

# **T2-5 Advanced Capture and Display Filtering**

April 1, 2008

**Tony Fortunato**

Sr Network Specialist | The Technology Firm

**SHARKFEST '08**

Foothill College

March 31 - April 2, 2008

# About your Presenter

Tony Fortunato, Sr Network Specialist, The Technology Firm

Certified Fluke Networks and Wireshark Instructor

Website: [www.thetechfirm.com](http://www.thetechfirm.com)

A Senior Network Specialist with experience in performance testing, network design, implementation, and troubleshooting LAN/WAN/Wireless networks, desktops and servers since 1989.

Tony has taught at Colleges/Universities, Network/Interop and many onsite corporate settings to thousands of analysts.

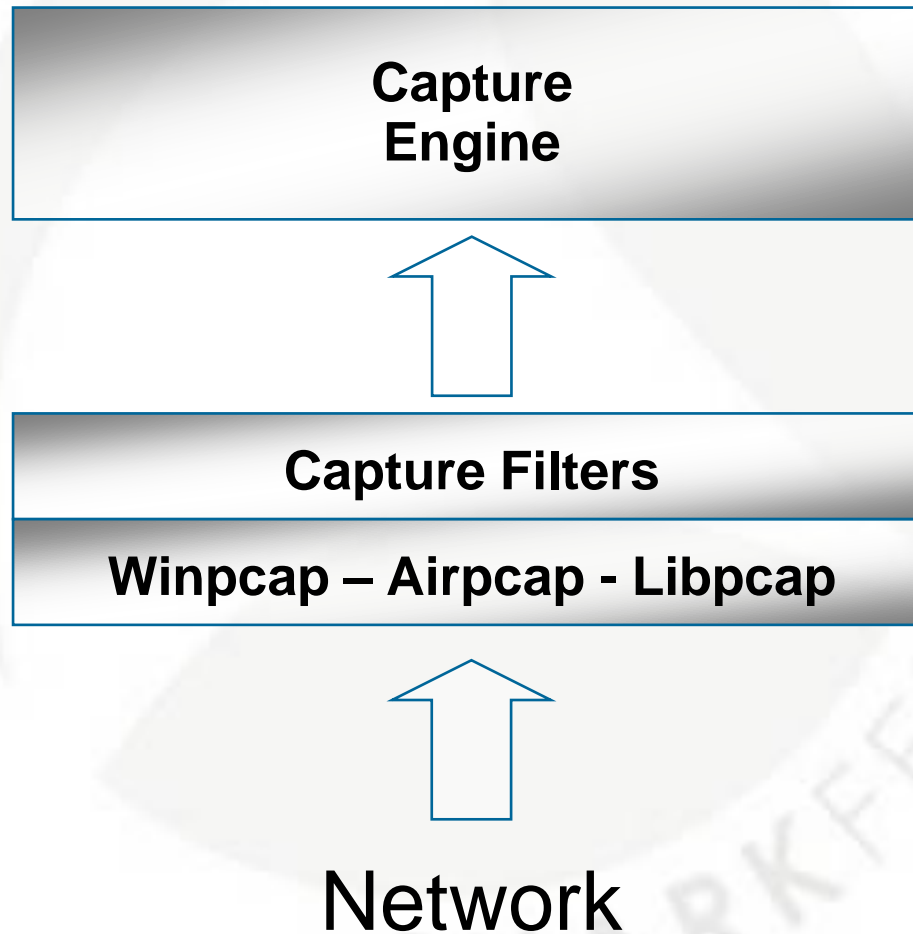
Tony is an authorized and certified Fluke Networks and Wireshark Instructor. His Pine Mountain Group CNA Level I and II certification demonstrates his vendor neutral approach to network design, support and implementations.

Tony has architected, installed and supported various types of Residential Wireless High Speed as well as hundreds of WIFI hotspots.. Tony combines custom programs, open source and commercial software to ensure a simple support infrastructure.

Tony works on networks from 2 to 120,000 nodes and specializes in post installation performance/design review. This process involves using various tools (Protocol analyzers, traffic generators and network management) and working on multi-vendor equipment (switches, routers, servers, etc).

Tony works at customer sites within a range of capacities from project management, network design, consulting, troubleshooting, designing customized courses and assisting with installing physical equipment.

# Capturing Traffic



# Options

When capturing with Wireshark, you have 2 options;

- Using the GUI
- Using the command prompt and *tshark*
  - *tshark -D*
  - *tshark -i*

# Capture Filters

- Based on the tcpdump format
- Location identified in Help -> About under Folders Tab

The image shows two overlapping windows. The top window is the 'About Wireshark' dialog box, with the 'Folders' tab selected. It contains a table with the following data:

Name	Folder
"File" dialogs	C:\Documents and Settings\Tony Fortunato\Desktop\ring files\
Temp	C:\DOCUMENTE~1\TONYFO~1\LOCALS~1\Temp\
Personal configuration	C:\Documents and Settings\Tony Fortunato\Application Data\Wireshark\
Global configuration	C:\Program Files\Wireshark\

The bottom window is a Windows Explorer window showing the folder 'C:\Documents and Settings\Tony Fortunato\Application Data\Wireshark'. The file list contains:

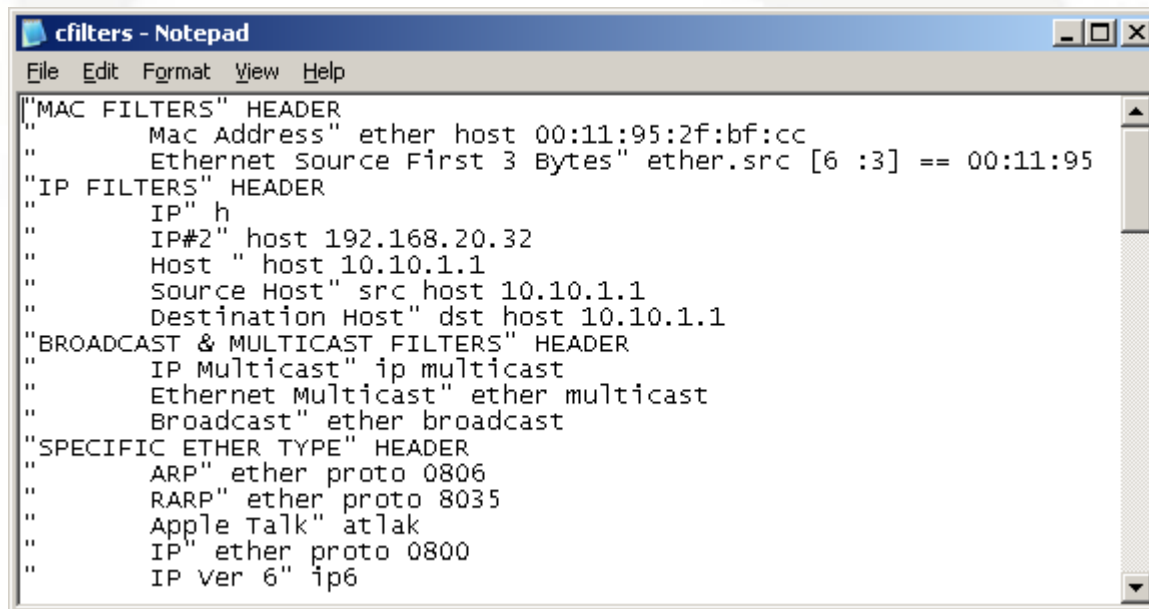
- cfilters (File, 3 KB)
- colorfilt (File, 2 KB)
- dfilters (File, 1 KB)
- recent (File, 9 KB)
- Copy of cfilters
- preferences (File, 114 KB)

