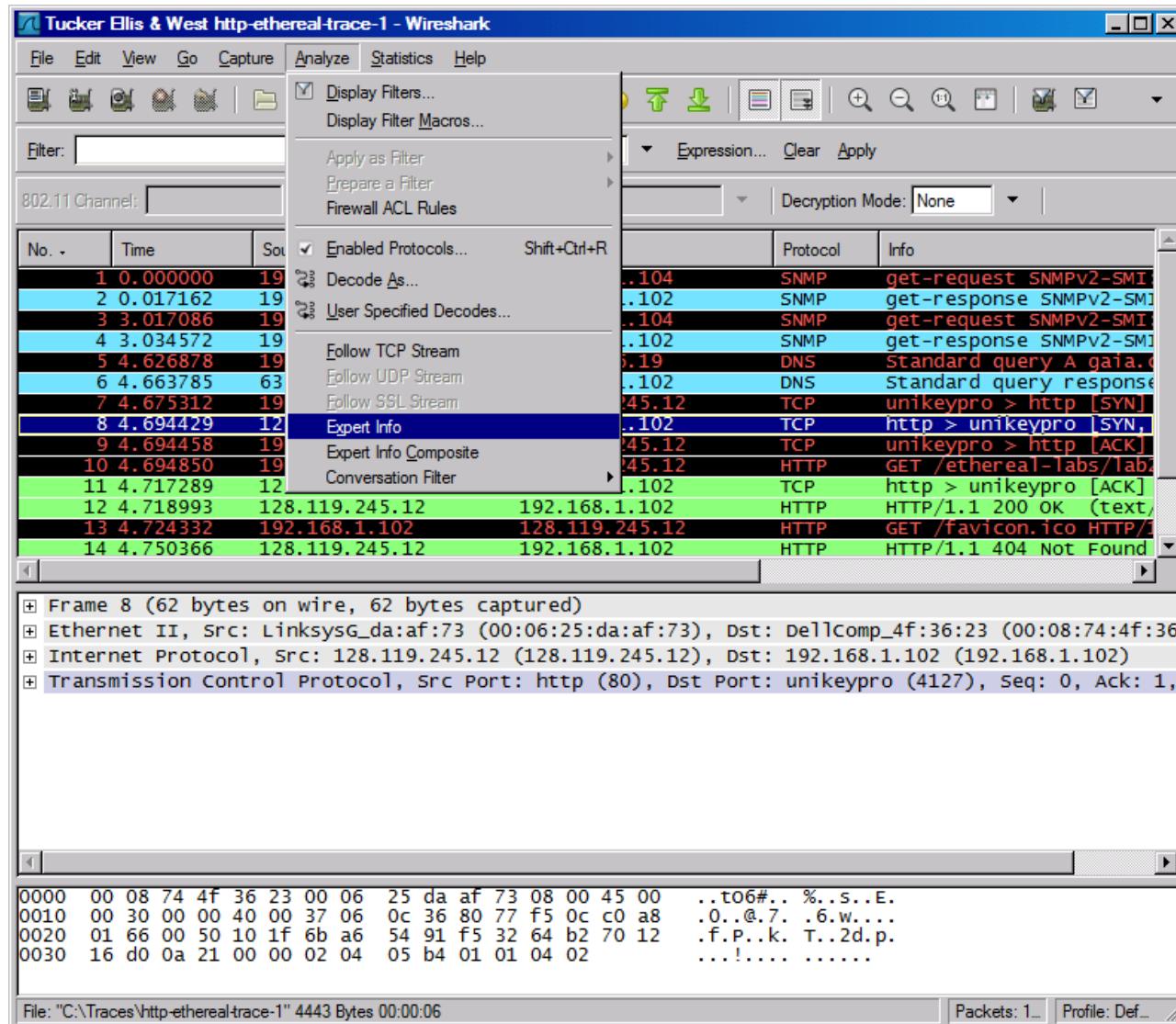
At the bottom, there are buttons for "Find", "Save As", "Print", and "Entire conversation (2714 bytes)". There are also radio buttons for "ASCII", "EBCDIC", "Hex Dump", "C Arrays", and "Raw", with "Raw" selected. Other buttons include "Help", "Close", and "Filter Out This Stream".

