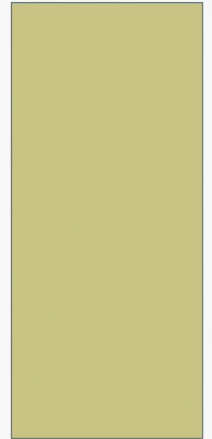


# USING WIRESHARK TO CAPTURE AND ANALYZE NETWORK DATA

CPSC 441 TUTORIAL – JANUARY 30, 2012  
TA: MARYAM ELAHI



The content of these slides are taken from CPSC 526 TUTORIAL by Nashd Safa  
(Extended and partially modified)

# WIRESHARK

- **Wireshark** (Originally named Ethereal) is a free and open-source packet analyzer
- It is used for network troubleshooting, analysis, software and communication protocol development, and education.
- It has a graphical front-end, and many more information sorting and filtering options.

# FEATURES AND FUNCTIONALITIES OF WIRESHARK

- **Wireshark** is software that "understands" the structure of different **networking protocols**. Thus, it is able to display the encapsulation and the fields along with their meanings of different packets specified by different networking protocols.
- Live data can be read from a number of types of network, including Ethernet, IEEE 802.11, PPP...
- Data display can be refined using a display filter.

# INSTALLING WIRESHARK

- Download Wireshark from <http://www.wireshark.org/download.html>
- Choose appropriate version according to your operating system
- (For Windows), during installation agree to install **winpcap** as well.
  - **pcap** (packet capture) is an application programming interface (API) for capturing network traffic. Unix-like systems implement pcap in the libpcap library. Windows uses a port of libpcap known as **WinPcap**.
- <http://wiki.wireshark.org/CaptureSetup>  
Provides a good tutorial on how to capture data using WireShark

# BEFORE CAPTURING DATA

- **Are you allowed to do this?**
- Ensure that you have the permission to capture packets from the network you are connected with. (Corporate policies or applicable law might prohibit capturing data from the network)
- **General Setup**
- Operating system must support packet capturing, e.g. capture support is enabled
- You must have sufficient privileges to capture packets, e.g. root / Administrator privileges
- Your computer's time and time zone settings should be correct

# CAPTURING DATA

- The available network interfaces are listed here.

The screenshot displays the Wireshark Network Analyzer interface. The main window is titled "The Wireshark Network Analyzer [Wireshark 1.6.5 (SVN Rev 40429 from /trunk-1.6)]". The interface is divided into several sections:

- Capture:** Contains the "Interface List" section, which shows a live list of capture interfaces. Below this, there are "Capture Options" and "Capture Help" sections.
- Files:** Contains "Open" (to open a previously captured file), "Open Recent", and "Sample Captures" (a rich assortment of example capture files on the wiki).
- Online:** Contains links to the "Website", "User's Guide", and "Security" pages.

A red arrow points from the text "The available network interfaces are listed here." to the "Interface List" section. A red banner at the bottom asks "Which interface we want to capture from?" and a green banner below it says "Probably the one that has some traffic." The status bar at the bottom shows "Ready to load or capture" and "No Packets".

# CHOOSING THE INTERFACE

The screenshot shows the Wireshark Network Analyzer interface with the 'Capture Interfaces' dialog box open. The dialog box displays a table of available network interfaces for capture. The second interface, 'Microsoft', is highlighted with a red box, indicating it is the selected interface for capture.

Description	IP	Packets	Packets/s	Stop
Intel(R) 82566MM Gigabit Network Connection	fe80::405e:b889:63a1:4389	0	0	Start Options Details
Microsoft	fe80::a86e:4798:aae2:9b3	430	1	Start Options Details
Microsoft	fe80::2d40:1deb:ff11:2c72	0	0	Start Options Details

Buttons: Help, Close

Bottom status bar: Ready to load or capture, No Packets, Profile: Default

# CAPTURING DATA

- Click on the specific interface you want to capture traffic from.

