Using the Wireshark Protocol Analyzer to Troubleshoot Network Problems

Paul Bokor
Network Protocol Specialists, LLC
paul@nps-llc.com

What this class is about

- Installing and Configuring Wireshark
- Using Capture and Display Filters
- Isolating the cause of application or network problems
- Wireshark from the command-line
- ~~Linux~~
- ~~Wireless~~
- ~~Remote Access~~

Who Are We? Network Protocol Specialists

- Established by Mike Pennacchi in 2002
  - mike@nps-llc.com
- Network analysis and training company
- Promoting cost effective, fact-based network analysis and troubleshooting
- Everyone is a trainer and an analyst (Best of both worlds !)
- Perform onsite analysis, coaching, and training nation wide as well as remote trace file analysis
- info@nps-llc.com

Who Am I? Paul Bokor

- Analyst at Network Protocol Specialists, LLC
- Troubleshooting networks for the last 22 years
- Networking instructor for last 20 years
- Top 1/10th of 1% of all Microsoft Executive Briefing Center (EBC) presenters in 2009 and 2010
- Previously a LAN administrator and application developer
- Focused on helping others improve their network troubleshooting skills

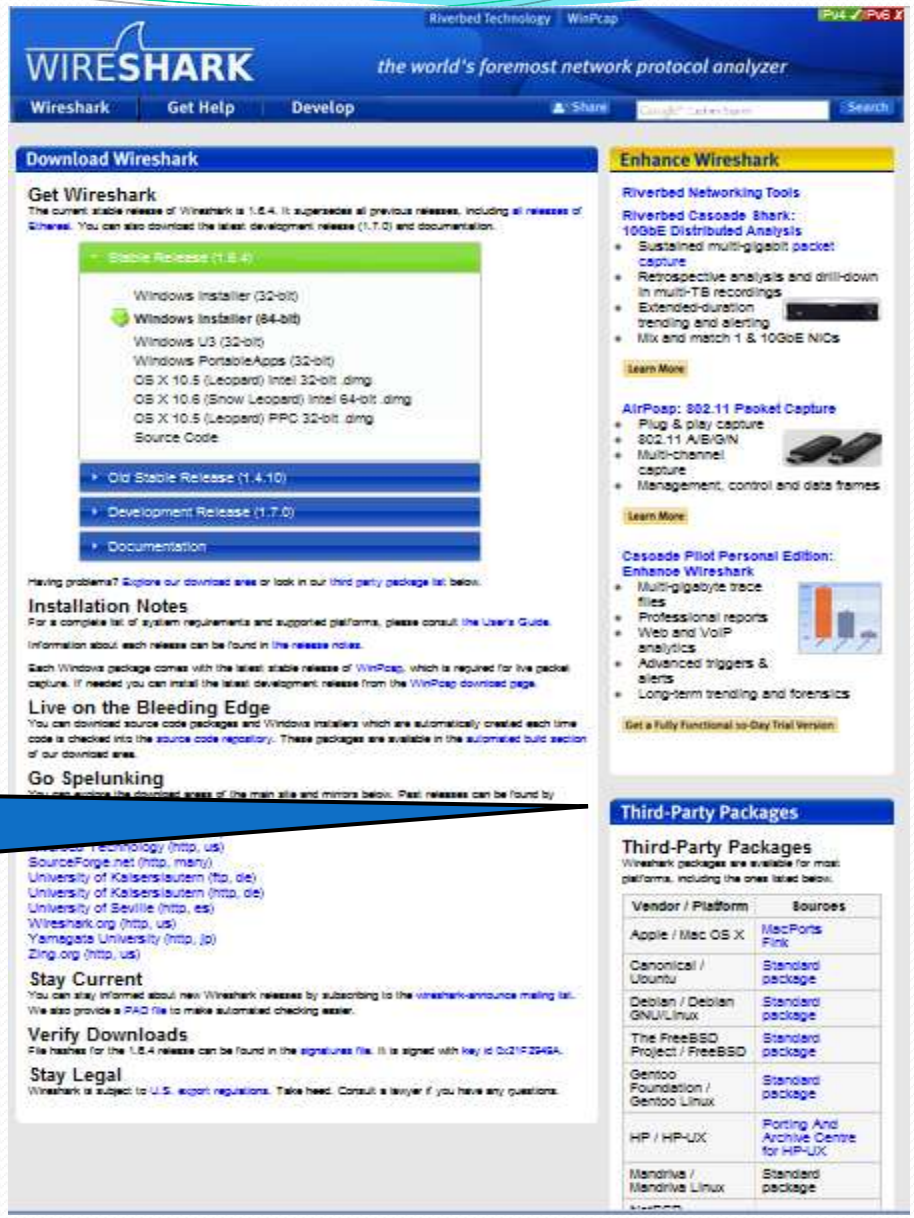
Free CD Contents

Wireshark

Go Deep
Go Quick
Go Ugly

Download

- www.wireshark.org/download.html
- 3-5 minutes on DSL



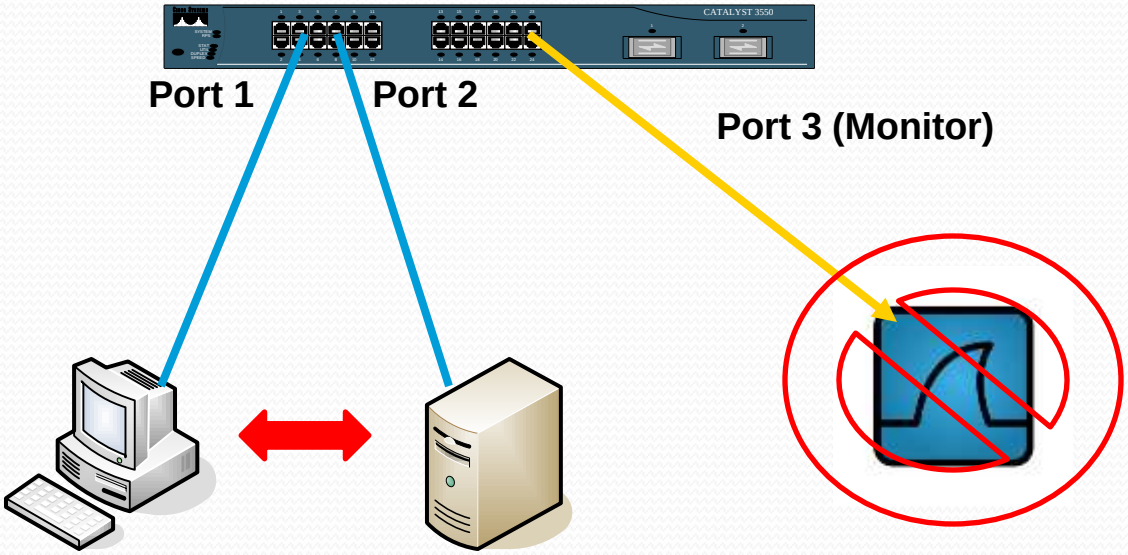
Linux / Open Source packages are located here

Install

- During the Wireshark installation, two components are installed
  - **Wireshark** – Application for configuring the capture filters, setting capture parameters, displaying frames, decoding frames, producing graphs, tables, and statistics
  - **PCAP** (**P**acket **Cap**ture) – API for capturing network traffic
    - **Winpcap** – Drivers used to capture packets in MS-Windows environment (CACE/Riverbed.com)
    - **Libcap** – Drivers used to capture packets in *NIX environment (tcpdump.org)

Monitoring Network Traffic

- When attached to a standard switch port, the Wireshark analyzer will not be able to observe traffic on adjacent switch ports. It will not be in the "path" of packets



Traffic Monitoring Methods

Hub

- Pros: Cheap, Somewhat Available, Easy to install
- Cons: Reduce link to half duplex, Changes the network under test, May not be a true hub, Not practical on servers or switch uplinks, If power drops, link drops, 10/100 Mbps speeds only

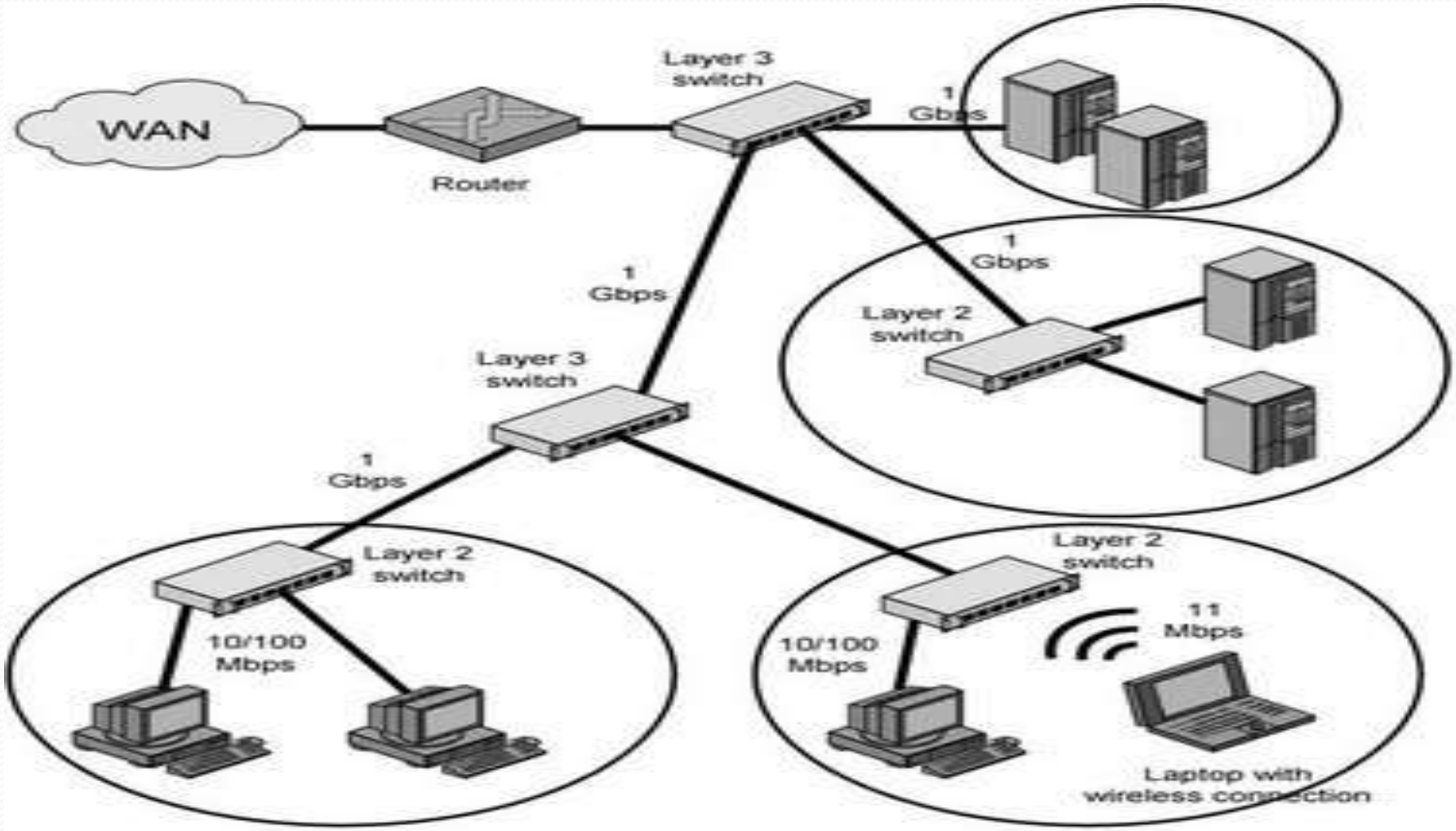
Inline Tap

- Pros: Truly monitors full-duplex traffic, If power is lost link stays active, Can monitor 1 Gigabit and 10 Gigabit links without packet loss, Once installed, can stay
- Cons: Most expensive option, Have to break the link to install, Can over-provision the monitor port and drop packets