# 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



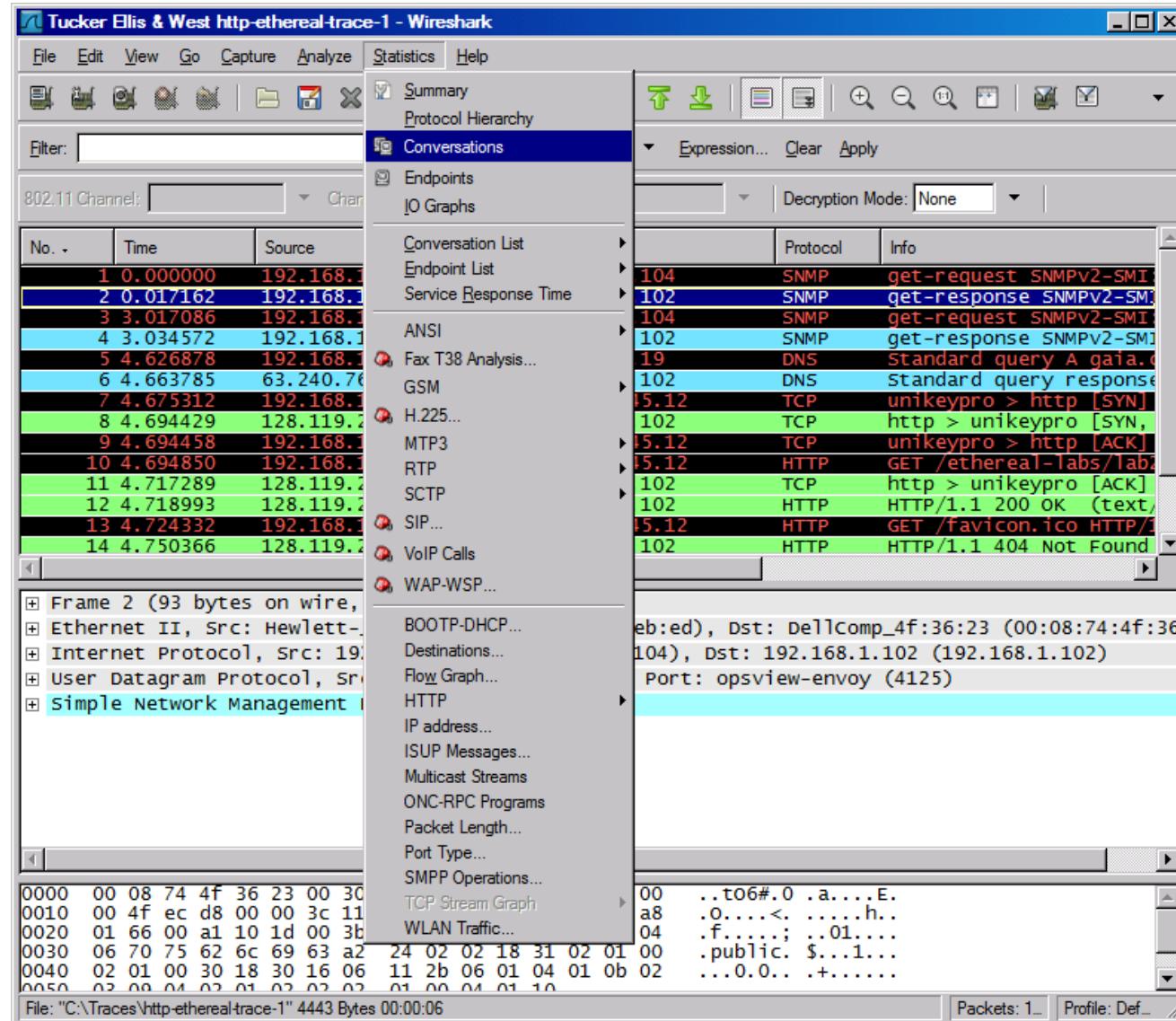
# Expert Info

Wireshark: 1527 Expert Infos

Group	Protocol	Summary	Count
+ Sequence	IPv4	"Time To Live" only 1	380
+ Sequence	TCP	Duplicate ACK (#1)	16
+ Malformed	HTTP	HTTP body subdissector failed, trying heuristic subdissector	8
+ Sequence	TCP	Retransmission (suspected)	4
+ Sequence	TCP	Duplicate ACK (#2)	1
+ Sequence	IPv4	"Time To Live" only 2	30
+ Sequence	IPv4	"Time To Live" only 3	30
+ Sequence	IPv4	"Time To Live" only 4	30

[Help](#) [Close](#)

# Conversations



# Conversations

Conversations: cs161-pp.trace

Ethernet | Fibre Channel | FDDI | IPv4: 173 | IPv6: 1 | IPX | JXTA | NCP | RSVP | SCTP | TCP: 155 | Token Ring | UDP: 2398 | USB | WLAN

