WIRESHARK

1)How to Download Wireshark:

Downloading and installing Wireshark is easy. Step one is to check the official https://www.wireshark.org/download.html for the operating system you need. The basic version of Wireshark is free.

- Wireshark for windows:

Wireshark comes in two flavors for Windows, 32 bit and 64 bit. Pick the correct version for your OS. The current release is 3.4.1 as of this writing. The installation is simple and shouldn't cause any issues.

- Wireshark for Mac:

  Wireshark is available on Mac as a Homebrew install.To install Homebrew, you need to run this command at your Terminal prompt: `/usr/bin/ruby -e "$(curl-fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"`.Once you have the Homebrew system in place, you can access several open-source projects for your Mac. To install Wireshark run this command from the Terminal

  ```
  brew install wireshark
  ```

  Homebrew will download and install Wireshark and any dependencies so it will run correctly.

- Wireshark for Linux:

  Installing Wireshark on Linux can be a little different depending on the Linux distribution. If you aren't running one of the following distros, please double-check the commands.

From a terminal prompt, run these commands:

1. `sudo apt-get install wireshark`
2. `sudo dpkg-reconfigure wireshark-common`
3. `sudo adduser $USER wireshark`

   Those commands download the package, update the package, and add user privileges to run Wireshark.

Red Hat Fedora:
From a terminal prompt, run these commands:

1. `sudo dnf install wireshark-qt`
2. `sudo usermod -a -G wireshark username`

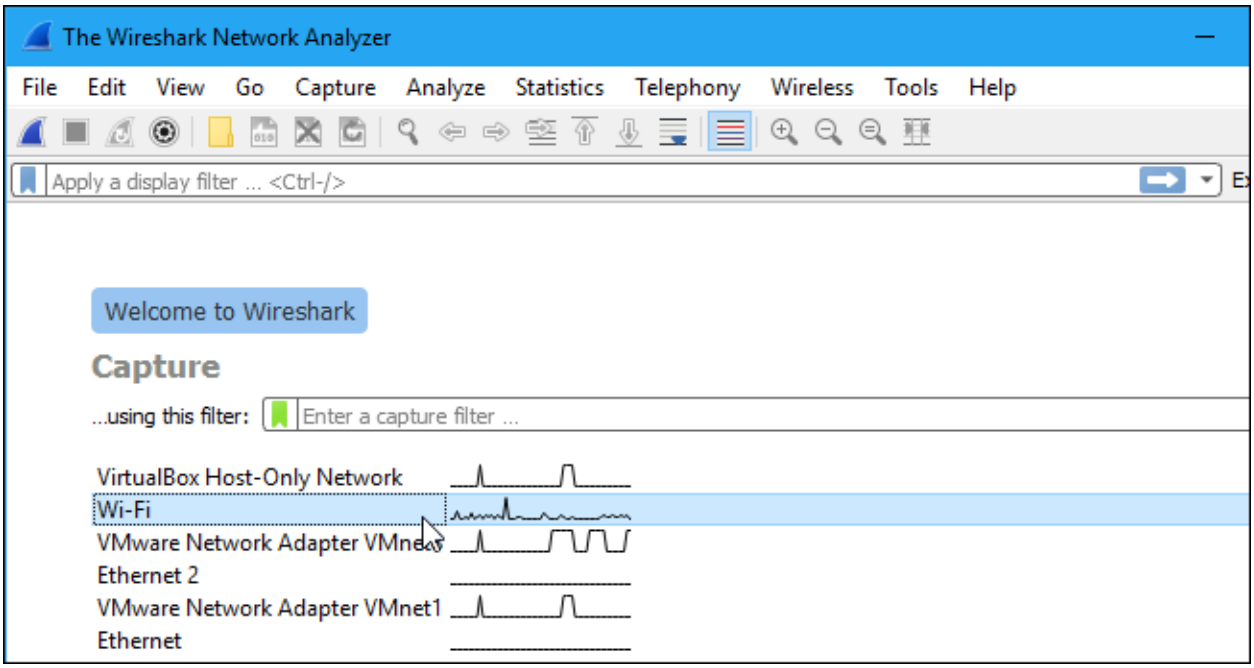
The first command installs the GUI and CLI version of Wireshark, and the second adds permissions to use Wireshark.