Switch span

- Pros: Free, Available, Does not require link to be dropped, Great for one-time link monitoring
- Cons: Requires switch access, configuration, Can quickly become over provisioned, Requires a free switch port

Monitoring Network Traffic



Setup – Main Page

WIRESHARK

The World's Most Popular Network Protocol Analyzer
Version 1.6.1 (SVN Rev 38096 from /trunk-1.6)

Capture

Interface List
Live list of the capture interfaces
(counts incoming packets)

Start capture on interface:

- Atheros L1C PCI-E Ethernet Controller
- Microsoft
- Microsoft
- VMware Virtual Ethernet Adapter
- VMware Virtual Ethernet Adapter

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

Open Recent:

- C:\Users\PBokor\Desktop\NPS CD\FNet Dir\BrokenDNS.cap (6
- Z:\1 NTFS\nps\Tracefiles\VerySlowHTTP.pcap [not found]
- Z:\1 NTFS\nps\Client ... PC\Dayton\Lion Stress Citrix Server.p
- F:_Captures\Office\ ... ffice wlan 4min - video stream - ina.ca
- F:_Captures\pkts_12-22-11_8_25_AM.cap (3096 Bytes)
- F:_Captures\HomeWireless_freeze_1605.cap [not found]
- C:\Users\PBokor\Desk ... ts\Glenn Friedland\Slow_Print_Clean.
- C:\Users\PBokor\Desktop\Slow_Print_Clean.pcap [not found]
- C:\Users\PBokor\Desktop\Slow_Print_Full.pcap [not found]
- C:\Users\PBokor\Desktop\Slow_Print_1.pcap [not found]
- C:\Users\PBokor\Downloads\waiting for data message.pcap (2
- C:\Users\PBokor\Desk ... ta message_prn_and_pserver_tcp_o
- C:\Users\PBokor\Desk ... r data message_prn_and_pserver_o
- C:\Users\PBokor\Desk ... d\waiting for data message_c-s_only
- C:\Users\PBokor\Desk ... Friedland\waiting for data message.

Sample Captures
A rich assortment of example capture files on the wiki

Online

Website
Visit the project's website

User's Guide
The User's Guide (local version, if installed)

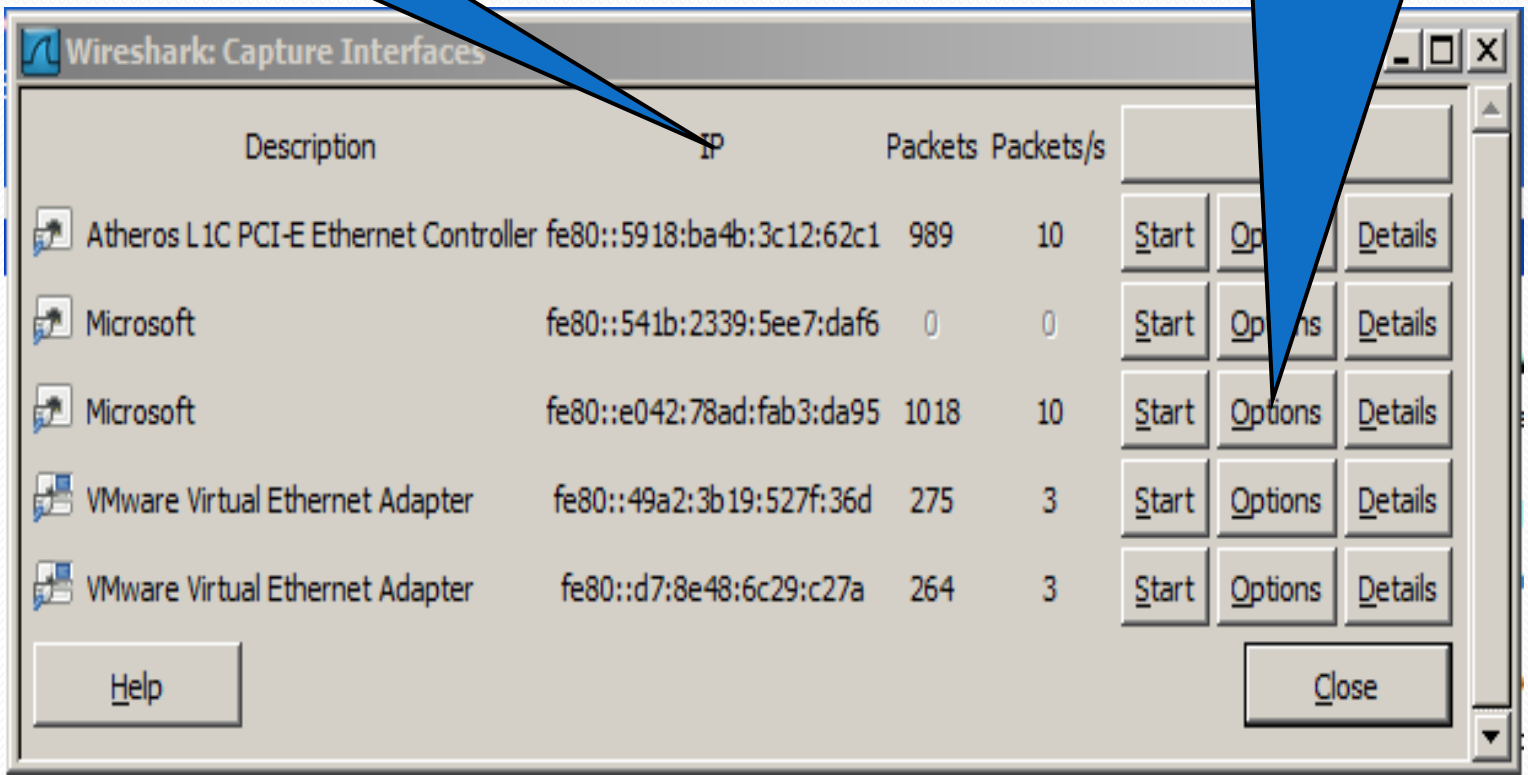
Security
Work with Wireshark as securely as possible

Setup – Select Interface Card

For each interface, the address and packet info are displayed

Start capture, set options, or view Details.

Select Options for buffer size, filters, and more



Setup – Capture Options

Capture Options

Capture Files & Stop Capture

Display Options & Name Resolution

Danger: Will flood DNS Server

Start Capture

Wireshark: Capture Options

Capture

Interface: Local | Atheros L1C PCI-E Ethernet Controller: \Device\NPF_{1ACA89FC-E

IP address: fe80::5918:ba4b:3c12:62c1, 192.168.1.108

Link-layer header type: Ethernet

Wireless Settings

Remote Settings

Capture packets in promiscuous mode

Capture packets in pcap-ng format

Buffer size: 1 megabyte(s)

Limit each packet to 65535 bytes

Capture Filter: | Compile BPF

Capture File(s)

File: | Browse...

Use multiple files

Next file every 1 megabyte(s)

Next file every 1 minute(s)

Ring buffer with 2 files

Stop capture after 1 file(s)

Stop Capture ...

... after 1 packet(s)

... after 1 megabyte(s)

... after 1 minute(s)