The screenshot shows the Wireshark interface with a live capture in progress. The packet list pane shows four captured packets:

No.	Time	Source	Destination	Protocol	Info
1	0.000000	67.228.110.120	192.168.0.100	TCP	http > 1232 [FIN, ACK] Seq=1 Ack=1 Win=65 Len=0
2	0.000073	192.168.0.100	67.228.110.120	TCP	1232 > http [ACK] Seq=1 Ack=2 Win=4313 Len=0
3	1.990387	192.168.0.100	67.228.110.120	TCP	1232 > http [FIN, ACK] Seq=1 Ack=2 Win=4313 Len=0
4	2.019462	67.228.110.120	192.168.0.100	TCP	http > 1232 [ACK] Seq=2 Ack=2 Win=65 Len=0

The packet details pane for the selected packet (No. 1) shows:

- Frame 1 (54 bytes on wire, 54 bytes captured)
- Ethernet II, Src: D-Link\_cf:ea:c7 (00:24:01:cf:ea:c7), Dst: HonHaiPr\_77:5d:a1 (00:25:56:77:5d:a1)
- Internet Protocol, Src: 67.228.110.120 (67.228.110.120), Dst: 192.168.0.100 (192.168.0.100)
- Transmission Control Protocol, src Port: http (80), dst Port: 1232 (1232), Seq: 1, Ack: 1, Len: 0