Two blue callout boxes with arrows point to specific files:

- A box labeled 'Capture Filter' points to the 'cfilters' file.
- A box labeled 'Display Filter' points to the 'dfilters' file.

# cfilters file notes

- Make sure you use a text editor, or save in a text format
- Ensure you have a blank line at the end of the file
- Good idea to create a header and indent the related filters



```
cfilters - Notepad
File Edit Format View Help
"MAC FILTERS" HEADER
"   Mac Address" ether host 00:11:95:2f:bf:cc
"   Ethernet Source First 3 Bytes" ether.src [6 :3] == 00:11:95
"IP FILTERS" HEADER
"   IP" h
"   IP#2" host 192.168.20.32
"   Host " host 10.10.1.1
"   Source Host" src host 10.10.1.1
"   Destination Host" dst host 10.10.1.1
"BROADCAST & MULTICAST FILTERS" HEADER
"   IP Multicast" ip multicast
"   Ethernet Multicast" ether multicast
"   Broadcast" ether broadcast
"SPECIFIC ETHER TYPE" HEADER
"   ARP" ether proto 0806
"   RARP" ether proto 8035
"   Apple Talk" atlak
"   IP" ether proto 0800
"   IP Ver 6" ip6
```

# Capture Filter Reference

Command	Description
<b>ether host</b> <i>MAC address</i>	Capture all packets to and from a <i>MAC address</i>
<i>IP Filters</i>	
<b>host</b> <i>ip address</i>	Capture all packets to and from an <i>ip address</i>
<b>src host</b> <i>ip address</i>	Capture all packets from an <i>ip address</i>
<b>dst host</b> <i>ip address</i>	Capture all packets to an <i>ip address</i>
<i>TCP/UDP Filters</i>	
<b>port</b> <i>port</i>	Capture all packets to and from a port number
<b>src port</b> <i>port</i>	Capture all packets from a port number
<b>dst port</b> <i>port</i>	Capture all packets to a port number
<i>IP Network Filters</i>	
<b>net</b> <i>net</i>	Capture all packets to and from a <i>subnet</i>
<b>src net</b> <i>net</i>	Capture all packets from a <i>subnet</i>
<b>dst net</b> <i>net</i>	Capture all packets to a <i>subnet</i>

# Capture Filter Examples

Command	Description
<b>ether host 00:15:c5:37:40:60</b>	Capture all packets to and from MAC 00:15:c5:37:40:60
<i>IP Filters</i>	
<b>host 10.44.10.1</b>	Capture all packets to and from 10.44.10.1
<b>host www.wireshark.org</b>	Capture all packets from www.wireshark.org
<i>TCP/UDP Filters</i>	
<b>port 80</b>	Capture all packets to and from TCP/UDP port number 80
<b>portrange 67-68</b>	Capture all DHCP bootps/bootpc
<b>port http</b>	Capture all packets from devices using http
<b>tcp portrange 1200-2000</b>	Capture all packets with TCP port # 1200-2000
<i>IP Network Filters</i>	
<b>net 10.44.10</b>	Capture all packets to and from a subnet 10.44.10
<b>arp</b>	Capture all arp packets
<b>udp</b>	Capture all udp packets
<b>tcp</b>	Capture all tcp packets

# Supported Capture Protocols

- *arp Address Resolution Protocol*
- *esp Encapsulating Security Payload*
- *icmp Internet Control Message Protocol*
- *icmp6 Internet Control Message Protocol, for IPv6*
- *igmp Internet Group Management Protocol*
- *igrp Interior Gateway Routing Protocol*
- *ip Internet Protocol*
- *ip6 Internet Protocol version 6*
- *pim Protocol Independent Multicast*
- *rarp Reverse Address Resolution Protocol*
- *stp Spanning Tree Protocol*
- *tcp Transmission Control Protocol*
- *udp User Datagram Protocol*
- *vrrp Virtual Router Redundancy Protocol*

# Data Pattern Offsets

To retrieve a single byte from a packet, use square brackets to indicate the offset of that byte from the beginning of a particular protocol. Offsets start at zero (e.g., tcp[0] gives the first byte in the TCP header and tcp[1] gives the second byte)

TCP Header Layout

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	26	27	28	29	30	31	32
Source Port																Destination Port															
Sequence Number																															
Acknowledgment Number																															
Data Offset		Reserved				U R G	A C K	P S H	R S T	S Y N	F I N	Windows																			
Checksum																Urgent Pointer															
Options																								Padding							
Data																															

# HTTP Get Offset Example

No. -	Time	Length B	Source	Destination
2932	01:26:19	367	69.181.134.143	63.147.175.35
2933	01:26:19	373	69.181.134.143	63.147.175.35

Internet Protocol	Control Protocol	Src	Dst
Internet Protocol	Control Protocol	69.181.134.143 (69.181.134.143)	63.147.175.35 (63.147.175.35)