Display Options

Update list of packets in real time

Automatic scrolling in live captu

Hide capture info dialog

Name Resolution

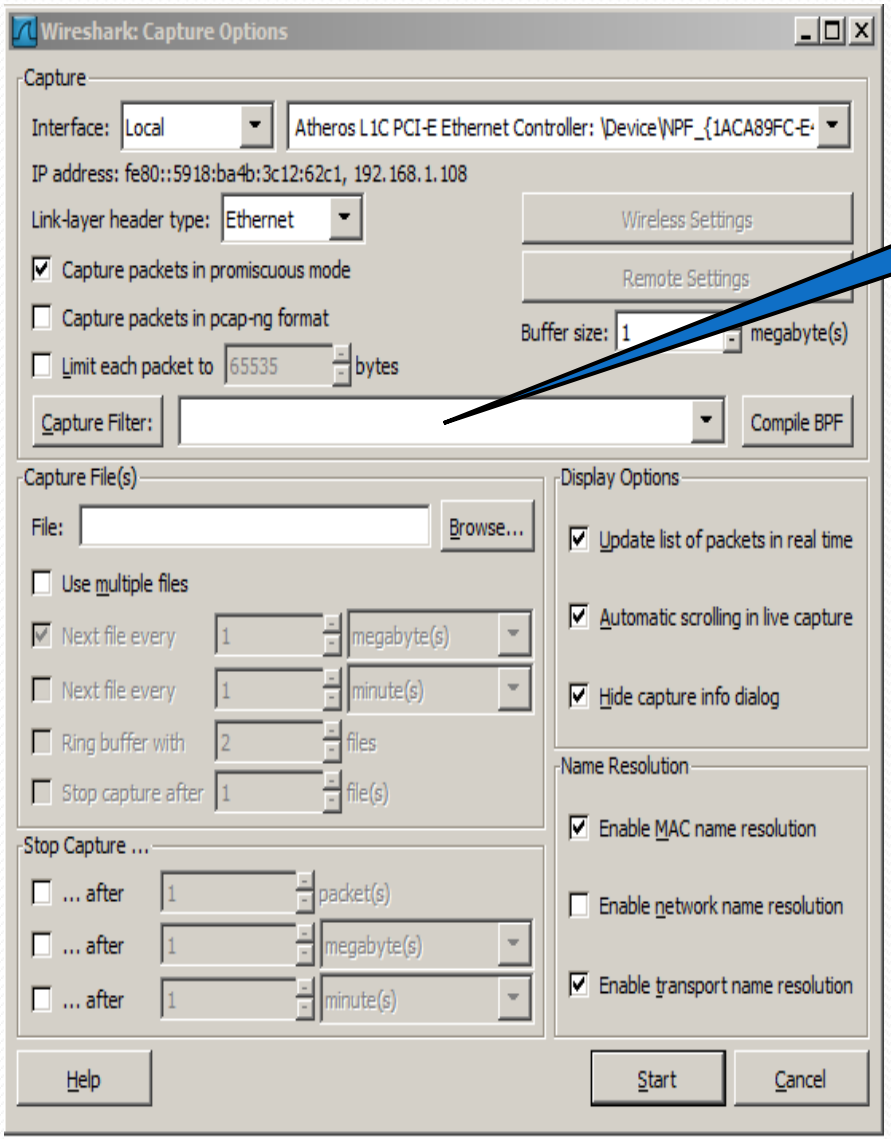
Enable MAC name resol

Enable network name resolution

Enable transport name resolution

Help | Start | Cancel

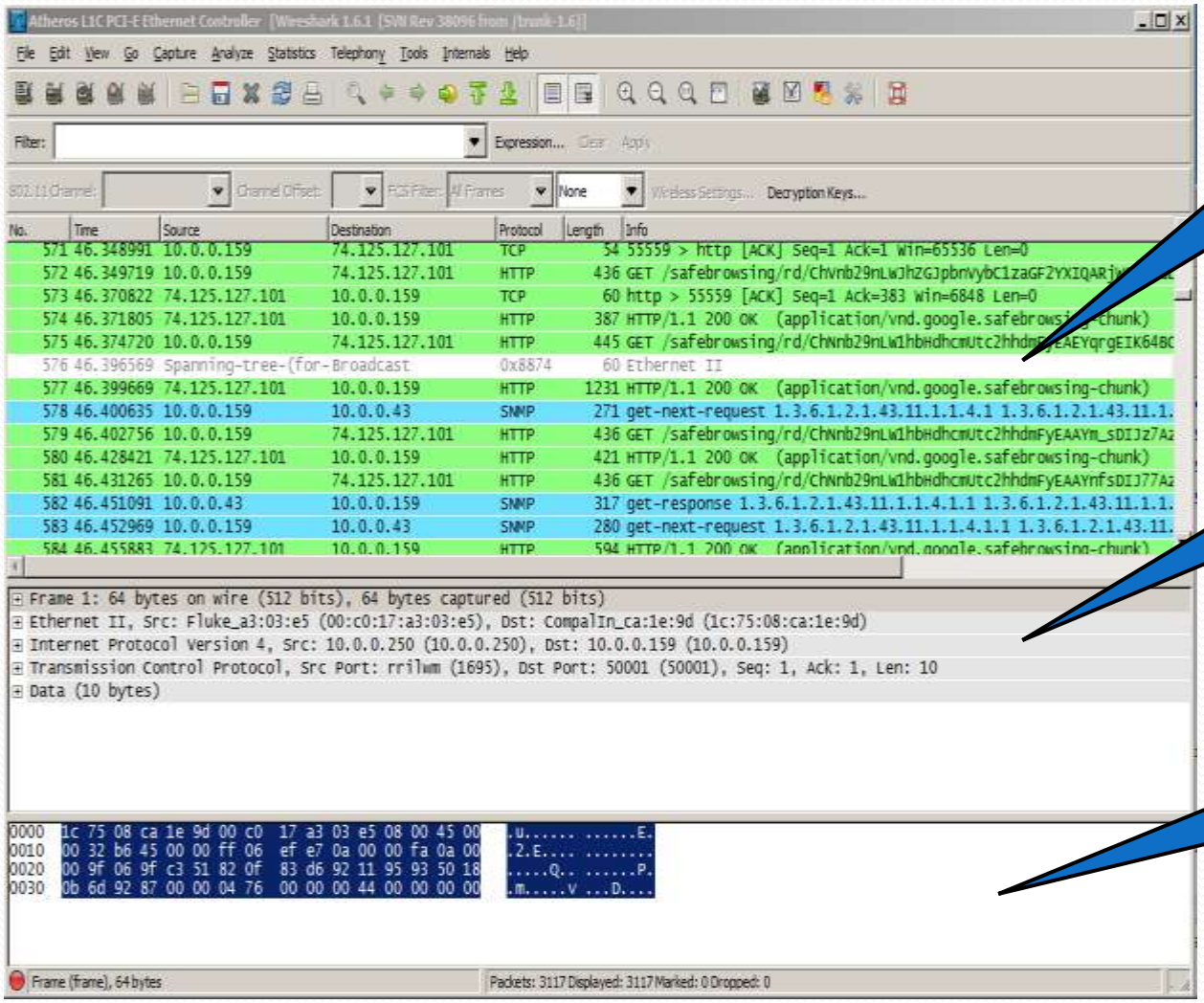
Setup – Capture Filters



Capture Filters

- COMPLETELY different from Display Filters
- Uses tcpdump filter language
- Series of primitives joined by **and / or / not**
- Examples:
  - tcp port 23 and host 10.0.0.5
  - tcp port 23 and not src host 10.0.0.5
  - not broadcast and not multicast