Kali Linux:

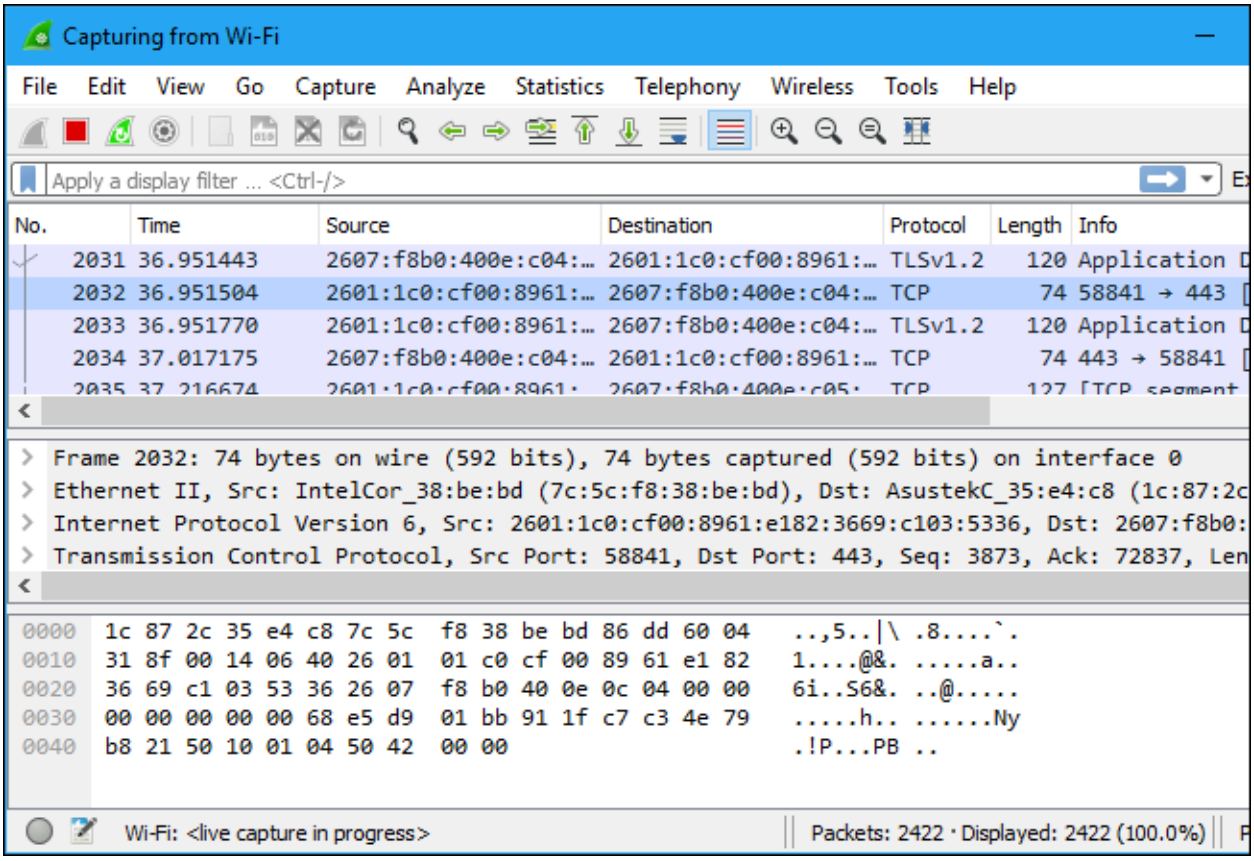
Wireshark is probably already installed! It's part of the basic package. Check your menu to verify. It's under the menu option "Sniffing & Spoofing."

1)Capturing Packets:

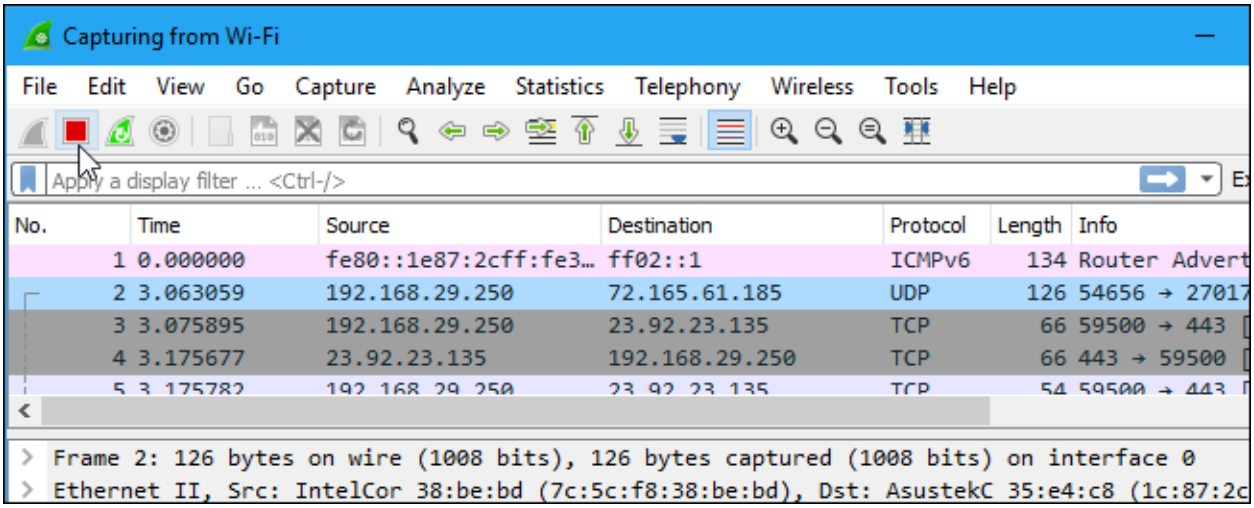
After downloading and installing Wireshark, you can launch it and double-click the name of a network interface under Capture to start capturing packets on that interface. For example, if you want to capture traffic on your wireless network, click your wireless interface.