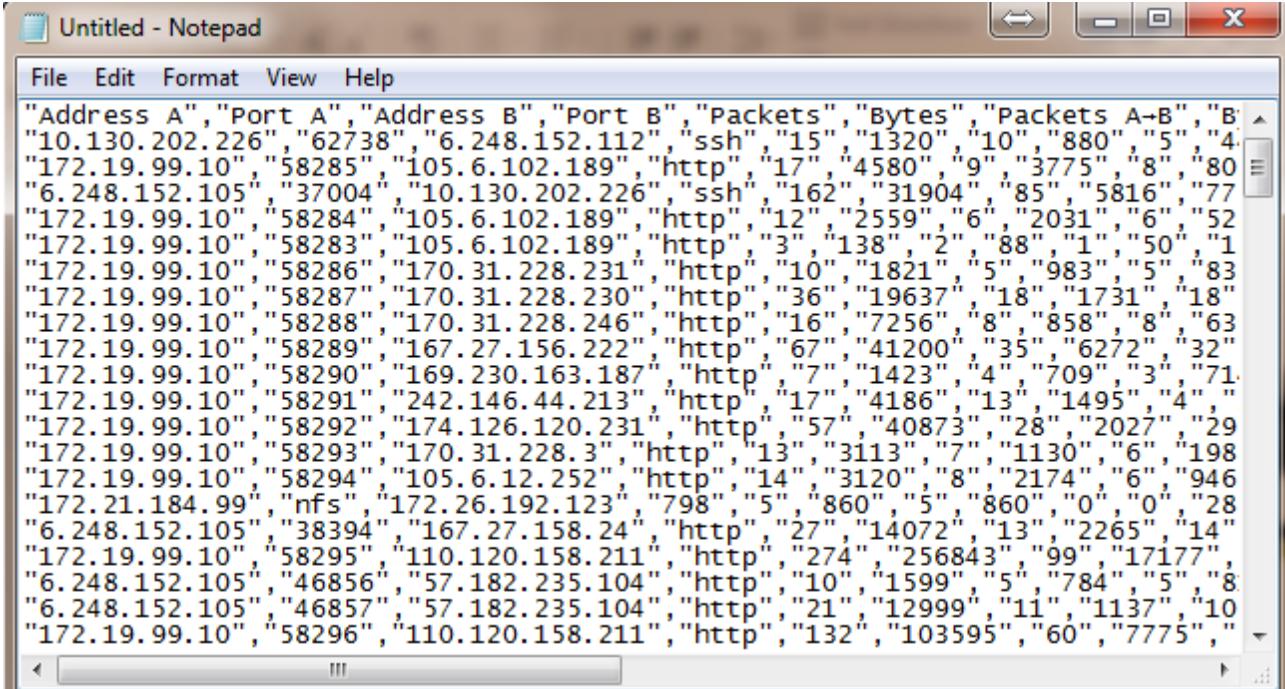
TCP Conversations

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A→B	Bytes A→B	Packets A←B	Bytes A←B
10.130.202.226	62738	6.248.152.112	ssh	15	1 320	10	880	5	5
172.19.99.10	58285	105.6.102.189	http	17	4 580	9	3 775	8	8
6.248.152.105	37004	10.130.202.226	ssh	162	31 904	85	5 816	77	77
172.19.99.10	58284	105.6.102.189	http	12	2 559	6	2 031	6	6
172.19.99.10	58283	105.6.102.189	http	3	138	2	88	1	1
172.19.99.10	58286	170.31.228.231	http	10	1 821	5	983	5	5
172.19.99.10	58287	170.31.228.230	http	36	19 637	18	1 731	18	18
172.19.99.10	58288	170.31.228.246	http	16	7 256	8	858	8	8
172.19.99.10	58289	167.27.156.222	http	67	41 200	35	6 272	32	32
172.19.99.10	58290	169.230.163.187	http	7	1 423	4	709	3	3
172.19.99.10	58291	242.146.44.213	http	17	4 186	13	1 495	4	4
172.19.99.10	58292	174.126.120.231	http	57	40 873	28	2 027	29	29