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0000 00 25 56 77 5d a1 00 24 01 cf ea c7 08 00 45 00  .%vw)..$. . . . .E.
0010 00 28 8c a0 40 00 36 06 44 c7 43 e4 6e 78 c0 a8  .(.,@.6. D.C.rX..
0020 00 64 00 50 04 d0 42 4e 7c f5 d3 a4 16 85 50 11  .d.P..BN |.....P.
0030 00 41 8d 9c 00 00  .A....
```



# ANALYZING CAPTURED DATA

Filter:  Expression... Clear Apply

No. -	Time	Source	Destination	Protocol	Info
154	97.803307	192.168.0.100	174.129.27.168	TLSv1	Application Data, Application Data, Application Data,
155	97.805312	192.168.0.100	174.129.27.168	TLSv1	Application Data,
156	97.848793	174.129.27.168	192.168.0.100	TCP	https > bvcontrol [ACK] Seq=1414 Ack=3545 win=16896 Len=0
157	97.848865	192.168.0.100	174.129.27.168	TLSv1	Application Data, Application Data, Application Data,
158	97.848872	192.168.0.100	174.129.27.168	TLSv1	Application Data, Application Data, Application Data,
159	97.890781	174.129.27.168	192.168.0.100	TCP	https > bvcontrol [ACK] Seq=1414 Ack=4993 win=19968 Len=0
160	97.890856	192.168.0.100	174.129.27.168	TCP	[TCP segment of a reassembled PDU]
161	97.890864	192.168.0.100	174.129.27.168	TCP	[TCP segment of a reassembled PDU]
162	97.897797	174.129.27.168	192.168.0.100	TCP	https > bvcontrol [ACK] Seq=1414 Ack=6441 win=23040 Len=0
163	97.897850	192.168.0.100	174.129.27.168	TCP	[TCP segment of a reassembled PDU]

Time of capturing the packet

Source IP

Destination IP

Protocol Name

Brief description of the packet data

# ANALYZING CAPTURED DATA

No. -	Time	Source	Destination	Protocol	Info
154	97.803307	192.168.0.100	174.129.27.168	TLSv1	Application Data, Application
155	97.805312	192.168.0.100	174.129.27.168	TLSv1	Application Data, Application
156	97.848793	174.129.27.168	192.168.0.100	TCP	https > bvcontrol [ACK] Seq=14
157	97.848865	192.168.0.100	174.129.27.168	TLSv1	Application Data, Application
158	97.848872	192.168.0.100	174.129.27.168	TLSv1	Application Data, Application
159	97.890781	174.129.27.168	192.168.0.100	TCP	https > bvcontrol [ACK] Seq=14
160	97.890856	192.168.0.100	174.129.27.168	TCP	[TCP segment of a reassembled
161	97.890864	192.168.0.100	174.129.27.168	TCP	[TCP segment of a reassembled
162	97.897797	174.129.27.168	192.168.0.100	TCP	https > bvcontrol [ACK] Seq=14
163	97.897850	192.168.0.100	174.129.27.168	TCP	[TCP segment of a reassembled

⊞ Frame 159 (54 bytes on wire, 54 bytes captured)

⊞ Ethernet II, Src: D-Link\_cf:ea:c7 (00:24:01:cf:ea:c7), Dst: HonHaiPr\_77:5d:a1 (00:25:56:77:5d:a1)

⊞ Internet Protocol, Src: 174.129.27.168 (174.129.27.168), Dst: 192.168.0.100 (192.168.0.100)

⊞ Transmission Control Protocol, Src Port: https (443), Dst Port: bvcontrol (1236), Seq: 1414, Ack: 4993, Len: 0

Hierarchical View

Frame (Bottom Layer)

Ethernet

IP

TCP (Top Layer)

- Note: The hierarchical display here is upside down compared to the Internet protocol stack that you have seen in the lectures.

# ANALYZING CAPTURED DATA

Microsoft (src net 192.168.1.6) [Wireshark 1.6.5 (SVN Rev 40429 from /trunk-1.6)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: http.request.version=="HTTP/1.1" Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Length	Info