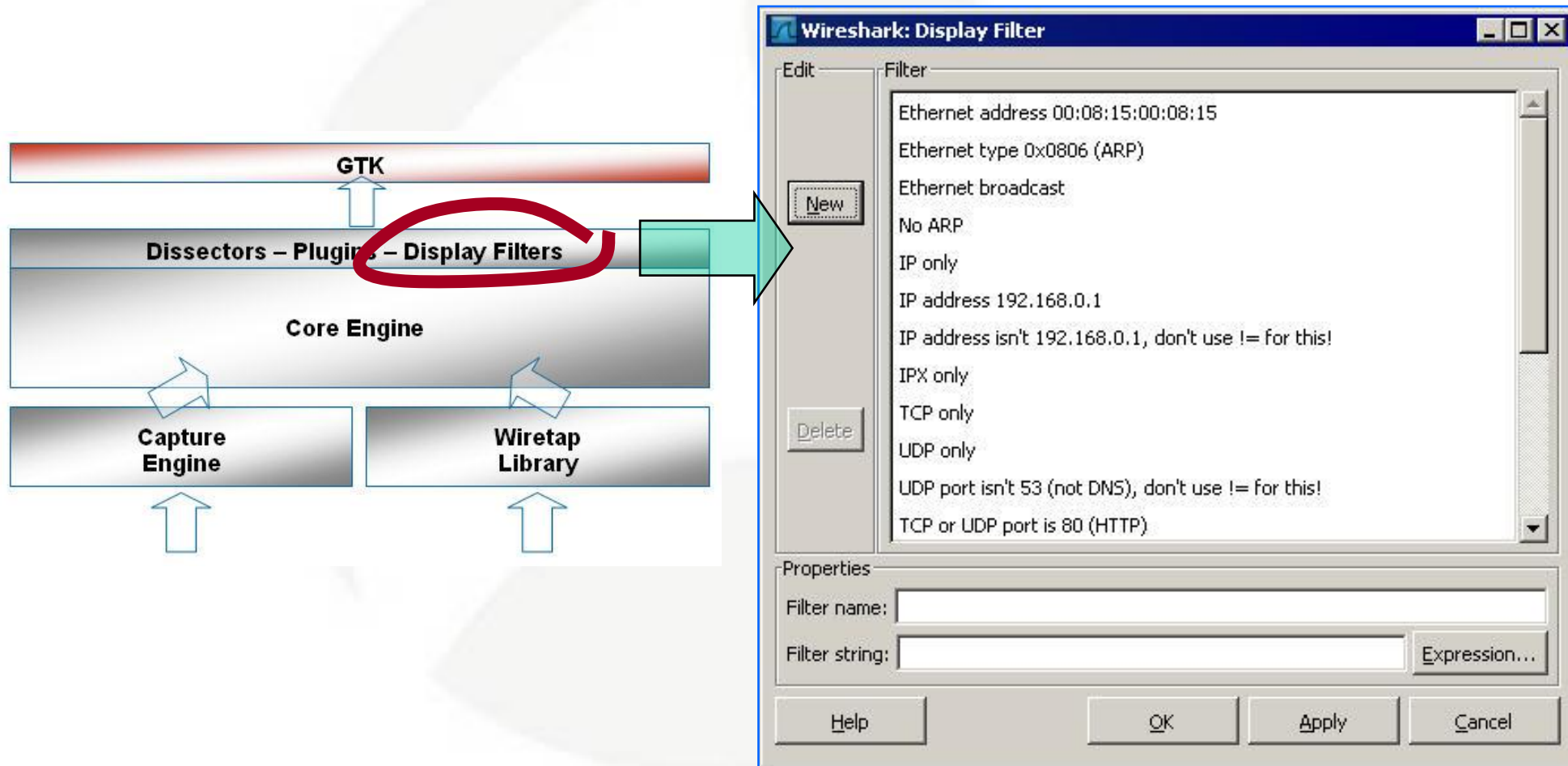
Hypertext Transfer Protocol	Offset	Hex	ASCII
Hypertext Transfer Protocol	0000	00 01 5c 22 a5 82 08 00 46 f4 3a 09 08 00 45 00	..\".... F....E.
	0010	01 62 68 61 40 00 40 06 16 3a 45 b5 86 8f 3f 93	.bha@.@. :E...?
	0020	af 23 0c 99 00 50 02 90 e5 5c e2 09 46 22 50 18	.#...P.. \\.F\"P.
	0030	fa f0 38 45 00 00 47 45 54 20 2f 70 68 6f 74 6f	.8E..GE T /photo
	0040	2f 32 30 30 36 2f 31 31 31 34 2f 6e 66 6c 5f 63	/2006/11 14/nfl_c
	0050	6f 61 63 68 65 73 5f 31 33 34 2e 6a 70 67 20 48	oaches   34.jpg H

TCP Offset 20

4 Bytes of Data

# Overview of Display Filters

- Can visit <http://www.wireshark.org/docs/dfref>



# Display Filter Syntax

Visit [www.wireshark.org](http://www.wireshark.org) for the master list of Display Filter field names, types, descriptions and versions

## Display Filter Reference

Wireshark's most powerful feature is its vast array of display filters (over 51000 as of version 0.99.5). They let you drill drill down to the exact traffic you want to see and are the basis of many of Wireshark's other features, such as the coloring rules.

This is a reference. If you need help using display filters, please see the [wireshark-filter](#) and the [User's Guide](#).

## Index

[2](#) [3](#) [9](#) [A](#) [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [J](#) [K](#) [L](#) [M](#) [N](#) [O](#) [P](#) [Q](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#) [X](#) [Y](#) [Z](#)

## Display Filter Reference: Transmission Control Protocol

**Protocol field name:** tcp

**Versions:** 0.10.0 to 0.99.5

[Back to Display Filter Reference](#)

Field name	Type	Description	Versions
tcp.ack	Unsigned 32-bit integer	Acknowledgement number	0.10.0 to 0.99.5
tcp.analysis.ack_lost_segment	None	ACKed Lost Packet	0.10.0 to 0.99.5
tcp.analysis.ack_rtt	Time duration	The RTT to ACK the segment was	0.10.0 to 0.99.5
tcp.analysis.acks_frame	Frame number	This is an ACK to the segment in frame	0.10.0 to 0.99.5
tcp.analysis.duplicate_ack	None	Duplicate	0.10.0 to