Name resolution     Limit to display filter

Help Copy Follow Stream Close

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



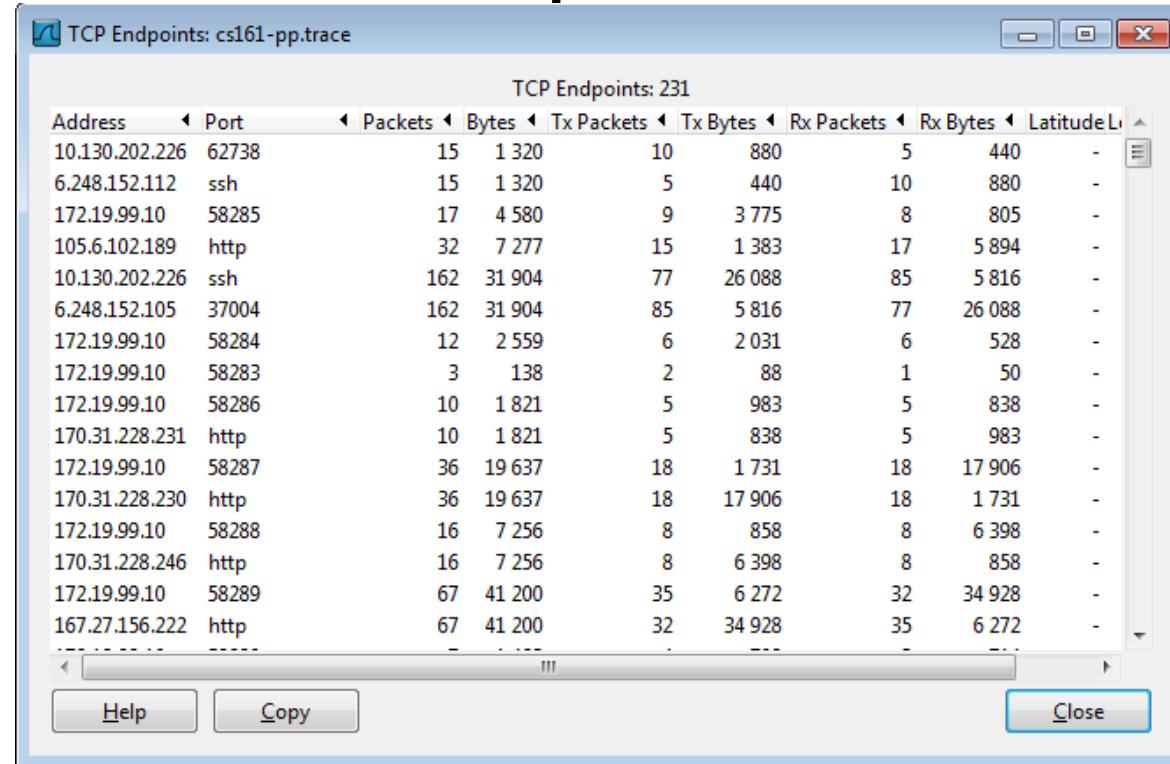
The screenshot shows a Windows Notepad window with the title "Untitled - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main text area contains a large block of log data. The data consists of numerous lines, each representing a log entry with fields separated by commas. Some fields contain numerical values like port numbers or byte counts, while others are descriptive strings like "ssh" or "http". The log entries appear to be network traffic statistics.

```
"Address A", "Port A", "Address B", "Port B", "Packets", "Bytes", "Packets A→B", "B→A",  
"10.130.202.226", "62738", "6.248.152.112", "ssh", "15", "1320", "10", "880", "5", "4",  
"172.19.99.10", "58285", "105.6.102.189", "http", "17", "4580", "9", "3775", "8", "80",  
"6.248.152.105", "37004", "10.130.202.226", "ssh", "162", "31904", "85", "5816", "77",  
"172.19.99.10", "58284", "105.6.102.189", "http", "12", "2559", "6", "2031", "6", "52",  
"172.19.99.10", "58283", "105.6.102.189", "http", "3", "138", "2", "88", "1", "50", "1",  
"172.19.99.10", "58286", "170.31.228.231", "http", "10", "1821", "5", "983", "5", "83",  
"172.19.99.10", "58287", "170.31.228.230", "http", "36", "19637", "18", "1731", "18",  
"172.19.99.10", "58288", "170.31.228.246", "http", "16", "7256", "8", "858", "8", "63",  
"172.19.99.10", "58289", "167.27.156.222", "http", "67", "41200", "35", "6272", "32",  
"172.19.99.10", "58290", "169.230.163.187", "http", "7", "1423", "4", "709", "3", "71",  
"172.19.99.10", "58291", "242.146.44.213", "http", "17", "4186", "13", "1495", "4", "  
",  
"172.19.99.10", "58292", "174.126.120.231", "http", "57", "40873", "28", "2027", "29",  
"172.19.99.10", "58293", "170.31.228.3", "http", "13", "3113", "7", "1130", "6", "198",  
"172.19.99.10", "58294", "105.6.12.252", "http", "14", "3120", "8", "2174", "6", "946",  
"172.21.184.99", "nfs", "172.26.192.123", "798", "5", "860", "5", "860", "0", "0", "28",  
"6.248.152.105", "38394", "167.27.158.24", "http", "27", "14072", "13", "2265", "14",  
"172.19.99.10", "58295", "110.120.158.211", "http", "274", "256843", "99", "17177",  
"6.248.152.105", "46856", "57.182.235.104", "http", "10", "1599", "5", "784", "5", "8",  
"6.248.152.105", "46857", "57.182.235.104", "http", "21", "12999", "11", "1137", "10",  
"172.19.99.10", "58296", "110.120.158.211", "http", "132", "103595", "60", "7775", "
```

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

# Find EndPoint Statistics

- Menu “statistics” → “endpoint list” → “TCP”