Capture: Viewing Frames



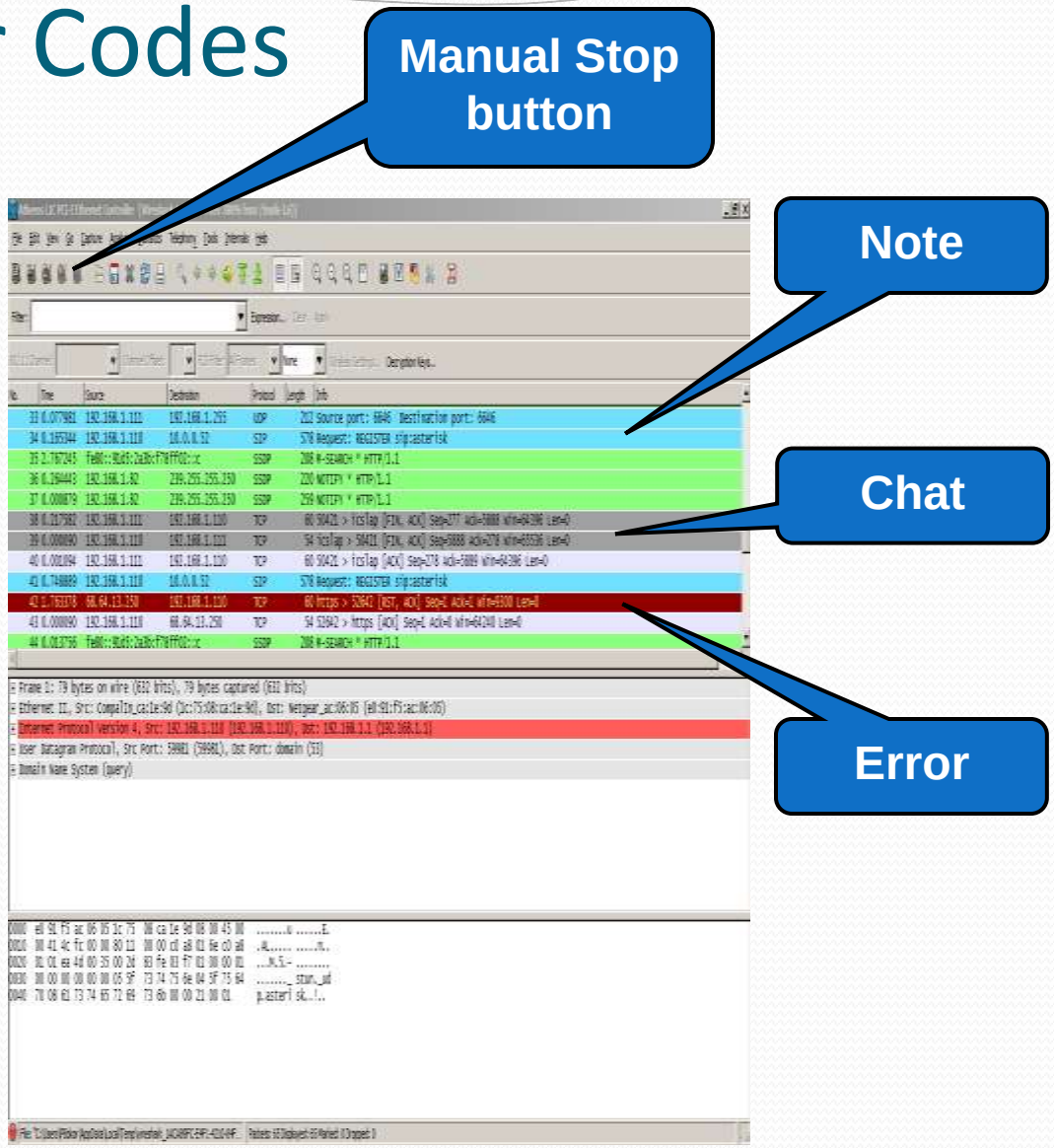
Scrolling Packet List

Packet Detail

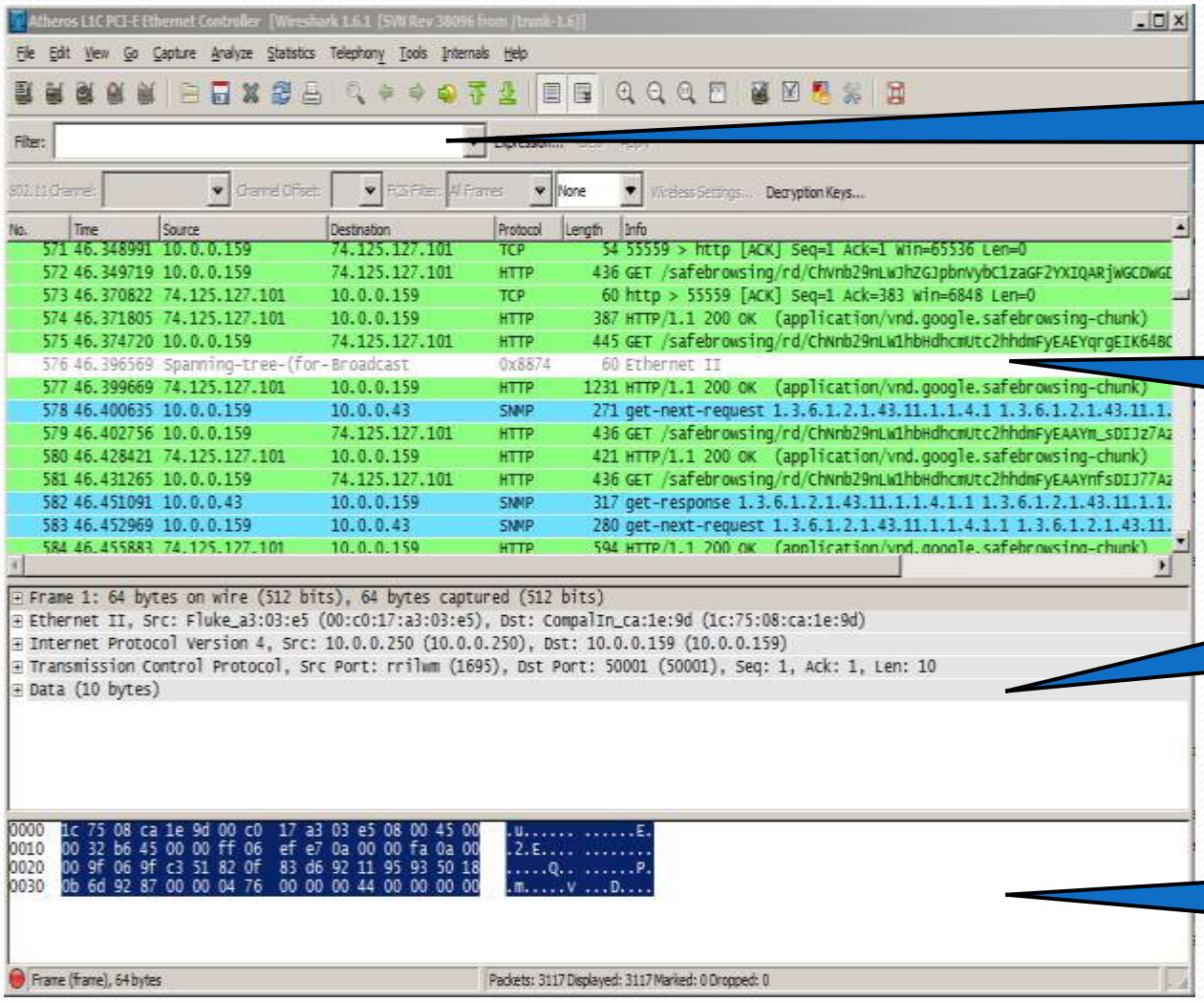
Packet Bytes

Capture – Color Codes

- Color Coding allows quick and easy identification
  - Grey - Normal
  - Cyan – Usual Error
    - http 404
  - Yellow- Unusual Error
    - Fast Retrans
  - Red-Serious Problem
    - Malformed Packet
- Capture will continue until:
  - Manually stopped
  - Programmatically stopped



Analyze: Viewing Frames



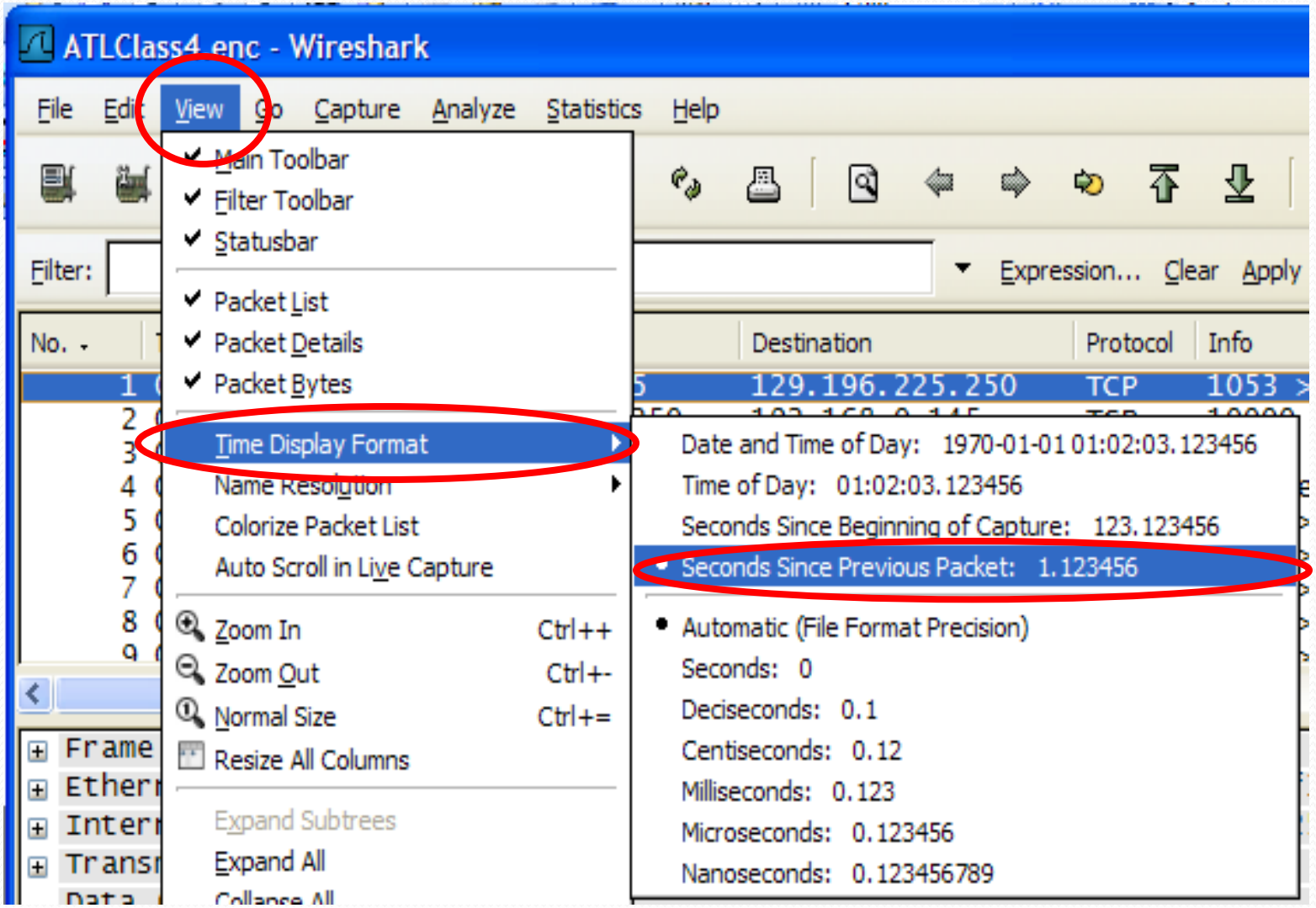
Display Filter

Packet List

Packet Detail

Packet Bytes

Analyze – Time Format – Delta Time



Analyze – Reading the Time

TCP Three-way Handshake

5	1.374060154	192.168.0.145	66.151.158.177	TCP	2099 > 8200 [SYN] Seq=0
6	0.070454836	66.151.158.177	192.168.0.145	TCP	8200 > 2099 [SYN, ACK] S
7	0.001919985	192.168.0.145	66.151.158.177	TCP	2099 > 8200 [ACK] Seq=1

Seconds

Microseconds

1.374060154

Milliseconds

Nanoseconds

Analyze – It's all about timing

- "The Network is Slow!" – This is usually why we are capturing packets and analyzing them.
- Trace files of slow applications will contain one of two things:
  - Few frames with long times between each frame.
  - Many frames with short times between each frame.

Analyze – Sum of the parts

- Summing the delta times will yield the total transaction time.
  - "Time reference" makes it easy
- When packing for a hiking trip, we count ounces, not pounds.
- When analyzing trace files, we count milliseconds, not seconds.
- **Find the delays and you will find the cause of the slowdown.**

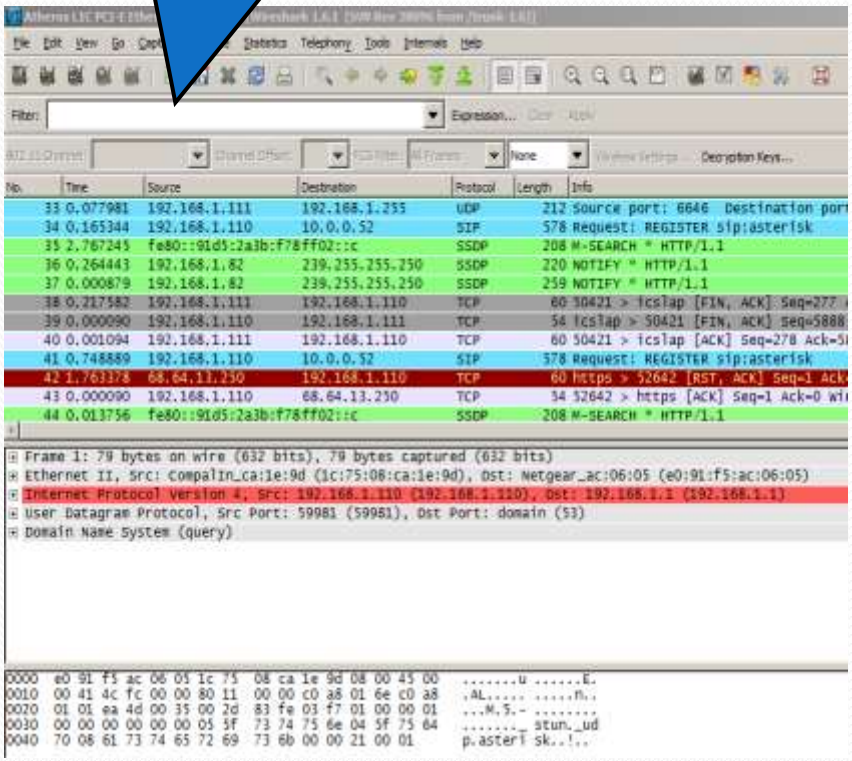
Analyze – Back to the handshake

5	1.374060154	192.168.0.
6	0.070454836	66.151.158
7	0.001919985	192.168.0.

- **Frame 5** – TCP SYN – Start of handshake, we don't care about the delta time.
- **Frame 6** – TCP SYN/ACK – Response from server. Represents round trip time between client and server. This took 70.454 milliseconds.
- **Frame 7** – TCP ACK – Sent by client. This took 1.919 milliseconds.

Analyze – Enter basic filters

Enter filter here. Turns green when valid, red means not so much.