686	12.794845	192.168.1.6	174.35.52.133	HTTP	462	GET /thumbnail/76d800a7jw1dpgsbd8ja6j.jpg HTTP/1.1
688	12.797284	192.168.1.6	174.35.52.133	HTTP	462	GET /thumbnail/69abd30bjw1dpgt36vcuvj.jpg HTTP/1.1
694	12.812058	192.168.1.6	174.35.52.142	HTTP	462	GET /thumbnail/7f1ef208jw1dpgu7nx3awj.jpg HTTP/1.1
729	13.007040	192.168.1.6	174.35.52.142	HTTP	462	GET /thumbnail/5f75ec4agw1dpgsmyhfinj.jpg HTTP/1.1
733	13.011754	192.168.1.6	174.35.52.142	HTTP	462	GET /thumbnail/93831636jw1dpg6jjkmzvj.jpg HTTP/1.1
734	13.012022	192.168.1.6	174.35.52.142	HTTP	462	GET /thumbnail/4711809ejw1dpgu2cd15rj.jpg HTTP/1.1
735	13.012321	192.168.1.6	174.35.52.142	HTTP	462	GET /thumbnail/7069fcb4jw1dpgrdksra2j.jpg HTTP/1.1
750	13.068017	192.168.1.6	174.35.52.142	HTTP	460	GET /middle/5f75ec4agw1dpgsmyhfinj.jpg HTTP/1.1

Frame 686: 462 bytes on wire (3696 bits), 462 bytes captured (3696 bits)

Ethernet II, Src: LiteonTe\_14:fc:f9 (68:a3:c4:14:fc:f9), Dst: Netgear\_2f:8b:49 (74:44:01:2f:8b:49)

Internet Protocol Version 4, Src: 192.168.1.6 (192.168.1.6), Dst: 174.35.52.133 (174.35.52.133)

Transmission Control Protocol, Src Port: 51529 (51529), Dst Port: http (80), Seq: 1, Ack: 1, Len: 408

Hypertext Transfer Protocol

- GET /thumbnail/76d800a7jw1dpgsbd8ja6j.jpg HTTP/1.1\r\n
- Host: ww1.sinaimg.cn\r\n
- Connection: keep-alive\r\n
- User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.7 (KHTML, like Gecko) Chrome/16.0.912.77 Safari/535.7\r\n
- Accept: \*/\*\r\n
- Referer: http://www.weibo.com/u/1740944337?wvr=3.6&lf=reg\r\n
- Accept-Encoding: gzip, deflate, sdch\r\n
- Accept-Language: en-US,en;q=0.8\r\n
- Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.3\r\n
- \r\n
- [Full request URI: <http://ww1.sinaimg.cn/thumbnail/76d800a7jw1dpgsbd8ja6j.jpg>]

- HTTP header

# SO MANY STRANGE PACKETS!

- Wireshark captures everything that is sent/recived on the chosen interface. You need to filter what you want.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	fe80::881b:db02:330	fe80::a86e:4798:aae	ICMPV6	86	Neighbor Solicitation for f
2	0.000130	fe80::a86e:4798:aae	fe80::881b:db02:330	ICMPV6	86	Neighbor Advertisement fe80
3	0.156783	fe80::a86e:4798:aae	fe80::881b:db02:330	ICMPV6	86	Neighbor Solicitation for f
4	0.158681	fe80::881b:db02:330	fe80::a86e:4798:aae	ICMPV6	86	Neighbor Advertisement fe80
5	0.630269	fe80::a86e:4798:aae	ff02::c	SSDP	208	M-SEARCH * HTTP/1.1
6	3.218276	192.168.0.101	192.168.0.255	NBNS	92	Name query NB BRW0CEEE6BC03
7	3.630462	fe80::a86e:4798:aae	ff02::c	SSDP	208	M-SEARCH * HTTP/1.1
8	3.968189	192.168.0.101	192.168.0.255	NBNS	92	Name query NB BRW0CEEE6BC03
9	4.718230	192.168.0.101	192.168.0.255	NBNS	92	Name query NB BRW0CEEE6BC03
10	5.469572	192.168.0.101	192.168.0.255	NBNS	92	Name query NB BRW0CEEE6BC03
11	6.219317	192.168.0.101	192.168.0.255	NBNS	92	Name query NB BRW0CEEE6BC03
12	6.969368	192.168.0.101	192.168.0.255	NBNS	92	Name query NB BRW0CEEE6BC03
13	7.631027	fe80::a86e:4798:aae	ff02::c	SSDP	208	M-SEARCH * HTTP/1.1
14	10.631837	fe80::a86e:4798:aae	ff02::c	SSDP	208	M-SEARCH * HTTP/1.1

Control Messages

NetBIOS Packets

Discovery Packets

# WIRESHARK FILTERS

- **Two types of filters:**
  - Capture Filters
  - Display Filters