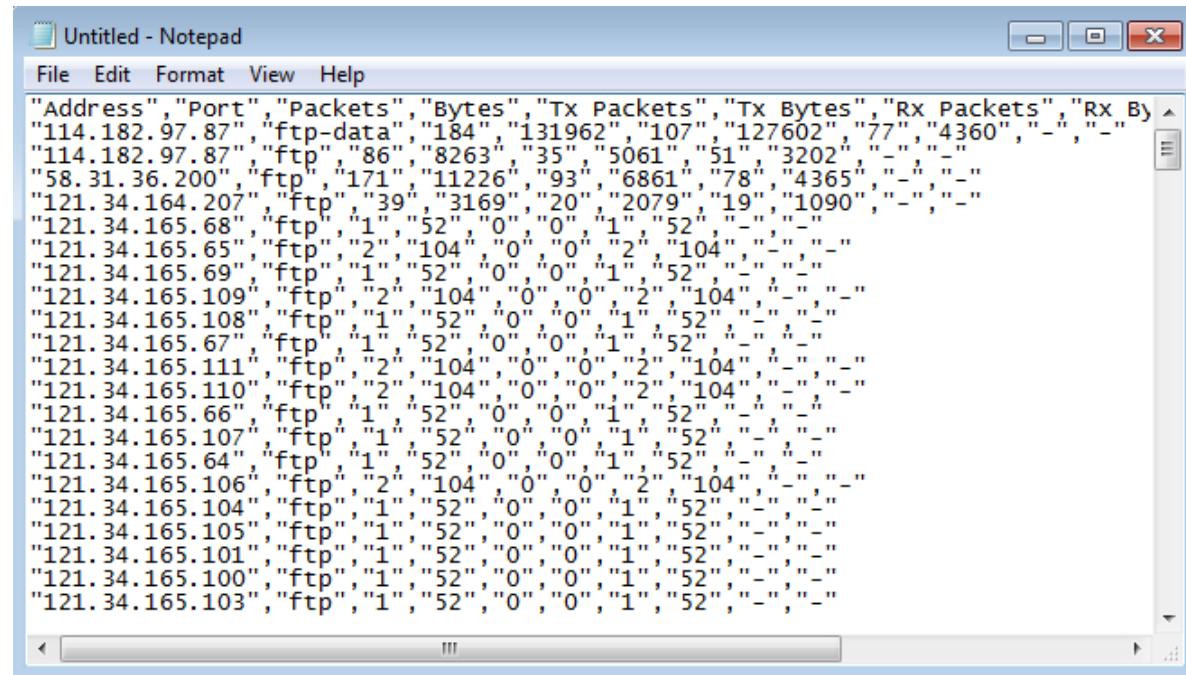
The screenshot shows a Windows-style dialog box titled "TCP Endpoints: cs161-pp.trace". The title bar also displays "TCP Endpoints: 231". The main area is a table with 231 rows, each representing a TCP endpoint. The columns are labeled: Address, Port, Packets, Bytes, Tx Packets, Tx Bytes, Rx Packets, Rx Bytes, Latitude, and Longitude. The table lists various IP addresses and ports, along with their corresponding network statistics. At the bottom of the dialog are buttons for "Help", "Copy", and "Close".

Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Latitude	Longitude
10.130.202.226	62738	15	1 320	10	880	5	440	-	-
6.248.152.112	ssh	15	1 320	5	440	10	880	-	-
172.19.99.10	58285	17	4 580	9	3 775	8	805	-	-
105.6.102.189	http	32	7 277	15	1 383	17	5 894	-	-
10.130.202.226	ssh	162	31 904	77	26 088	85	5 816	-	-
6.248.152.105	37004	162	31 904	85	5 816	77	26 088	-	-
172.19.99.10	58284	12	2 559	6	2 031	6	528	-	-
172.19.99.10	58283	3	138	2	88	1	50	-	-
172.19.99.10	58286	10	1 821	5	983	5	838	-	-
170.31.228.231	http	10	1 821	5	838	5	983	-	-
172.19.99.10	58287	36	19 637	18	1 731	18	17 906	-	-
170.31.228.230	http	36	19 637	18	17 906	18	1 731	-	-
172.19.99.10	58288	16	7 256	8	858	8	6 398	-	-
170.31.228.246	http	16	7 256	8	6 398	8	858	-	-
172.19.99.10	58289	67	41 200	35	6 272	32	34 928	-	-
167.27.156.222	http	67	41 200	32	34 928	35	6 272	-	-