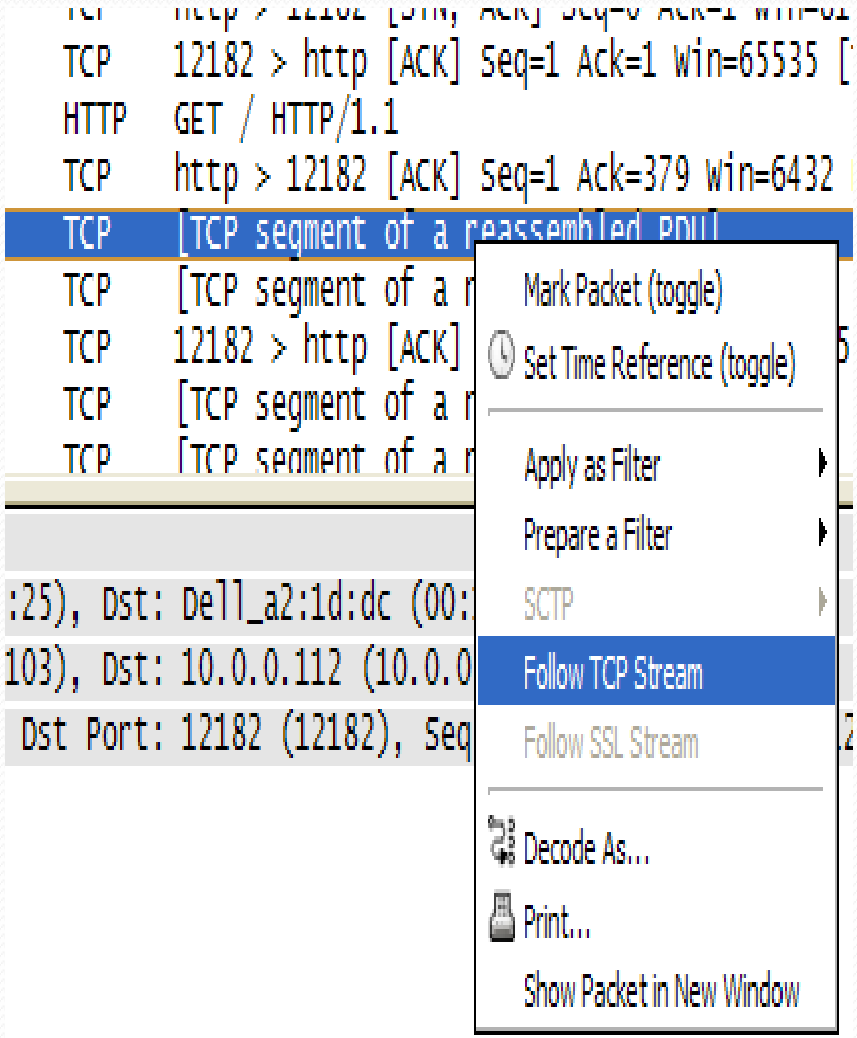


- Filtering in Wireshark can get quite complex.
- Operators:
  - eq / ne == / !=
  - gt / lt > / <
  - ge / le >= / <=
- Logic:
  - and &&
  - or ||
  - xor ^^
  - not !
- Misc
  - contains
  - matches (perl,string within field)

Analyze – More Display Filters

- Ethernet
  - eth.addr
  - eth.src
  - bootp (dhcp)
- 802.1Q
  - vlan.id
- IPv4
  - ip.addr
  - ip.dst
- tcp contains “google”
- TCP
  - tcp.flags
  - tcp.analysis.flags
  - tcp.segment
  - tcp.window_size
- http
  - http.connection
  - http.host
  - http.request
  - http.response

Follow TCP Stream



- Select any frame that is part of a conversation of interest.
- Right click on the frame.
- Select Follow TCP Stream.
- Wireshark will create a filter on that IP address pair and port numbers.
- The data portion of the conversation will be assembled into a text window.

Follow TCP Stream

Follow TCP Stream

Stream Content

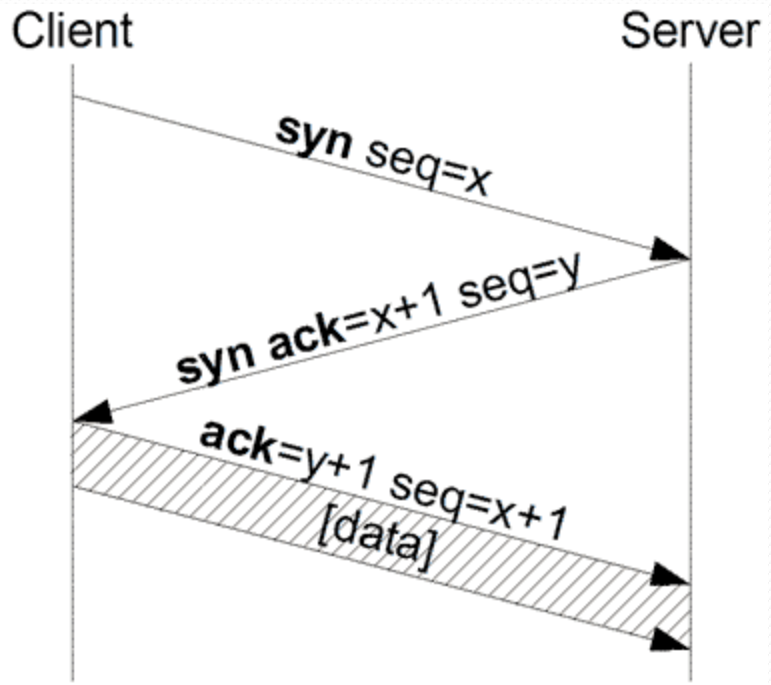
```
GET / HTTP/1.1
Accept: */*
Accept-Language: en-us
UA-CPU: x86
X-McProxyFilter: **************
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR
1.1.4322; .NET CLR 2.0.50727; Google-TR-3)
Host: www.google.com
Connection: Keep-Alive
Cookie:
PREF=ID=aa972514870a6949:FF=4:TB=2:LD=en:NR=10:TM=1164937700:LM=1165217462:S=QwfVr3
testcookie=

HTTP/1.1 200 OK
X-TR: 1
Cache-Control: private
Content-Type: text/html; charset=UTF-8
Server: GWS/2.1
Transfer-Encoding: chunked
Date: Tue, 01 May 2007 18:27:00 GMT
```

Save As | Print | Entire conversation (5362 bytes) | ASCII | EBCDIC | Hex Dump | C Arrays | Raw

Filter Out This Stream | Close

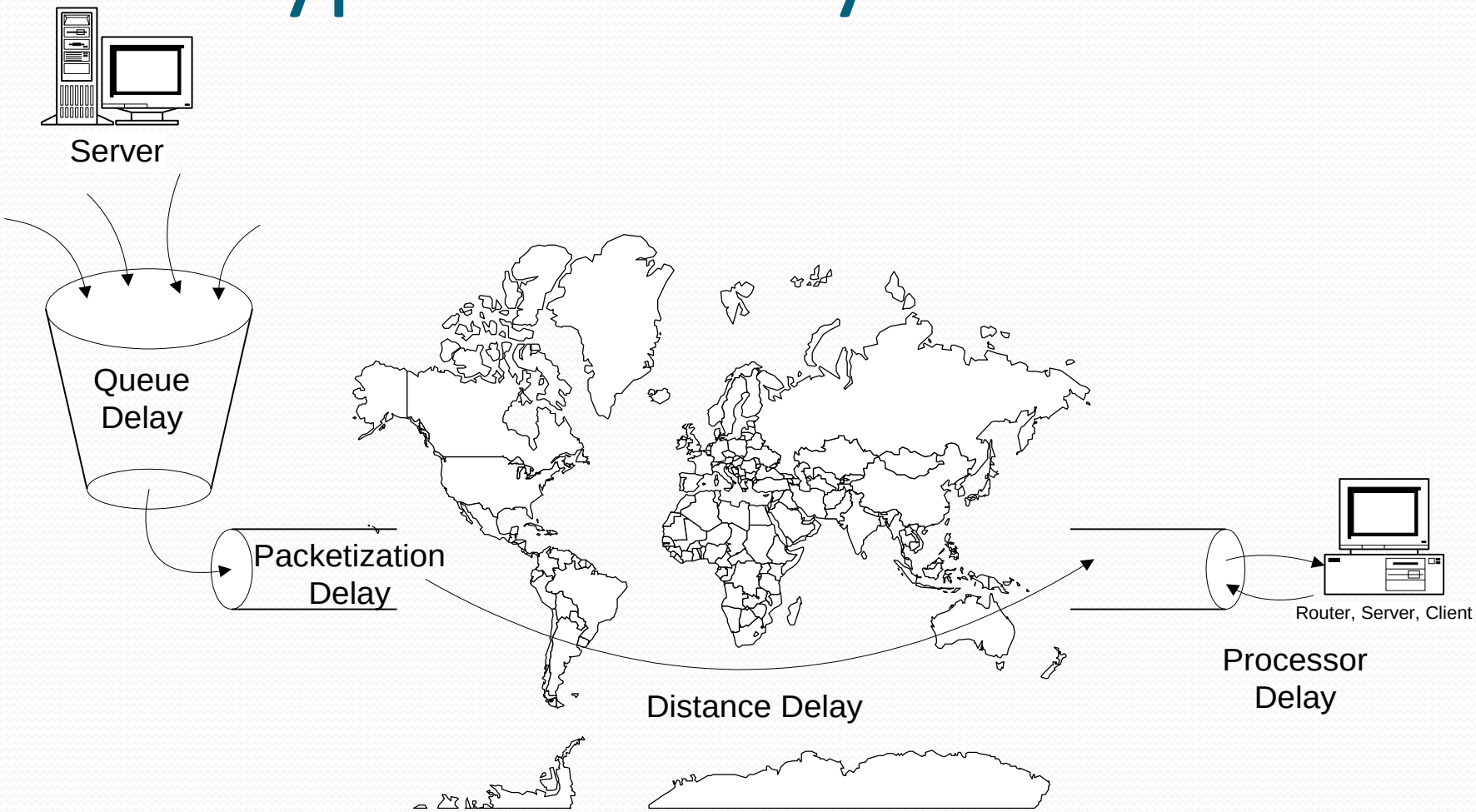
Network RTT

- If the capture was taken client-side, the RTT can be measured from the SYN/SYN-ACK.
- If the capture was taken server-side, the RTT can be measured using the SYN-ACK/ACK in the handshake.



- Why is this the case? (Discuss as a class)

Types of Delay



Queue Delay: 0 to several seconds
Packetization Delay: 1000 Bytes at: 1544 Kbps = 5.2 ms, 512 Kbps = 16 ms, 128 Kbps = 63 ms
Distance Delay: Fiber Speed = .7C, 1000 miles = 7.6 ms
Processor Delay:Typical router Ping reply 0 to 40 ms when CPU busy

Measuring Server Performance

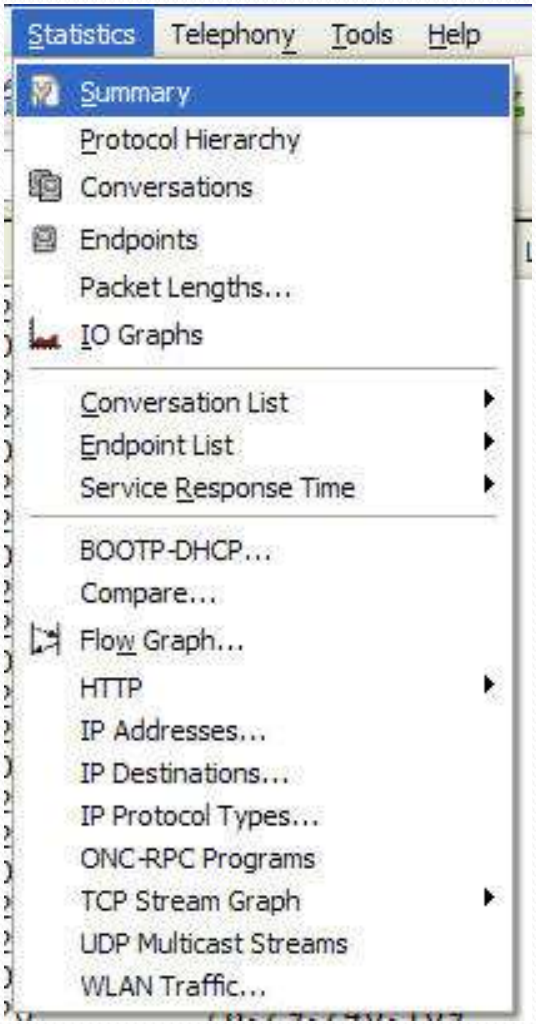
- It is best to measure server performance when capturing near the server. This way network delay does not affect the timers.
- To measure how long a server takes to respond to a request, simply look at the time between when the request is received and when the response is sent.
- A simple way to observe this is when looking at an HTTP transaction. The time between a GET and the first byte from the server is the server response time.