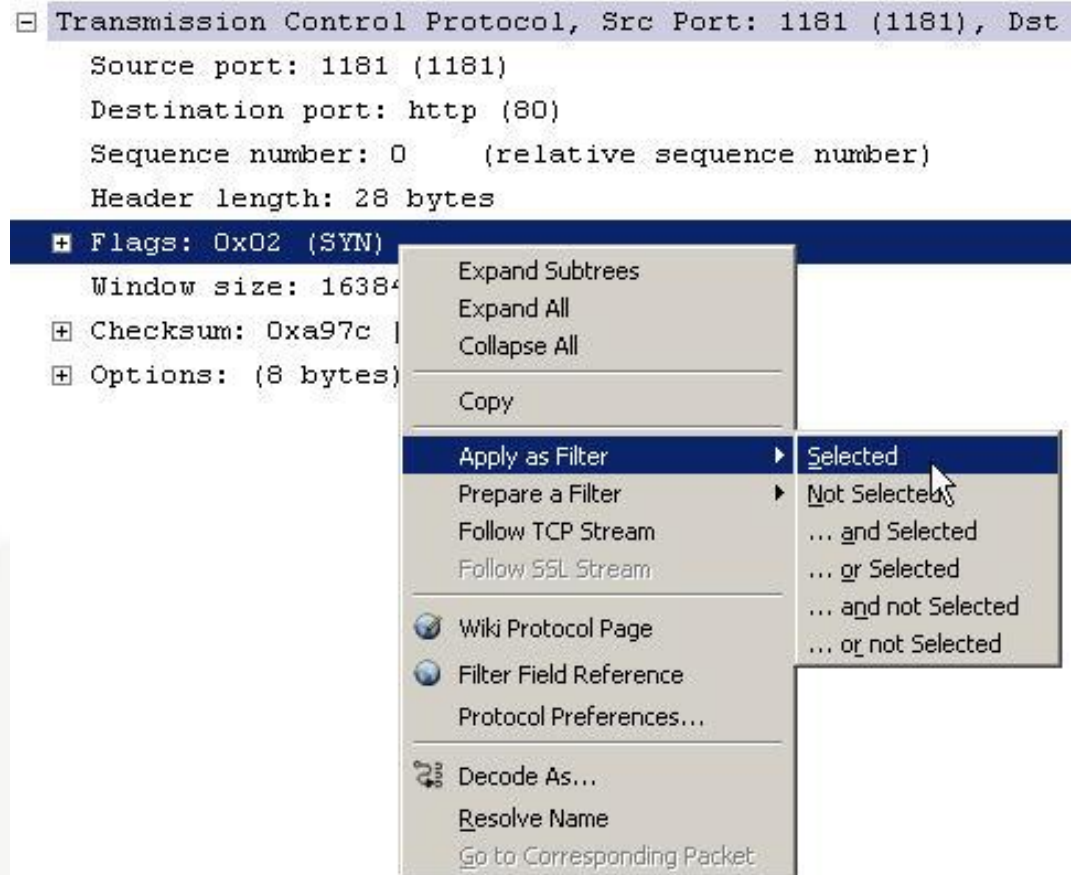
# Operators and Advanced Filters

## Operators

<code>==</code>	or	<code>eq</code>	equal to
<code>  </code>	or	<code>or</code>	or
<code>&gt;</code>	or	<code>gt</code>	greater than
<code>&gt;=</code>	or	<code>ge</code>	greater than or equal to
<code>&lt;=</code>	or	<code>le</code>	less than or equal to
<code>&lt;</code>	or	<code>lt</code>	less than
<code>!</code>	or	<code>not</code>	not
<code>!=</code>	or	<code>ne</code>	not equal to
<code>contains</code>			
<code>matches</code>			

# Build Filters Based on Captured Packet

Right mouse click on any field and either **Apply** or **Prepare** a filter based on the field and value (with an implied 'equal to' operator).



# Build Filters Based on Expressions

**Field Name**                      **Relation**                      **Value**

Field name

- FractalGeneratorProtocol
- Frame
- FRSAPI
- FRSRPC
- FTAM
- FTBP
- FTP
  - ftp.response - Response (TRUE if FTP response)
  - ftp.request - Request (TRUE if FTP request)
  - ftp.request.command - Request command
  - ftp.request.arg - Request arg
  - ftp.response.code - Response code**
  - ftp.response.arg - Response arg

Relation

- is present
- ==
- !=
- >
- <
- >=
- <=

Value (unsigned, 4 bytes)

227

Predefined values:

- Help message
- NAME system type
- Service ready for new user
- Service closing control connection
- Data connection open; no transfer
- Closing data connection
- Entering Passive Mode**
- User logged in, proceed
- Requested file action okay, file

Range (offset:length)

OK      Cancel

**Predefined Values**

# Build Filters from Statistics Reports

I use the 'Prepare a Filter' to build my filter

The screenshot shows two overlapping windows from Wireshark. The top window is titled 'Conversations: novotel full capture.cap' and displays a table of IPv4 conversations. The bottom window is titled 'Tony Fortunato - novotel full capture.cap - Wireshark' and shows the main interface with a filter applied.

**IPv4 Conversations Table:**

Address A	Address B	Packets	Bytes	Packets A->B	Bytes A->B	Packets A<-B	Bytes A<-B	Rel Start	Dur
10.0.1.200	128.46.156.117	33	78126	1966	2976524	0	0	0.184944000	425
10.0.1.245	239.255.255.250	36	15486	0	0	0	0	276.284156999	6.7
10.0.1.239	239.255.255.250	29	0	0	0	0	0	121.839433999	197
10.0.1.230	10.255.255.255	10	0	0	0	0	0	2.902963000	415
10.0.1.247	224.0.0.251	10	0	0	0	0	0	50.612903999	347
10.0.1.205	239.255.255.250	18	0	0	0	0	0	270.685227999	8.4
10.0.1.243	10.255.2	0	0	0	0	0	0		
10.0.1.200	72.14.21	0	0	0	0	0	0		
10.0.1.200	10.0.1.2	0	0	0	0	0	0		
10.0.1.239	255.255.	0	0	0	0	0	0		
10.0.1.230	224.0.0.	0	0	0	0	0	0		
10.0.1.239	10.255.2	0	0	0	0	0	0		
10.0.1.9	10.255.2	0	0	0	0	0	0		
10.0.1.52	10.255.2	0	0	0	0	0	0		
10.0.2.0	10.255.2	0	0	0	0	0	0		

**Filter:** `ip.addr==10.0.1.200 && ip.addr==128.46.156.117`

**Packet List:**

No.	Time	Length	Source	Destination	Protocol	Info
1	0.000000	250	10.0.1.219	10.255.255.255	BROWSE	Domain/Workgroup Annou...
2	0.184944	1514	128.46.156.117	10.0.1.200	TCP	53186 > 1106 [ACK] Seq=1
3	0.007969	1514	128.46.156.117	10.0.1.200	TCP	53186 > 1106 [ACK] Seq=1
4	0.000032	54	10.0.1.200	128.46.156.117	TCP	1106 > 53186 [ACK] Seq=1
5	0.007989	1514	128.46.156.117	10.0.1.200	TCP	53186 > 1106 [ACK] Seq=2
6	0.195927	54	10.0.1.200	128.46.156.117	TCP	1106 > 53186 [ACK] Seq=1

# Top 10 Useful Filters

IP Address:	<code>ip.addr == x.x.x.x</code>
MAC Address:	<code>eth.addr == xx:xx:xx:xx:xx:xx</code>
ICMP!	<code>icmp</code>
My MAC Address:	<code>eth.addr == xx:xx:xx:xx:xx:xx</code>
DHCP:	<code>bootp</code>
High Delta Time:	<code>frame.time_delta &gt; 1</code>
TCP Port:	<code>tcp.port == x</code>
UDP Port:	<code>udp.port == x</code>
TCP ACK RTT:	<code>tcp.analysis.ack_rtt &gt; 1</code>
TCP Length: > x < y	<code>tcp.len &gt; x &amp;&amp; tcp.len &lt; y</code>