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

# Find EndPoint Statistics

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



The screenshot shows a Windows Notepad window with the title "Untitled - Notepad". The window contains a large amount of text, which appears to be a log or configuration file. The text is organized into several columns of data, likely representing network endpoint statistics. The columns include "Address", "Port", "Packets", "Bytes", "Tx Packets", "Tx Bytes", "Rx Packets", and "Rx Bytes". The data is repeated multiple times for different addresses and ports, such as "114.182.97.87", "121.34.165.68", and "121.34.165.103". Each entry consists of eight fields separated by commas.

```
"Address", "Port", "Packets", "Bytes", "Tx Packets", "Tx Bytes", "Rx Packets", "Rx Bytes"
"114.182.97.87", "ftp-data", "184", "131962", "107", "127602", "77", "4360", "-", "-"
"114.182.97.87", "ftp", "86", "8263", "35", "5061", "51", "3202", "-", "-"
"58.31.36.200", "ftp", "171", "11226", "93", "6861", "78", "4365", "-", "-"
"121.34.164.207", "ftp", "39", "3169", "20", "2079", "19", "1090", "-", "-"
"121.34.165.68", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.65", "ftp", "2", "104", "0", "0", "2", "104", "-", "-"
"121.34.165.69", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.109", "ftp", "2", "104", "0", "0", "2", "104", "-", "-"
"121.34.165.108", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.67", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.111", "ftp", "2", "104", "0", "0", "2", "104", "-", "-"
"121.34.165.110", "ftp", "2", "104", "0", "0", "2", "104", "-", "-"
"121.34.165.66", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.107", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.64", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.106", "ftp", "2", "104", "0", "0", "2", "104", "-", "-"
"121.34.165.104", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.105", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.101", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.100", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
"121.34.165.103", "ftp", "1", "52", "0", "0", "1", "52", "-", "-"
```

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

# Export HTTP

Tucker Elis & West http-ethereal-trace-1 - Wireshark

File Edit View Go Capture Analyze Statistics Help

Open... Ctrl+O  
Open Recent  
Merge...  
Close Ctrl+W  
Save Ctrl+S  
Save As... Shift+Ctrl+S  
File Set  
Export File... Destination Protocol Info  
168.1.102 192.168.1.104 SNMP get-request SNMPv2-SMI  
168.1.104 192.168.1.102 SNMP get-response SNMPv2-SMI  
168.1.102 192.168.1.104 SNMP get-request SNMPv2-SMI  
168.1.104 192.168.1.102 SNMP get-response SNMPv2-SMI  
168.1.102 192.168.1.102 DNS Standard query A gaia.cs.uma.edu  
168.1.102 128.119.245.12 TCP unikeypro > http [SYN]  
119.245.12 192.168.1.102 TCP http > unikeypro [SYN]  
9 4.694458 192.168.1.102 128.119.245.12 TCP unikeypro > http [ACK]  
10 4.694850 192.168.1.102 128.119.245.12 HTTP GET /ethereal-Tabs/Tab1.html  
11 4.717289 128.119.245.12 192.168.1.102 TCP http > unikeypro [ACK]  
12 4.718993 128.119.245.12 192.168.1.102 HTTP HTTP/1.1 200 OK (text)  
13 4.724332 192.168.1.102 128.119.245.12 HTTP GET /favicon.ico HTTP/1.1  
14 4.750366 128.119.245.12 192.168.1.102 HTTP HTTP/1.1 404 Not Found

Source port: unikeypro (4127)  
Destination port: http (80)  
Sequence number: 1 (relative sequence number)  
[Next sequence number: 502 (relative sequence number)]  
Acknowledgement number: 1 (relative ack number)  
Header length: 20 bytes  
Flags: 0x18 (PSH, ACK)  
0.... .... = Congestion Window Reduced (CWR): Not set  
.0.... .... = ECN-Echo: Not set  
.0. .... = Urgent: Not set

0020 f5 0c 10 1f 00 50 f5 32 64 b2 6b a6 54 92 50 18 .P.2 d.k.T.P.  
0030 fa f0 39 a2 00 00 47 45 54 20 2f 65 74 68 65 72 ..9..GE T /ether  
0040 65 61 6c 2d 6c 61 62 73 2f 6c 61 62 32 2d 31 2e eal-labs /lab2-1.  
0050 68 74 6d 6c 20 48 54 54 50 2f 31 2e 31 0d 0a 48 html HTT P/1.1..H  
0060 6f 73 74 3a 20 67 61 69 61 2e 63 73 2e 75 6d 61 ost: gai a.cs.uma.edu  
0070 72 72 20 65 61 75 0d 0a 55 73 65 72 2d 41 67 65 cs.edu User AGO

Destination Port (tcp.dstport), 2 bytes Packets: 17 Displayed: 17 Marked: 0 Profile: Default

# Export HTTP Objects

Now you can save all files transmitted in Web traffic!