9	0.000495	192.168.0.3	167.187.3.153	406 HTTP	GET / HTTP/1.1
10	0.125025	167.187.3.153	192.168.0.3	64 TCP	http > telindus [ACK] Seq=1 Ack=349 win
11	4.851946	167.187.3.153	192.168.0.3	249 TCP	[TCP segment of a reassembled PDU]

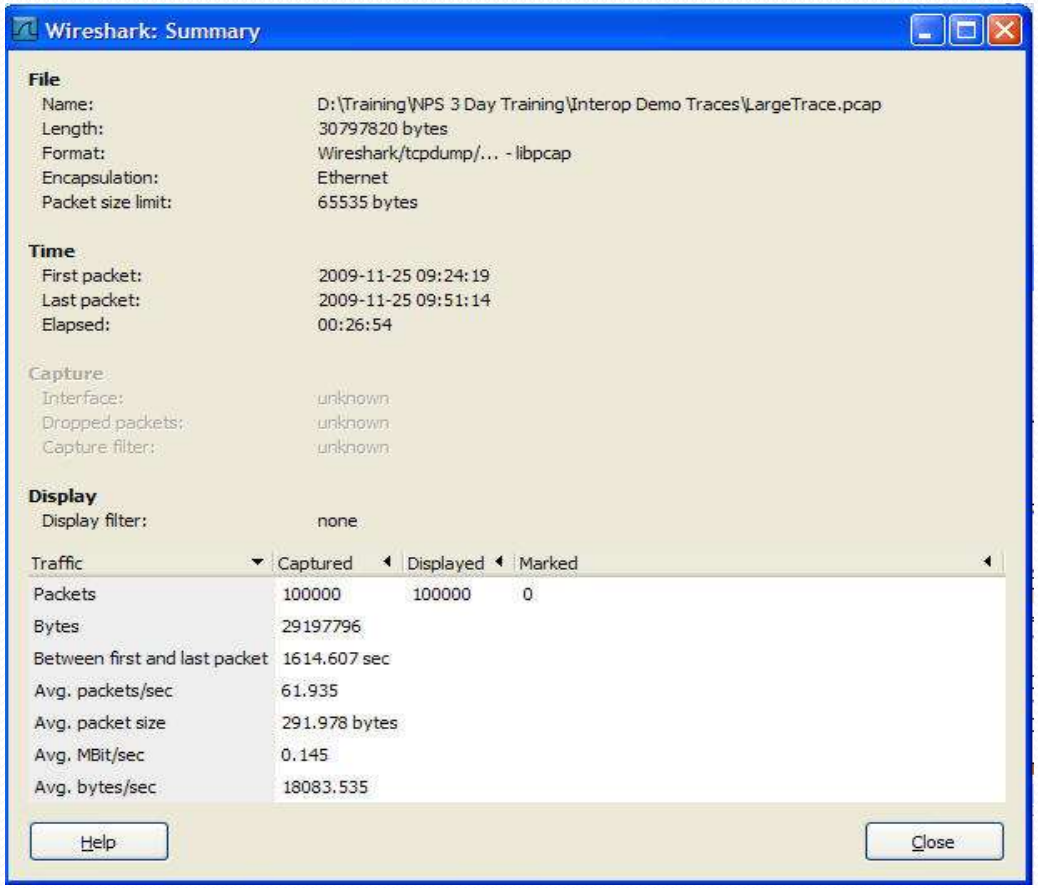
Traffic Statistics

Statistics

- Wireshark can provide statistics on traffic in a trace file.
- This makes determining top talkers, protocols, and conversations very easy.



Statistics - Summary



Statistics – Protocol Hierarchy

Wireshark: Protocol Hierarchy Statistics

Display filter: none

Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes
Frame	100.00 %	100000	29197796	0.145	0	0
Ethernet	100.00 %	100000	29197796	0.145	0	0
Internet Protocol	98.67 %	98670	29111288	0.144	0	0
User Datagram Protocol	52.47 %	52468	9496820	0.047	0	0
Data	11.64 %	11638	3544565	0.018	11638	3544565
Session Initiation Protocol	0.86 %	864	499958	0.002	864	499958
Syslog message	0.54 %	544	93239	0.000	544	93239
Simple Network Management Protocol	37.50 %	37502	5070807	0.025	37502	5070807
Domain Name Service	0.89 %	886	106211	0.001	886	106211
Service Location Protocol	0.40 %	399	95603	0.000	399	95603
NetBIOS Name Service	0.30 %	300	27780	0.000	300	27780
Hypertext Transfer Protocol	0.11 %	114	24465	0.000	114	24465
NetBIOS Datagram Service	0.07 %	69	16570	0.000	0	0
SMB (Server Message Block Protocol)	0.07 %	69	16570	0.000	0	0
SMB MailSlot Protocol	0.07 %	69	16570	0.000	0	0
Microsoft Windows Browser Protocol	0.07 %	69	16570	0.000	69	16570
T.38	0.04 %	37	2220	0.000	0	0
Unreassembled Fragmented Packet	0.00 %	2	120	0.000	2	120
Malformed Packet	0.04 %	35	2100	0.000	35	2100
Teredo IPv6 over UDP tunneling	0.10 %	97	12766	0.000	0	0

Help Close

Statistics - Conversations

Conversations: LargeTrace.pcap

Ethernet: 52 | Fibre Channel | FDDI | IPv4: 1027 | IPX | JXTA | NCP | RSVP | SCTP | TCP: 1408 | Token Ring | UDP: 7367 | USB | WLAN

Ethernet Conversations

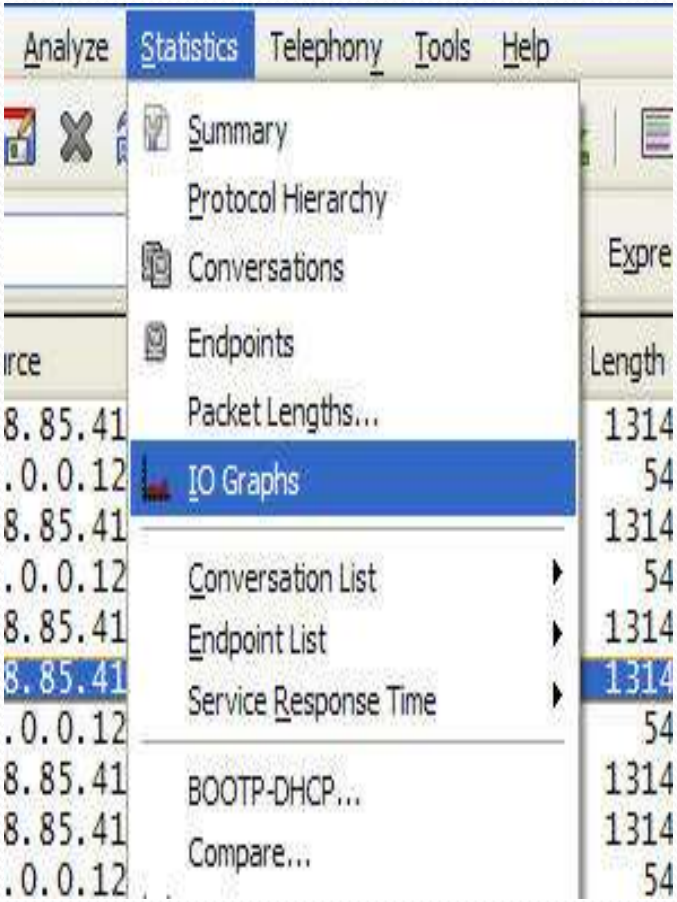
Address A	Address B	Packets	Bytes	Packets A->B	Bytes A->B	Packets A<-B	Bytes A<-B	Rel Start
Cisco_e6:9e:62	Cisco-Li_85:8b:20	1	60	0	0	1	60	1437.053803
AlphaNet_3a:d7:ce	Broadcast	1	60	1	60	0	0	1454.055674
HewlettP_ed:59:4a	Broadcast	2	120	2	120	0	0	970.8332080
Cisco_e6:9e:62	Broadcast	3	180	3	180	0	0	1293.232306
Grandstr_18:8e:bc	Cisco-Li_85:8b:20	3	270	3	270	0	0	1597.601806
AsustekC_21:79:bc	Fluke_a3:02:29	4	328	2	208	2	120	571.9120810
Dell_ca:4e:9d	Fluke_a3:02:29	4	328	2	208	2	120	571.9121740
Dell_a2:1d:dc	Fluke_a3:02:29	4	352	2	232	2	120	571.9124320
AsustekC_21:86:30	Broadcast	5	762	5	762	0	0	107.8158420
Fluke_a3:02:29	IPv4mcast_00:00:fd	5	410	5	410	0	0	318.7173980
AsustekC_21:79:bc	Broadcast	5	704	5	704	0	0	61.83793900
American_67:45:52	Broadcast	6	360	6	360	0	0	118.4782370
American_c2:d6:40	Broadcast	6	360	6	360	0	0	54.34674500
Cisco-Li_85:8b:20	American_c2:d6:40	8	658	6	360	2	298	54.34744900
Dell_a3:ea:80	Dell_ca:4e:9d	8	1247	5	476	3	771	54.40195700