## **Bonus**

Not My MAC: `!eth.addr == xx:xx:xx:xx:xx:xx`

# Manually Editing the *dfilters* File

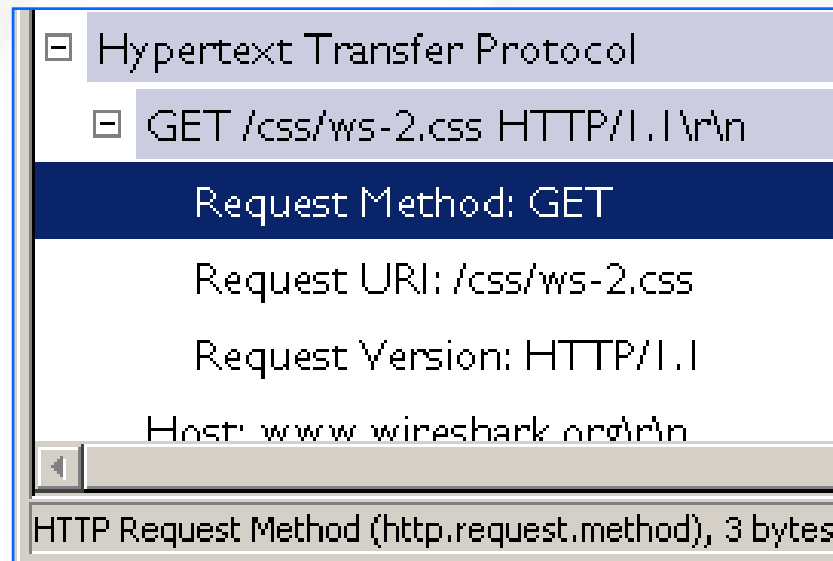
```
"Ethernet address 00:08:15:00:08:15" eth.addr == 00:08:15:00:08:15
"Ethernet type 0x0806 (ARP)" eth.type == 0x0806
"Ethernet broadcast" eth.addr == ff:ff:ff:ff:ff:ff
"No ARP" not arp
"IP only" ip
"IP address 192.168.0.1" ip.addr == 192.168.0.1
"IP address isn't 192.168.0.1, don't use != for (192.168.0.1)"
"IPX only" ipx
"TCP only" tcp
"UDP only" udp
"UDP port isn't 53 (not DNS), don't use != for (53)"
"TCP or UDP port is 80 (HTTP)" tcp.port == 80 || udp.port == 80
"HTTP" http
"No ARP and no DNS" not arp and !(udp.port == 53)
"Non-HTTP and non-SMTP to/from 192.168.0.1" not (tcp.port == 25) and ip.addr == 192.168.0.1
"Macof window=512" tcp.window_size == 512
"ICMP type 8 code not 0" (icmp.type == 8) && !(icmp.code == 0)
"Ping code not 0" (icmp.type == 8) && !(icmp.code == 0)
```

Do not append a file extension

Include blank line after last filter

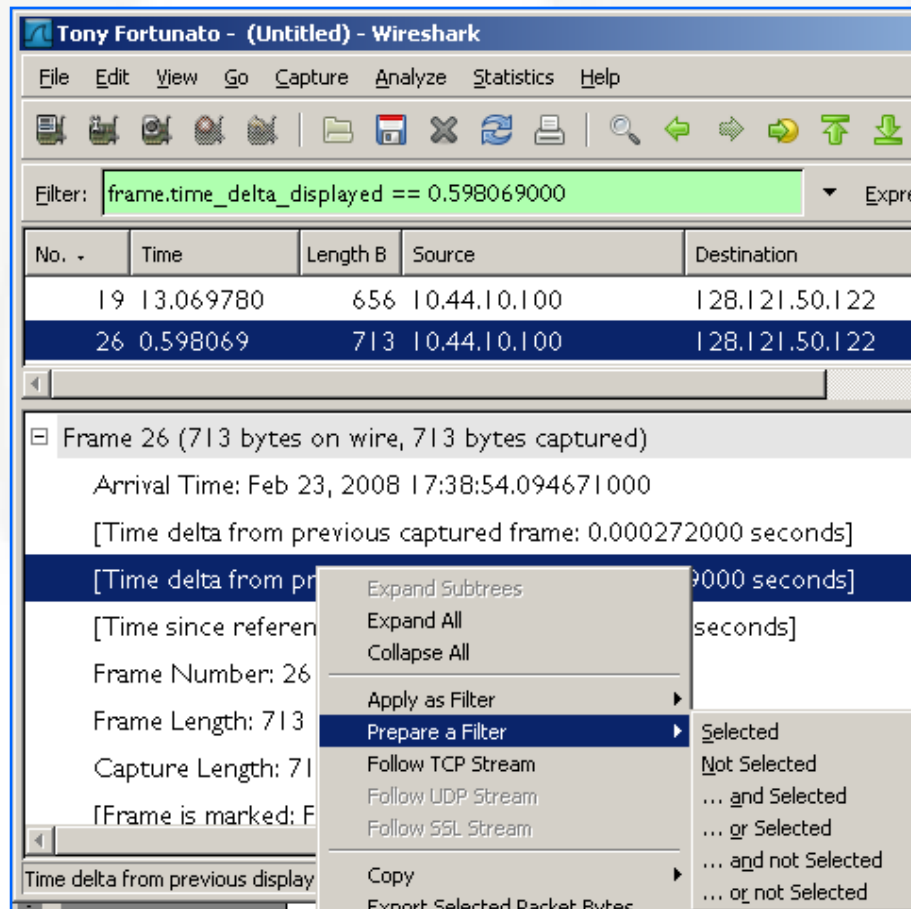
# Display Filters

The easiest way to learn is to look at existing traces and reference the field name you are interested in;