Packet num	Hostname	Content Type	Bytes	Filename
22	www.wireshark.org	application/x-javascript	1300	common.js
28	www.wireshark.org	image/png	137	clear.png
32	www.wireshark.org	application/x-javascript	5141	menu.js
41	www.wireshark.org	image/png	156	nav.bg.png
52	www.wireshark.org	application/x-javascript	1048	mirrors.js
70	www.wireshark.org	application/x-javascript	1213	downloads-1.0.2.js
119	www.wireshark.org	image/png	46317	banner.png
129	s9.addthis.com	image/gif	1505	button1-share.gif
137	s7.addthis.com	application/x-javascript	11373	addthis_widget.js
144	s7.addthis.com	text/css	811	addthis_widget.css
147	www.wireshark.org	image/png	798	feed16.png
159	s7.addthis.com	image/gif	924	addthis-mini.gif

# HTTP Analysis

Tucker Ellis & West internet-capture-113pm-07242008.cap - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: 802.11 Channel: Char

No. Time

1 2008-07-24 13:12:59

2 2008-07-24 13:12:59

3 2008-07-24 13:12:59

4 2008-07-24 13:12:59

5 2008-07-24 13:12:59

6 2008-07-24 13:12:59

7 2008-07-24 13:12:59

8 2008-07-24 13:12:59

9 2008-07-24 13:12:59

10 2008-07-24 13:12:59

11 2008-07-24 13:12:59

12 2008-07-24 13:12:59

13 2008-07-24 13:12:59

14 2008-07-24 13:12:59

15 2008-07-24 13:12:59

16 2008-07-24 13:12:59

17 2008-07-24 13:12:59

18 2008-07-24 13:12:59

19 2008-07-24 13:12:59

Conversation List

Endpoint List

Service Response Time

ANSI

Fax T38 Analysis...

GSM

H.225...

MTP3

RTP

SCTP

SIP...

VoIP Calls

WAP-WSP...

BOOTP-DHCP...

Destinations...

Flow Graph...

HTTP

IP address...

ISUP Messages...

Multicast Streams

ONC-RPC Programs

Packet Length...

Port Type...

SMPP Operations...

TCP Stream Graph

WLAN Traffic...

Load Distribution...

Packet Counter...

Requests...

00 00 15 c7 46 80 00 00 03

0010 05 dc 36 ab 40 00 3b 06

0020 0f 68 00 50 0a f0 94 cf

0030 1b 96 ab f0 00 00 46 1f

0040 9a b1 fd cf c7 ff fd ca

0050 22 c9 05 00 2c 96 fb 25

00 00 15 c7 46 80 00 00 03

01 00 00 00 00 00 00 00 00

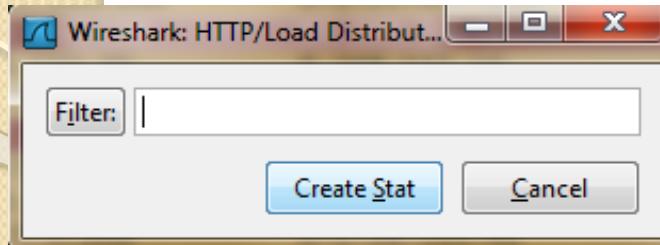
10 00 00 00 00 00 00 00 00

19 00 00 00 00 00 00 00 00

51 00 00 00 00 00 00 00 00

File: "C:\Users\ro2.TEW\Desktop\wireshark\sample capture..." Packets: 16612 Displayed: 16612 Marked: 0 Profile: Default

# HTTP Analysis – Load Distribution



Click “Create Stat” button  
You can add “filter” to only  
Show selected traffic