☑ Name resolution ☐ Limit to display filter

Help | Copy | Close

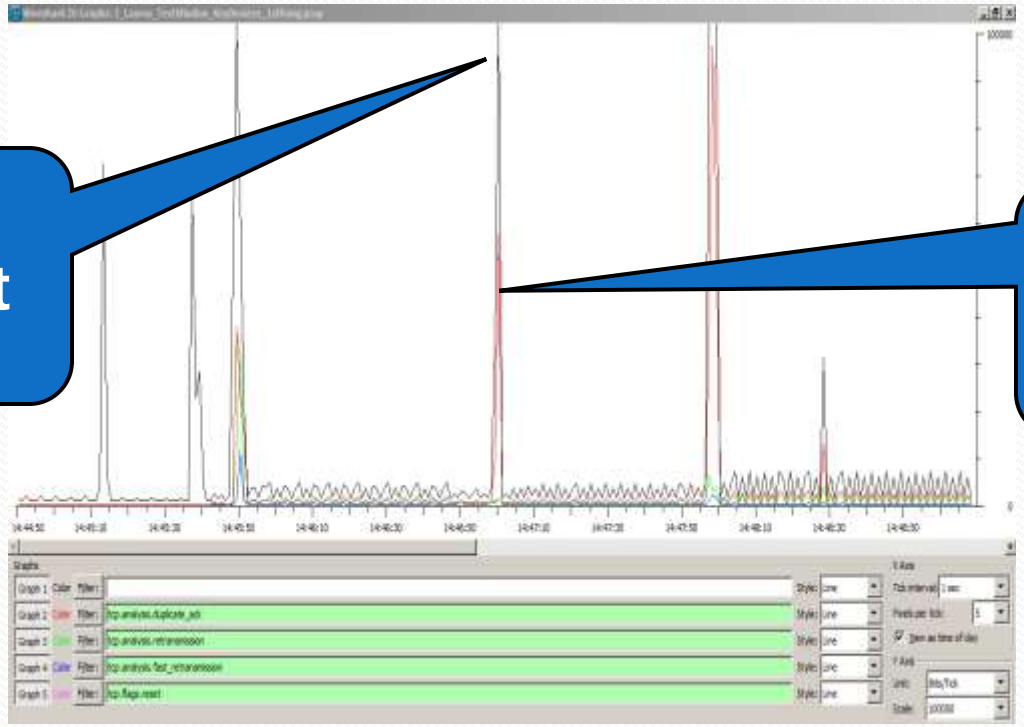
I/O Graphs

- Very powerful method of conveying complex data, especially associations
- Basic use assists in measuring bandwidth consumed by a specific application, client, protocol, and much more.
- X and Y axis can both be modified
- Advanced use includes ability to include calculations



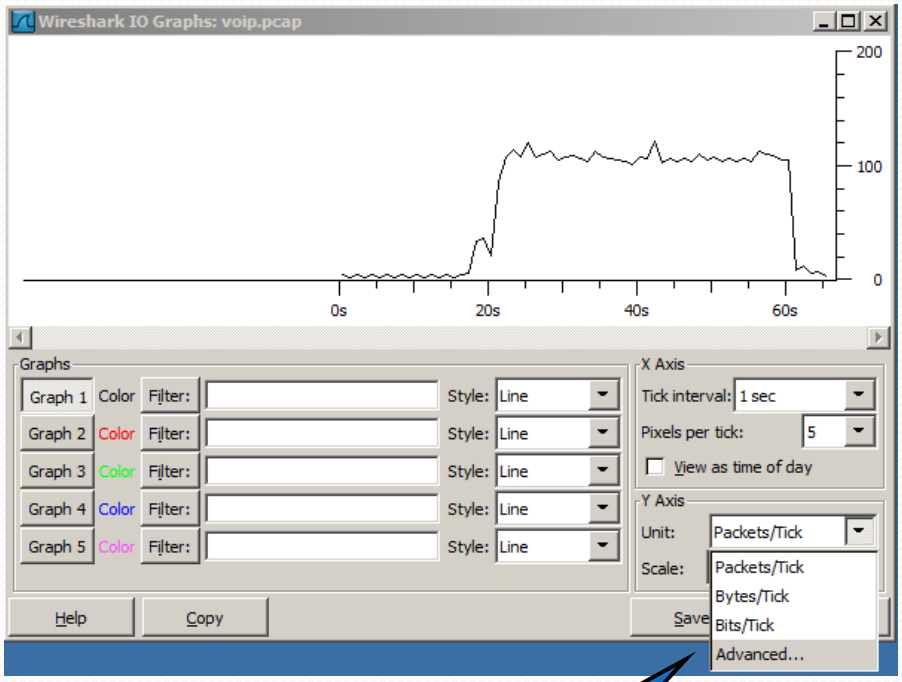
Basic I/O Graph

- This graph shows the relationship between a clients total BW consumption and the adverse affect of duplicate acknowledgments.



Advanced I/O Graph

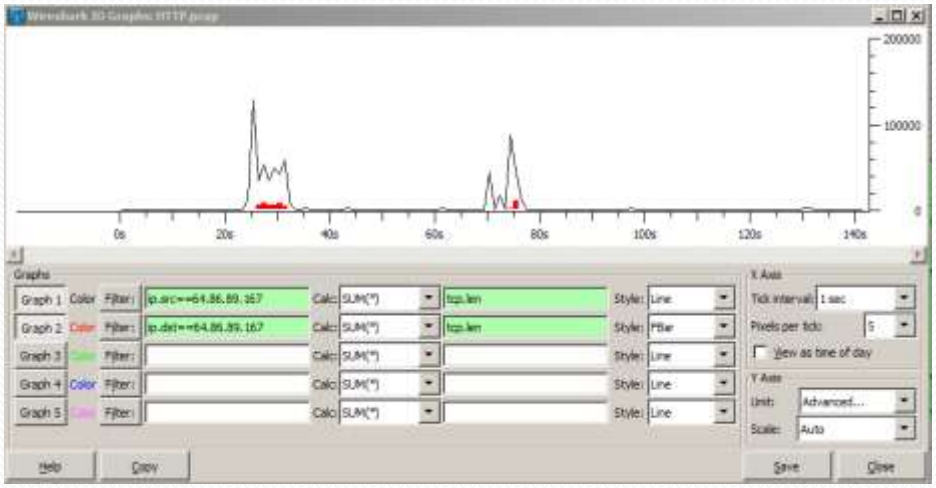
- Accessed from Y axis drop down.
- Provides the following calc options:
  - SUM
  - MIN
  - AVG
  - MAX
  - COUNT
  - LOAD



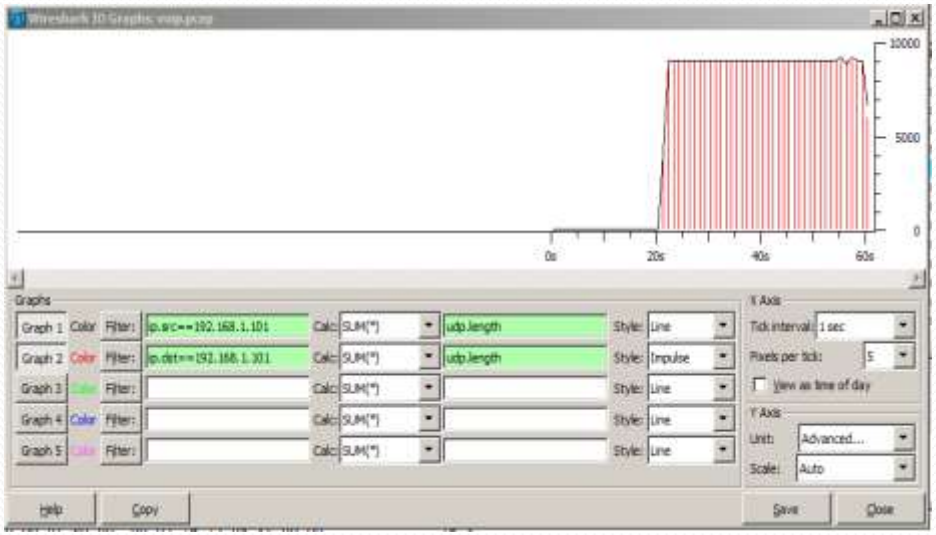
Cleverly hidden here

Advanced I/O Graph

- HTTP Session
- Asymmetrical data transfer
- Small requests, large response profile