- Wireshark contains a powerful **capture** filter engine that helps **remove unwanted packets** from a packet trace and only retrieves the packets of our interest.
- **Display** filters let you compare the fields within a protocol against a specific value, compare fields against fields, and check the existence of specified fields or protocols

# CAPTURE OPTIONS

The screenshot shows the Wireshark Network Analyzer interface with the 'Capture Interfaces' dialog box open. The dialog box contains a table of available network interfaces for capture. The 'Options' button for the second Microsoft interface is highlighted with a red box.

Description	IP	Packets	Packets/s	Stop
Intel(R) 82566MM Gigabit Network Connection	fe80::405e:b889:63a1:4389	0	0	Start Options Details
Microsoft	fe80::a86e:4798:aae2:9b3	430	1	Start <b>Options</b> Details
Microsoft	fe80::2d40:1deb:ff11:2c72	0	0	Start Options Details

Buttons: Help, Close

Bottom status bar: Ready to load or capture | No Packets | Profile: Default

# EXAMPLE OF A CAPTURE FILTER

**Capture**

Interface: Local Microsoft: \Device\NPF\_{C372DBF0-E317-4323-96CD-9A93BB8} IP address: fe80::b8d8:a9c3:ac04:ce48, 192.168.0.100

Link-layer header type: Ethernet Wireless Settings Remote Settings

Capture packets in promiscuous mode

Capture packets in pcap-ng format (experimental)

Limit each packet to 1 megabyte(s) Buffer size: 1 megabyte(s)

**Capture Filter:** (host 192.168.0.100) || (host 174.36.30.66)

**Capture File(s)**

File: 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)

Automatic scrolling in live capture

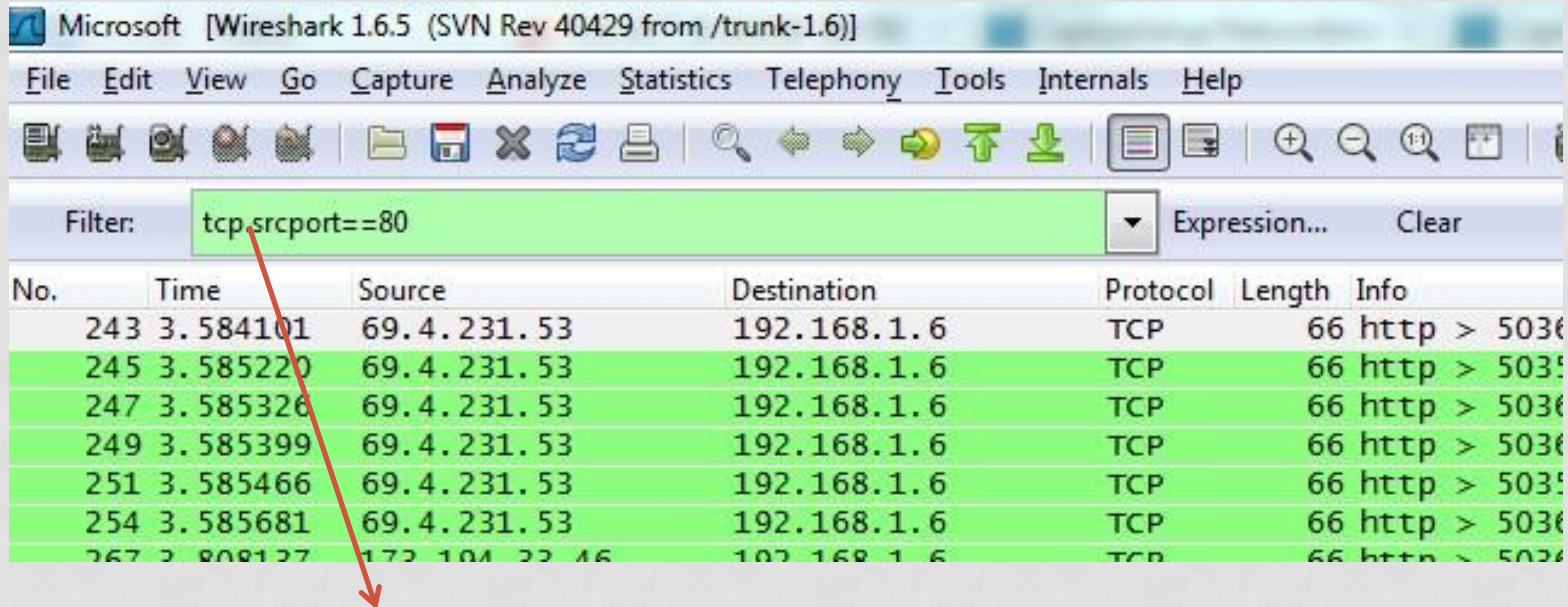
Hide capture info dialog

**Name Resolution**

Enable MAC name resolution

Enter a capture filter to reduce the amount of packets to be captured. See "Capture Filters" in the online help for further information how to use it.

# EXAMPLE OF A DISPLAY FILTER



Microsoft [Wireshark 1.6.5 (SVN Rev 40429 from /trunk-1.6)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: `tcp.srcport==80` Expression... Clear

No.	Time	Source	Destination	Protocol	Length	Info
243	3.584101	69.4.231.53	192.168.1.6	TCP	66	http > 5036
245	3.585220	69.4.231.53	192.168.1.6	TCP	66	http > 5035
247	3.585326	69.4.231.53	192.168.1.6	TCP	66	http > 5036
249	3.585399	69.4.231.53	192.168.1.6	TCP	66	http > 5036
251	3.585466	69.4.231.53	192.168.1.6	TCP	66	http > 5035
254	3.585681	69.4.231.53	192.168.1.6	TCP	66	http > 5036
267	3.808127	172.16.17.16	192.168.1.6	TCP	66	http > 5036

- Display filter separates the packets to be displayed (In this case, only packets with source port 80 are displayed)



# WIRESHARK FILTERS

- **Comparison operators**
- Fields can also be compared against values. The comparison operators can be expressed either through English-like abbreviations or through C-like symbols:
  - eq, == Equal
  - ne, != Not Equal
  - gt, > Greater Than
  - lt, < Less Than
  - ge, >= Greater than or Equal to
  - le, <= Less than or Equal to

# WIRESHARK FILTERS

- **Logical Expressions**

Tests can be combined using logical expressions. These too are expressible in C-like syntax or with English-like abbreviations:

and, && Logical AND

or, | | Logical OR

not, ! Logical NOT