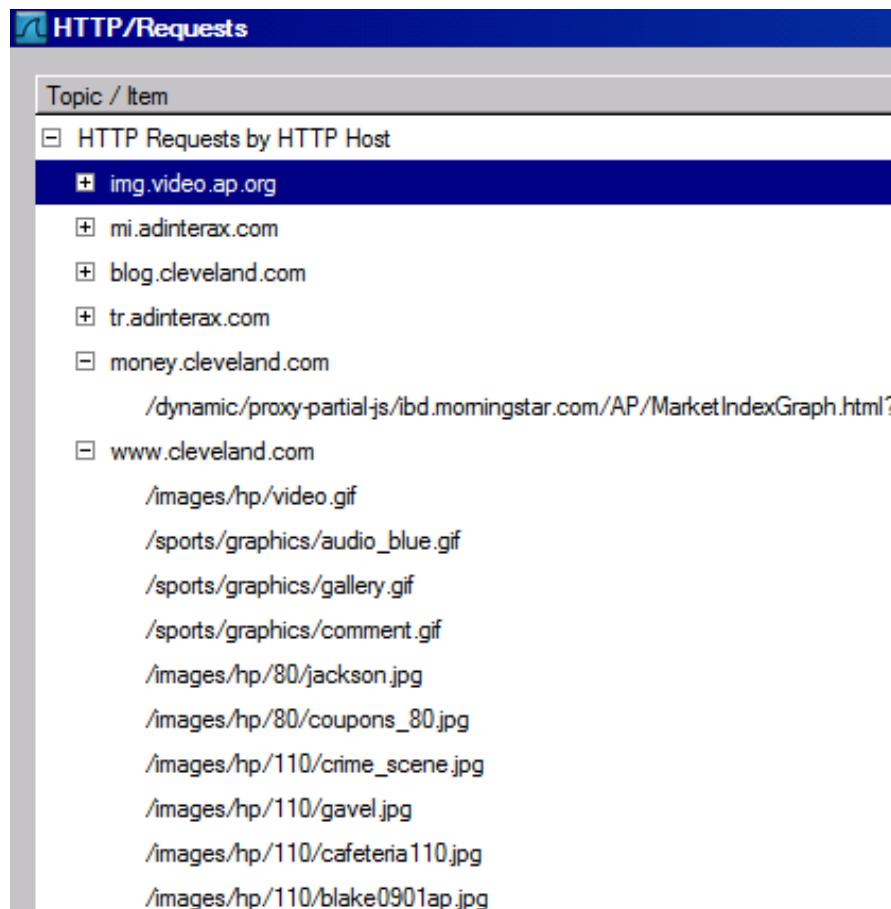
Topic / Item	Count	Rate (ms)	Percent
HTTP Requests by Server	269	0.001013	
+ HTTP Requests by Server Address	269	0.001013	100.00%
+ HTTP Requests by HTTP Host	269	0.001013	100.00%
HTTP Responses by Server Address	168	0.000632	
+ 105.6.102.189	3	0.000011	1.79%
+ 170.31.228.231	1	0.000004	0.60%
+ 170.31.228.246	1	0.000004	0.60%
+ 167.27.156.222	5	0.000019	2.98%
+ 169.230.163.187	1	0.000004	0.60%
+ 242.146.44.213	1	0.000004	0.60%
+ 174.126.120.231	1	0.000004	0.60%
+ 105.6.12.252	2	0.000008	1.19%
+ 167.27.158.24	4	0.000015	2.38%
+ 57.182.235.104	25	0.000094	14.88%
+ 110.120.158.211	33	0.000124	19.64%
+ 127.210.25.255	1	0.000004	0.60%

# HTTP Analysis – Packet Counter

Topic / Item	Count	Rate	Percent
Total HTTP Packets	3267	0.148466	
HTTP Request Packets	915	0.041581	28.01%
GET	859	0.039037	93.88%
POST	47	0.002136	5.14%
HEAD	5	0.000227	0.55%
LOCK	1	0.000045	0.11%
PROPFIND	3	0.000136	0.33%
HTTP Response Packets	877	0.039855	26.84%
?:?: broken	0	0.000000	0.00%
+ 1xx: Informational	1	0.000045	0.11%
+ 2xx: Success	634	0.028812	72.29%
+ 3xx: Redirection	229	0.010407	26.11%
+ 4xx: Client Error	13	0.000591	1.48%
5xx: Server Error	0	0.000000	0.00%
Other HTTP Packets	1475	0.067030	45.15%

[Close](#)

# HTTP Analysis – Requests



The screenshot shows a NetworkMiner interface titled "HTTP/Requests". The left pane displays a hierarchical tree of requests categorized by host. The "img.video.ap.org" node is currently selected, highlighted with a dark blue background. Underneath it, several other hosts are listed with their corresponding request URLs. The right pane contains a detailed list of the selected host's requests, showing the full URL for each.

Host	Request URL
img.video.ap.org	/dynamic/proxy-partial.js/lbd.momingstar.com/AP/MarketIndexGraph.html?
mi.adinterax.com	
blog.cleveland.com	
tr.adinterax.com	
money.cleveland.com	
www.cleveland.com	
	/images/hp/video.gif
	/sports/graphics/audio_blue.gif
	/sports/graphics/gallery.gif
	/sports/graphics/comment.gif
	/images/hp/80/jackson.jpg
	/images/hp/80/coupons_80.jpg
	/images/hp/110/crime_scene.jpg
	/images/hp/110/gavel.jpg
	/images/hp/110/cafeteria110.jpg
	/images/hp/110/blake0901ap.jpg



# Improving Wireshark Performance

- Don't use capture filters
- Increase your read buffer size
- Don't update the screen dynamically
- Get a faster computer
- Use a TAP
- Don't resolve DNS hostnames



# Post-Processing Text File

- For saved text-format packet files, further analysis needs coding or special tools
- One useful tool on Unix: Grep
  - 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
- 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/>



# On-line Wireshark Trace Files

- Public available .pcap files:
  - <http://www.netresec.com/?page=PcapFiles>
- <http://www.tp.org/jay/nwanalysis/traces/Lab%20Trace%20Files/>
- Wiki Sample capture
  - <https://wiki.wireshark.org/SampleCaptures>



# Example Trace File and Questions

- Network Forensic Puzzle Contests
  - <http://forensicscontest.com/2010/02/03/puzzle-4-the-curious-mr-x>
- SharkFest'15 Packet Challenge
  - <https://sharkfest.wireshark.org/assets/presentations/15/packetchallenge.zip>