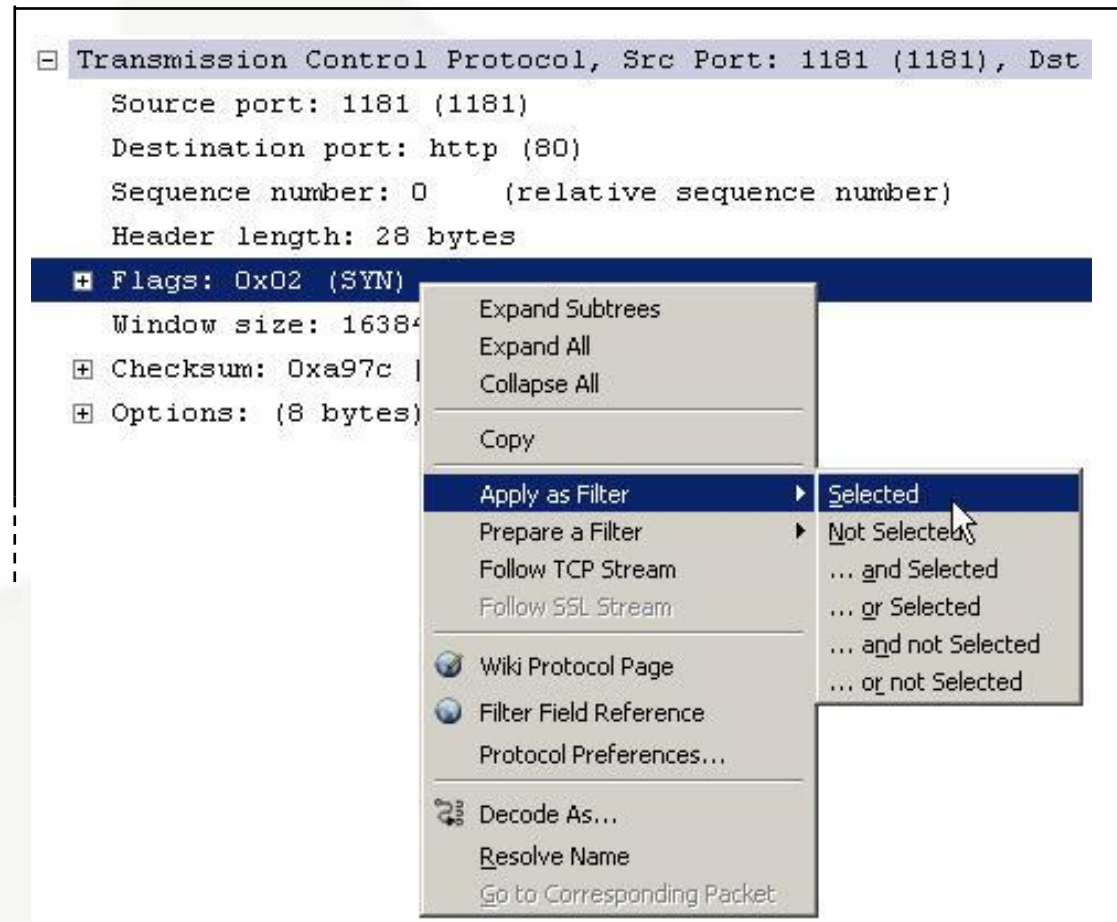
# Display Filter Time Saver

When I want to build a filter, but don't want to type out a long fieldname, I simply right click on the field name and use the Prepare A Filter-> Selected and then modify the filter from there.

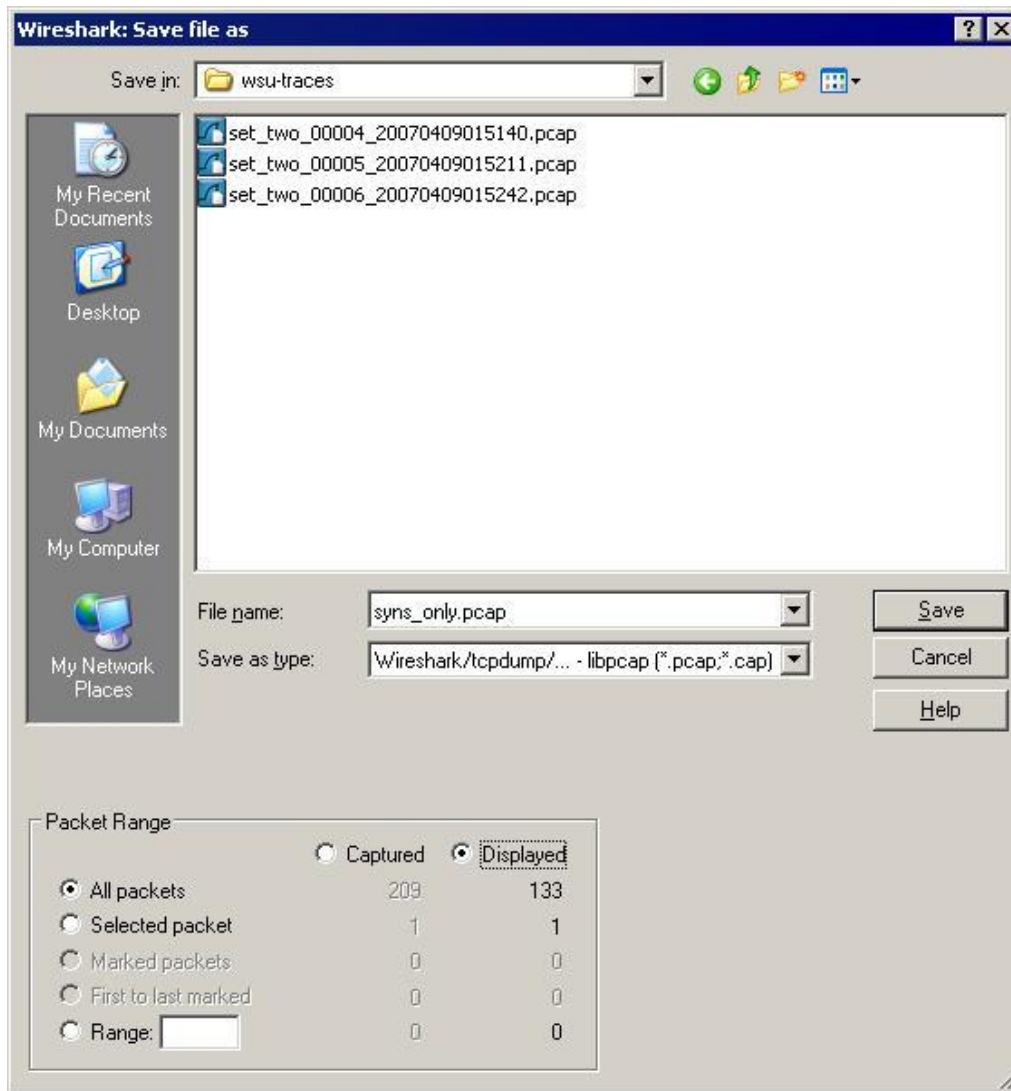


# Build Filters Based on Packets

Right mouse click on any field and either **Apply** or **Prepare** a filter based on the field and value (with an implied 'equal to' operator).



# Save Filtered, Marked and Ranges of Packets



## Packet Range Selection

- Captured/Displayed
- All packets
- Selected packets
- Marked packets
- First to last marked packet
- Range

# Follow the Stream

This feature creates a display filter of the selected Packet's IP address and port pairs

4 10.10.10.22 128.121.50.122 TCP 3695 > 80 [ACK] Seq=1 A

22 HTTP GET /lists/ HTTP/1.1

Mark Packet (toggle)

Set Time Reference (toggle)

Apply as Filter

Prepare a Filter

Conversation Filter

SCTP

Follow TCP Stream

Follow UDP Stream

Follow SSL Stream

Copy

Export Selected Packet Bytes...

Decode As...

Print...

Show Packet in New Window

Tony Fortunato - (Untitled) - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: (ip.addr eq 10.10.10.22 and ip.addr eq 128.121.50.122) and (tcp.p...

Follow TCP Stream

Stream Content

GET /lists/ HTTP/1.1

Host: www.wireshark.org

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:...