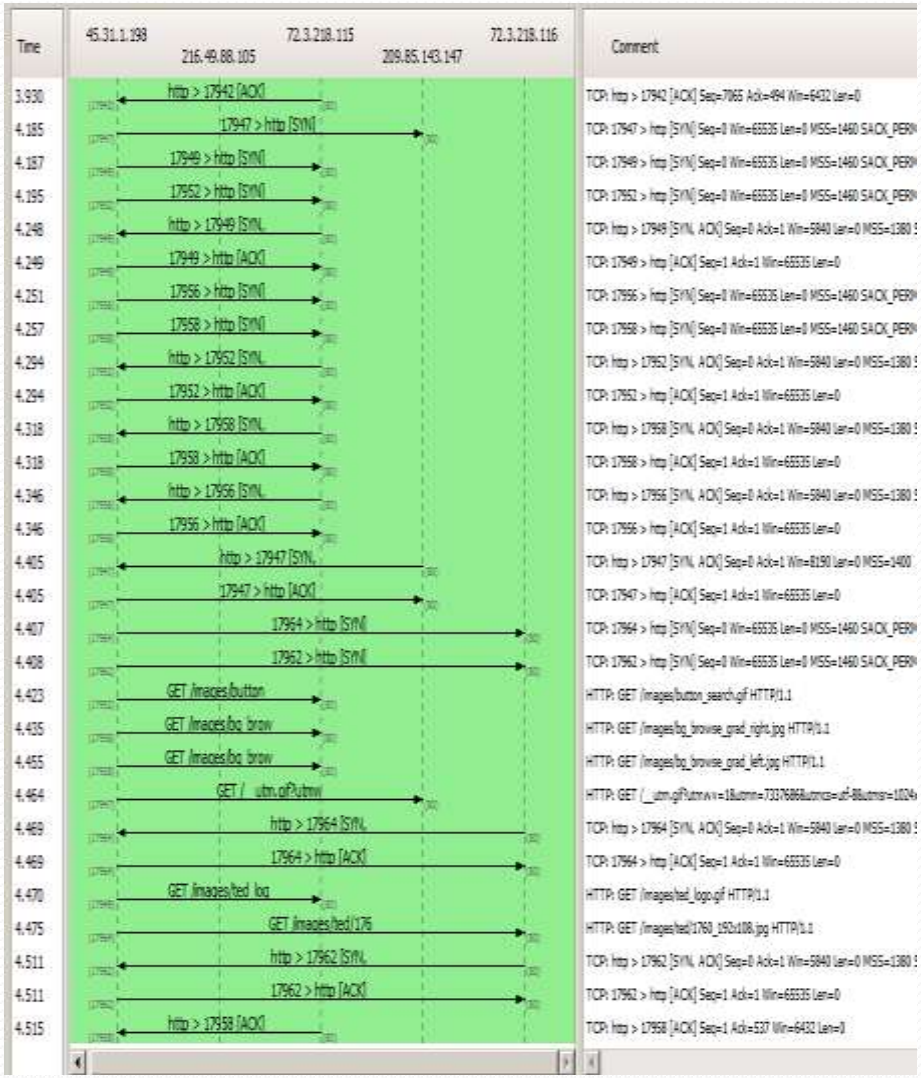


- VOIP Session
- Symmetrical data transfer
- Identical request, response profile



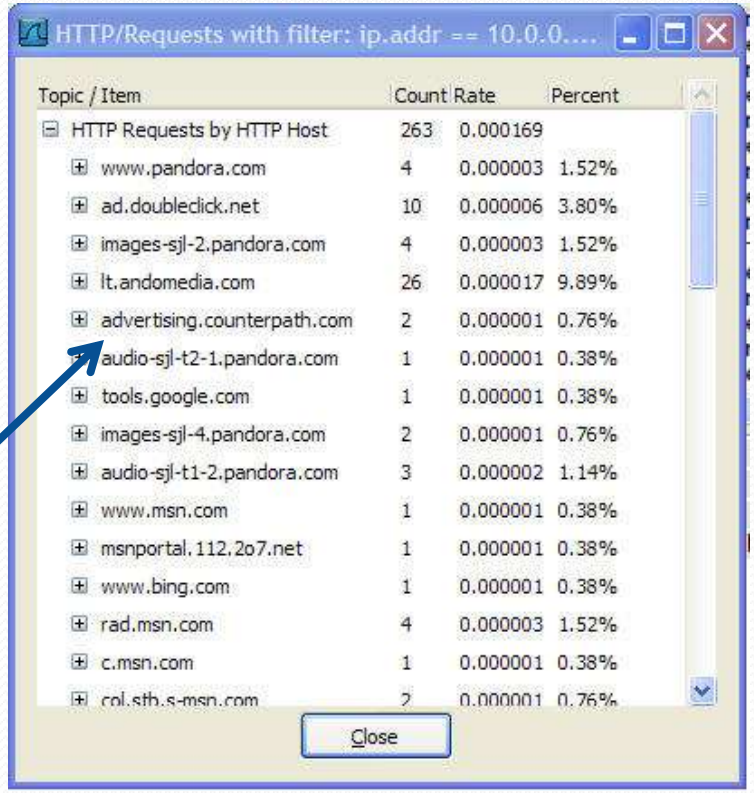
Flow Graph

- Very useful for n-tiered application analysis
- Shows relationships, dependencies, and delays throughout entire system



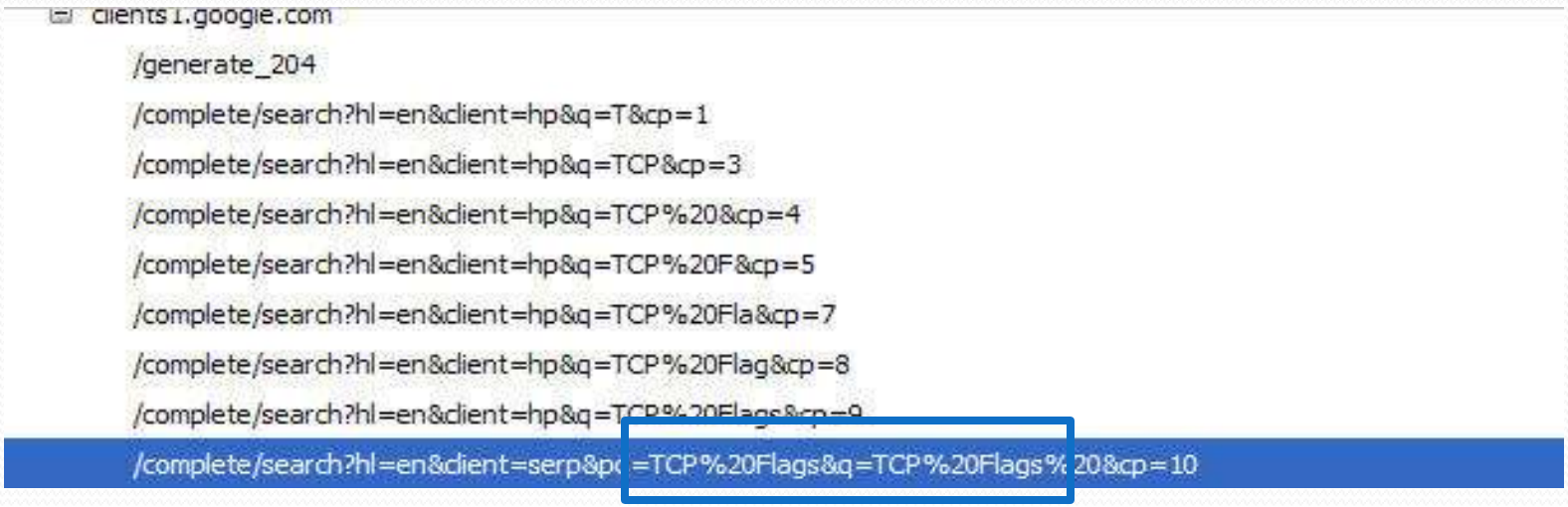
Statistics - HTTP

- Enter an address filter into the box.
- A list of the HTTP requests sent from this machine will be created.



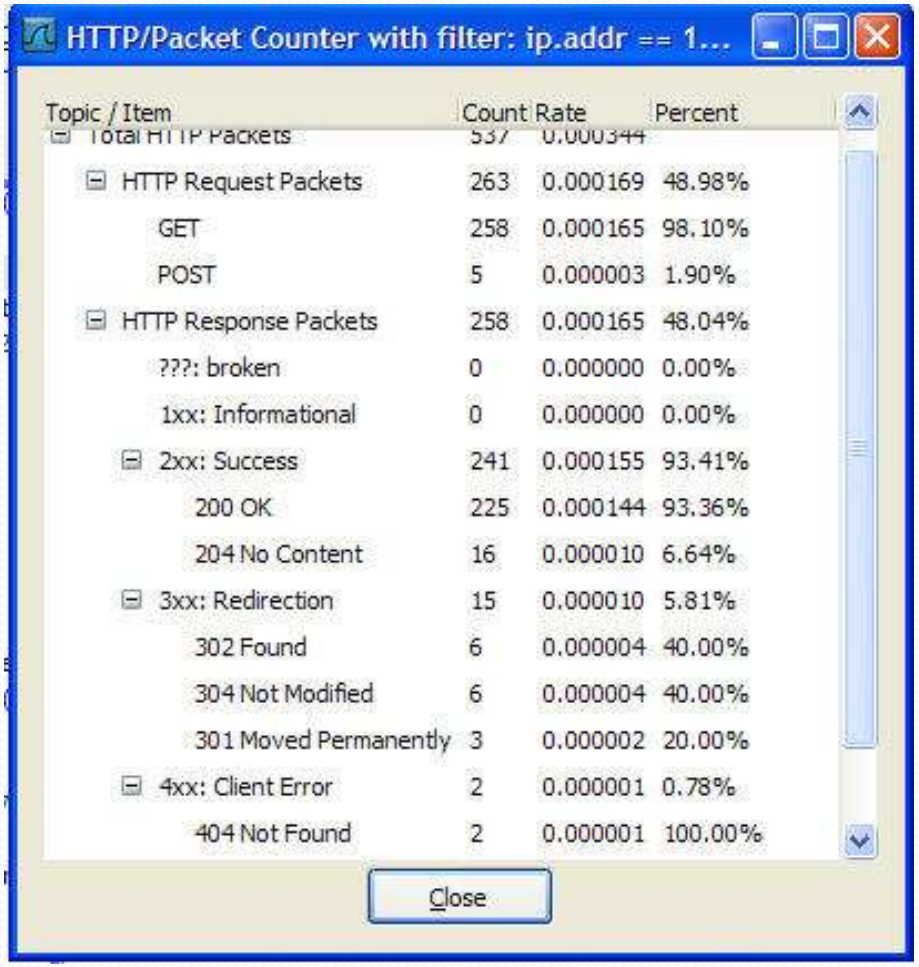
Google Search Queries

- In the HTTP Requests filter, enter:
  - TCP contains google
- The resulting list will display all requests sent to Google. From these calls, you can determine what your clients are searching for.



HTTP Packet Counter

- HTTP Responses can be determined using the HTTP Packet Counter. Request types such as GET or POST are listed, along with the response codes.
- These are huge when troubleshooting applications using a web front end.
- Look for 4xx client errors and 5xx server errors. These will impact the application and may be the root cause under client disconnects and other problems.



Topic / Item	Count	Rate	Percent
Total HTTP Packets	537	0.000344	
HTTP Request Packets	263	0.000169	48.98%
GET	258	0.000165	98.10%
POST	5	0.000003	1.90%
HTTP Response Packets	258	0.000165	48.04%
???: broken	0	0.000000	0.00%
1xx: Informational	0	0.000000	0.00%
2xx: Success	241	0.000155	93.41%
200 OK	225	0.000144	93.36%
204 No Content	16	0.000010	6.64%
3xx: Redirection	15	0.000010	5.81%
302 Found	6	0.000004	40.00%
304 Not Modified	6	0.000004	40.00%
301 Moved Permanently	3	0.000002	20.00%
4xx: Client Error	2	0.000001	0.78%
404 Not Found	2	0.000001	100.00%

Time for Trace Files

Command Line Utilities

- **T shark** - terminal version of Wireshark for capturing and displaying packets when a GUI isn't necessary or available.
- **Tcpdump** - remote capture and do not want the network load associated with running Wireshark remotely
- **Editcap** - remove packets, convert files from one format to another, and print information.
- **Mergecap** – allows multiple files to be merged
- **Capinfos** - utility to print information about binary .cap files
- **Text2cap** - reads in an ASCII hex dump and writes the data into a libpcap-style .cap file
- **Dumpcap** - captures packet data from a live network and writes the packets to a file
- **Rawshark** - reads a stream of packets from a file or pipe, and prints a line describing its output, followed by a set of matching fields for each packet on stdout

Tshark Basics

- Tshark is the command line version of Wireshark
- It allows packets to be captured without opening the Wireshark GUI
- It is installed along with Wireshark
- Tshark can be accessed by navigating to the ***[\bin\...]*** directory
- If you type 'tshark' into the command line and hit enter, tshark will automatically start capturing on the first interface ID
- In order to specify a different interface, using Tshark switches is necessary. The switch to use a different interface is the –i switch.
- Select interface number 2 to try capturing on the second interface
- Try the interfaces on your laptop until you see the copper NIC that is connected to the network.
- You will know when you see summarized traffic flying by on the command line.

Tshark Basics

- Traffic goes by way too fast to analyze, it needs to be captured into a trace file and saved.
- To do this use the following switches.
- -b filesize:64000
  - This switch collects the packets into 64MB files
- -b files:100
  - This switch will cause tshark to capture files only. Then it will start overwriting the oldest ones
- -w d:\data\trace.pcap
  - This is the name and location of the files. Set it to a place you want on your Laptop.
- The final command line entry will be:
  - (replace the interface number with the one appropriate to your laptop, as well as the file destination location)

Tshark –i 3 –b filesize:64000 –b files:100 –w d:\data\traces.pcap

Now these files can be opened, analyzed, and filtered using the regular Wireshark GUI.