- Some Valid Display Filters
- `tcp.port == 80 and ip.src == 192.168.2.1`
- `http and frame[100-199] contains "wireshark"`

# WIRESHARK FILTERS

- **The Slice Operator**

- You can take a slice of a field if the field is a text string or a byte array. For example, you can filter the HTTP header fields . Here the header “location” indicates the REDIRECTION happens.

`http.location[0:12]== "http://pages"`

- Another example is:

`http.content_type[0:4] == "text"`

# CAPTURE FILTERS

Syntax	Protocol	Direction	Host(s)	Logical Op.	Other Express.
Example	tcp	dst	136.159.5.20	and	host 136.159.5.6

- **Protocol:**
  - Values: ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp and udp.
  - If no protocol is specified, all the protocols are used.
- **Direction:**
  - Values: src, dst, src and dst, src or dst
  - If no source or destination is specified, the "src or dst" keywords are applied.
  - For example, "host 136.159.5.20" is equivalent to "src or dst host 136.159.5.20".

# CAPTURE FILTERS

- **Host(s):**

- Values: net, port, host, portrange.
- If no host(s) is specified, the "host" keyword is used.
- For example, "src 136.159.5.20" is equivalent to "src host 136.159.5.20".

- **Logical Operations:**

- Values: not, and, or.
- Negation ("not") has highest precedence. Alternation ("or") and concatenation ("and") have equal precedence and associate left to right.
- For example,  
"not tcp port 3128 and tcp port 80" is equivalent to "(not tcp port 3128) and tcp port 80".

# CAPTURE FILTERS (EXAMPLES)

- **tcp port 80**

Displays packets with tcp protocol on port 80.

- **ip src host 136.159.5.20**

Displays packets with source IP address equals to 136.159.5.20.

- **host 136.159.5.1**

Displays packets with source or destination IP address equals to 136.159.5.1.

- **src portrange 2000-2500**

Displays packets with source UDP or TCP ports in the 2000-2500 range.

# CAPTURE FILTERS(EXAMPLES)

- **src host 136.159.5.20 and not dst host 136.159.5.1**

Displays packets with source IP address equals to 136.159.5.20 and in the same time not with the destination IP address 136.159.5.1.

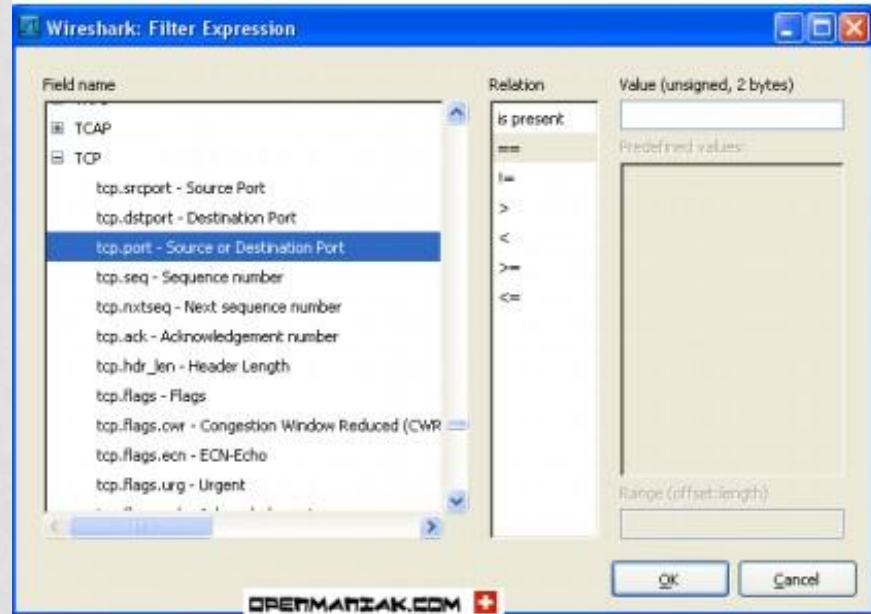
- **(src host 136.159.5.1 or src host 136.159.5.3) and tcp dst portrange 200-10000 and dst host 136.159.5.2**

Displays packets with source IP address 136.159.5.1 or source address 136.159.5.3, the result is then concatenated with packets having destination TCP portrange from 200 to 10000 and destination IP address 136.159.5.2.

# DISPLAY FILTERS

Syntax	Protocol	.	String 1	.	String 2	Comparison operators	Value	Logical Op.	Other Expr.
Example	http	.	request	.	method	==	get	or	tcp.port == 80

- String1, String2 (Optional settings): Sub protocol categories inside the protocol. To find them, look for a protocol and then click on the "+" character.





# DISPLAY FILTERS(EXAMPLES)

- **ip.addr == 136.159.5.20**

Displays the packets with source or destination IP address equals to 136.159.5.20 .

- **http.request.version=="HTTP/1.1"**

Display http Version

- **tcp.dstport == 25**

- **tcp.flags**

Display packets having a TCP flags

- **tcp.flags.syn == 0x02**

Display packets with a TCP SYN flag.