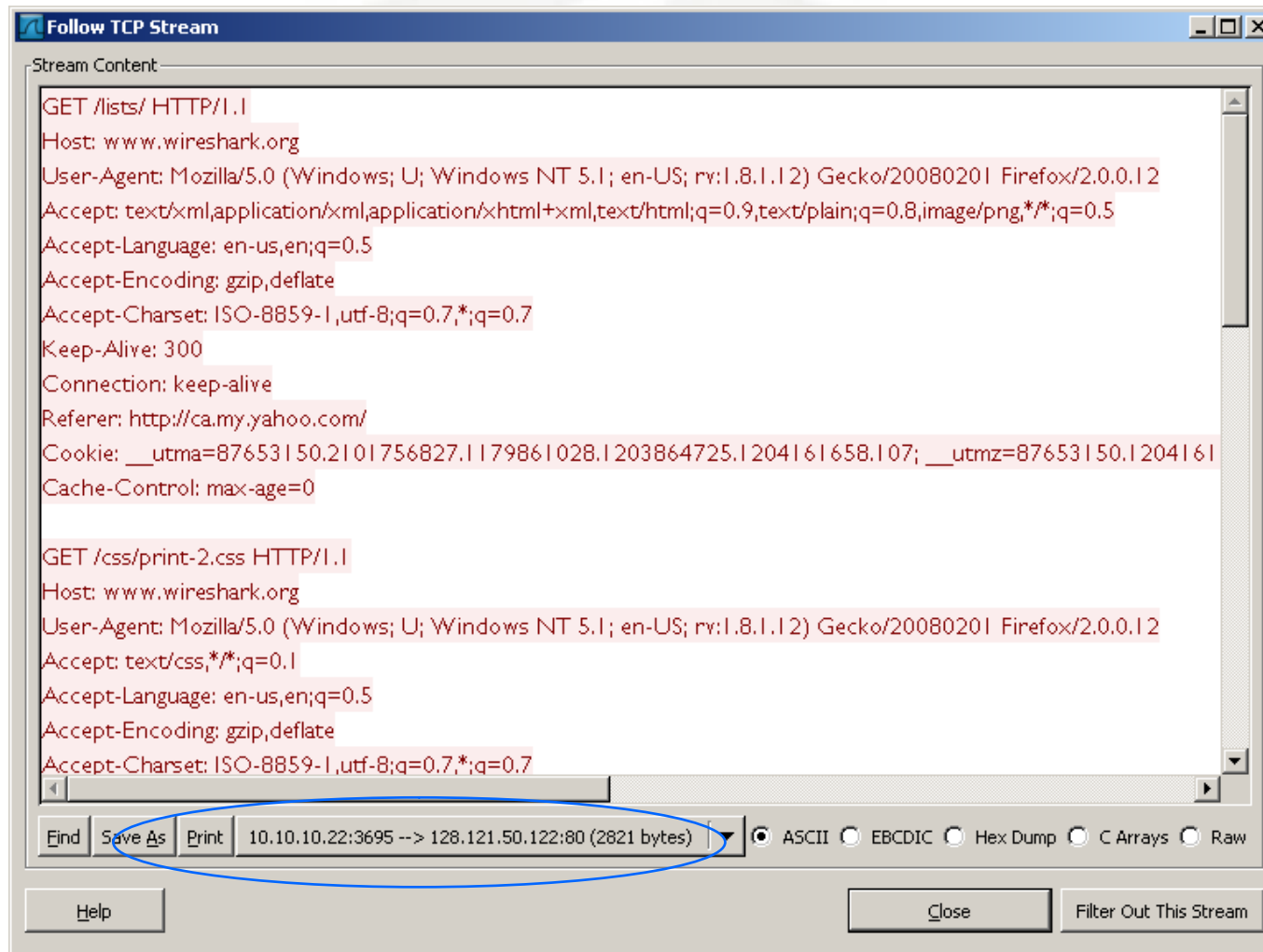
Accept: text/xml,application/xml,application/xhtml+xml;text/html;q=...

Accept-Language: en-us,en;q=0.5

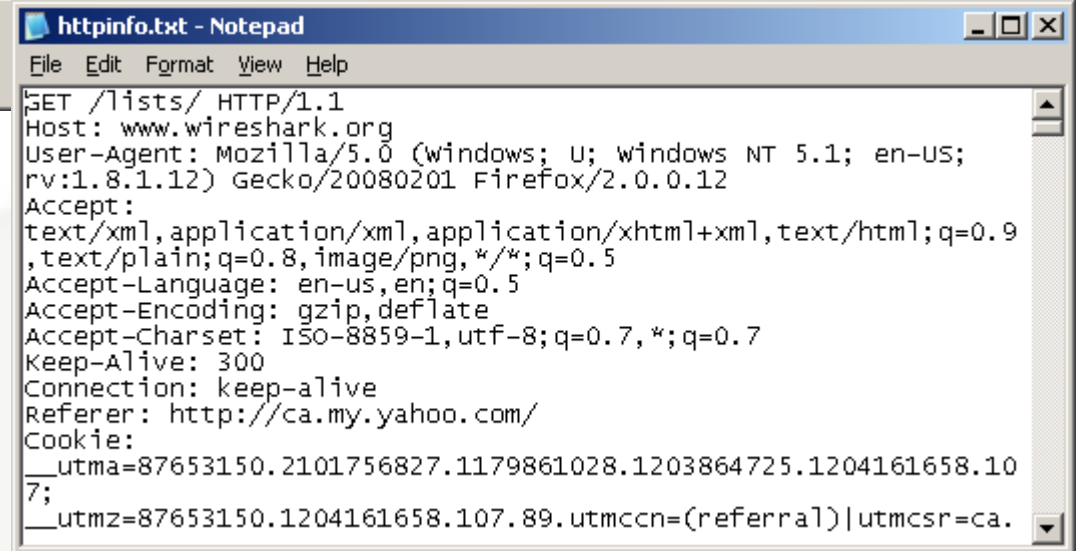
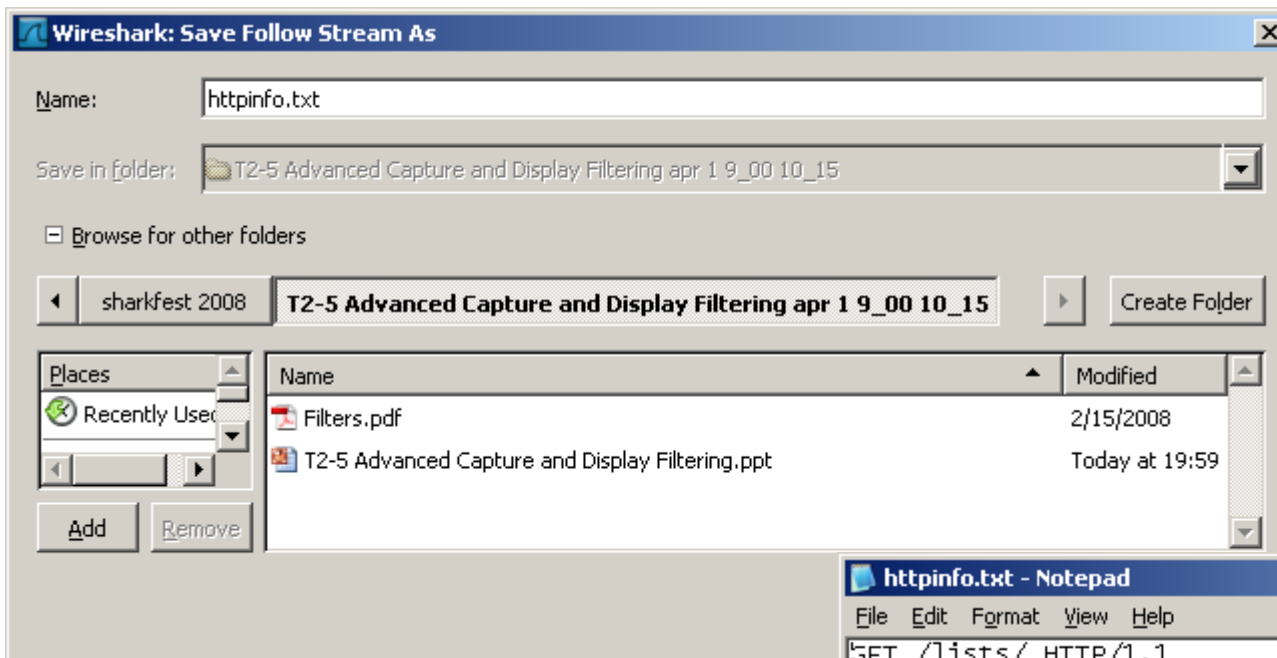
Packets: 1546 Displayed: 28 Marked: 0 Dropped: 0

# One Way stream

You can select the stream data from the client or Server



# Saving A Stream



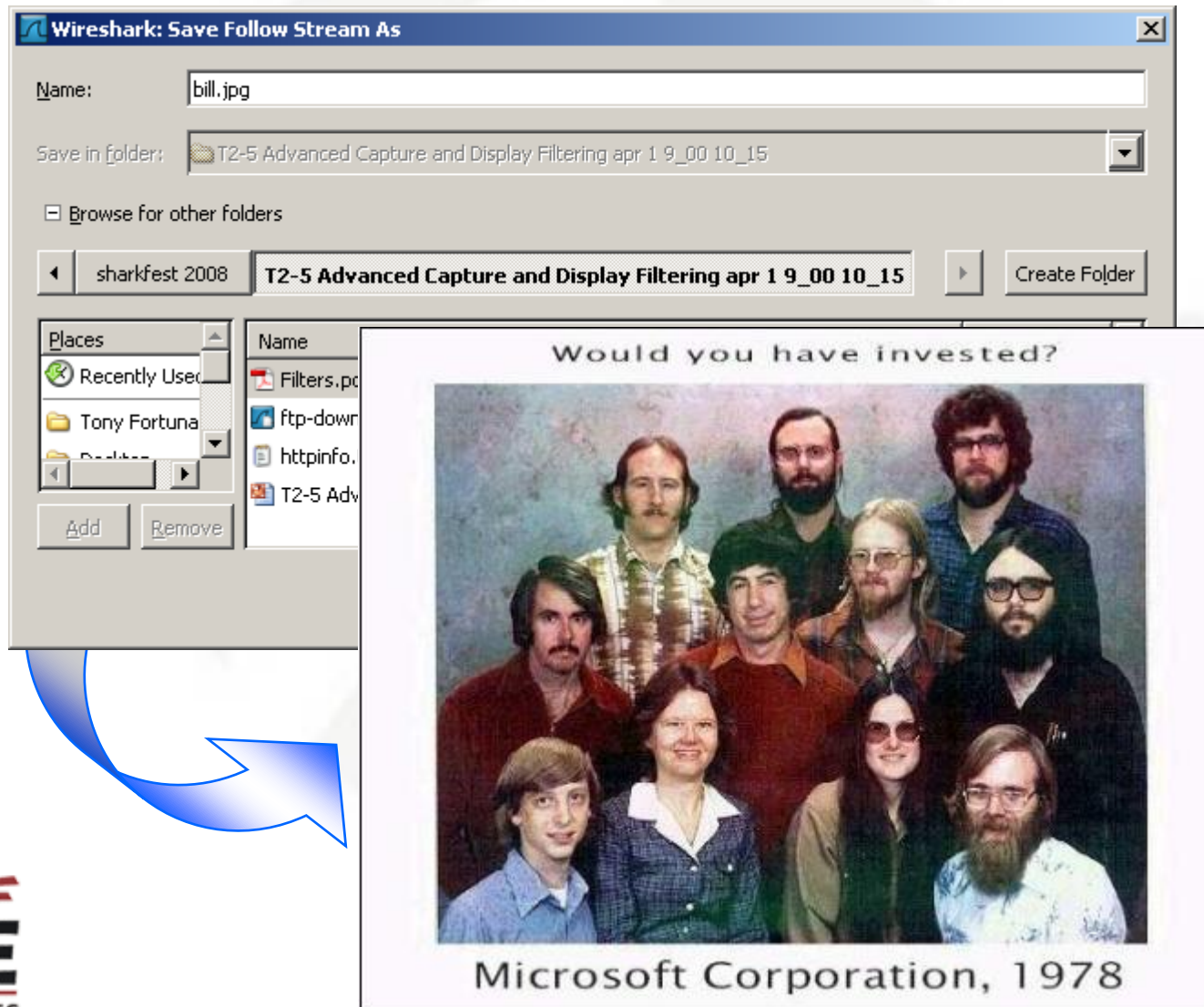
The image shows a Notepad window titled 'httpinfo.txt - Notepad'. The text content is as follows:

```
File Edit Format View Help
GET /lists/ HTTP/1.1
Host: www.wireshark.org
User-Agent: Mozilla/5.0 (windows; U; windows NT 5.1; en-US;
rv:1.8.1.12) Gecko/20080201 Firefox/2.0.0.12
Accept:
text/xml,application/xml,application/xhtml+xml,text/html;q=0.9
,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://ca.my.yahoo.com/
Cookie:
__utma=87653150.2101756827.1179861028.1203864725.1204161658.10
7;
__utmz=87653150.1204161658.107.89.utmccn=(referral)|utmcsr=ca.
```

# Rebuilding a file

ftp-download-good.pcap

You can rebuild files from the stream



# HTTP Filter Example

Common filters for HTTP

```
http.request.version == "HTTP/1.1"
```

Will return commands and responses