As soon as you click the interface's name, you'll see the packets start to appear in real time. Wireshark captures each packet sent to or from your system.



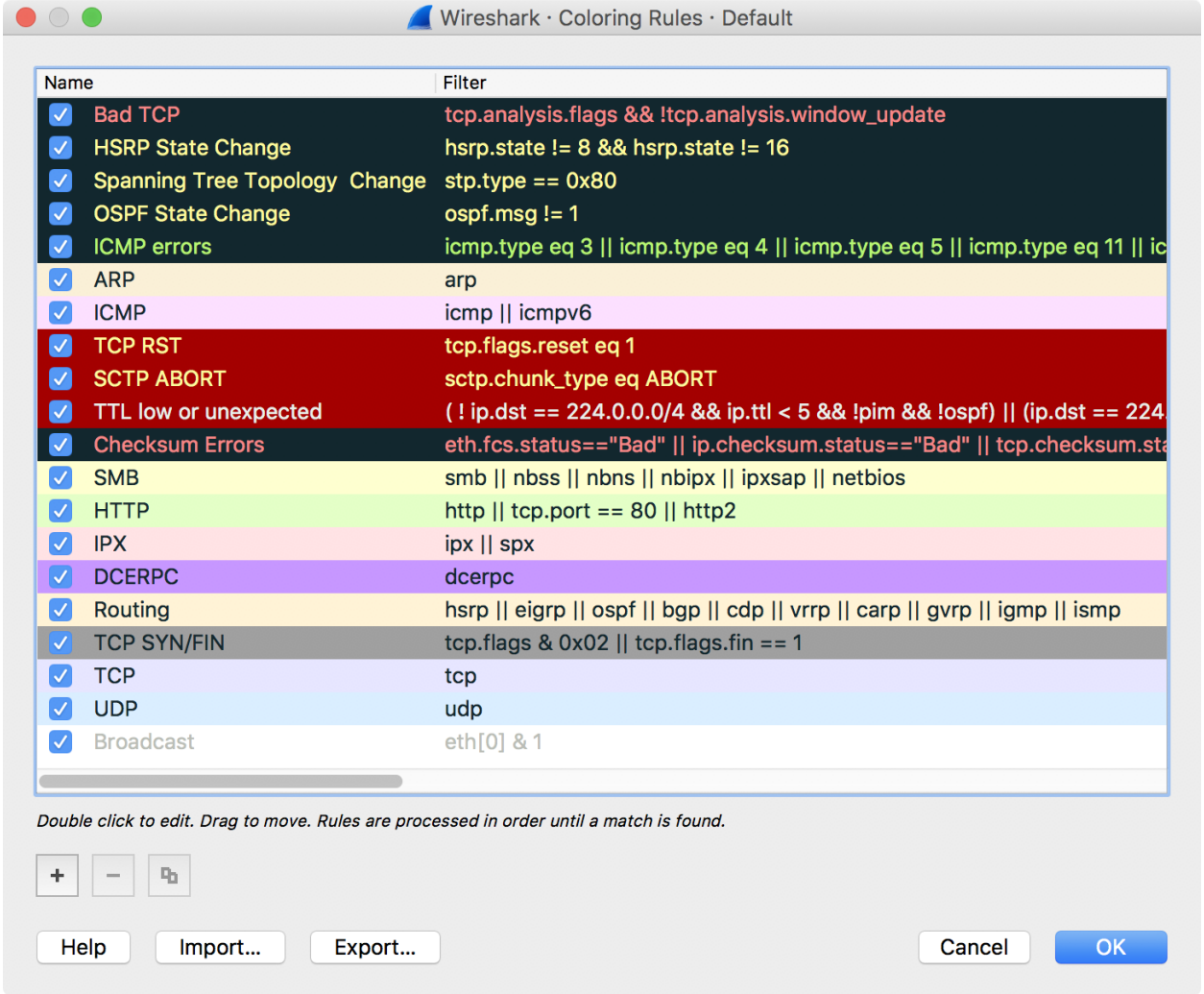
Click the red “Stop” button near the top left corner of the window when you want to stop capturing traffic.



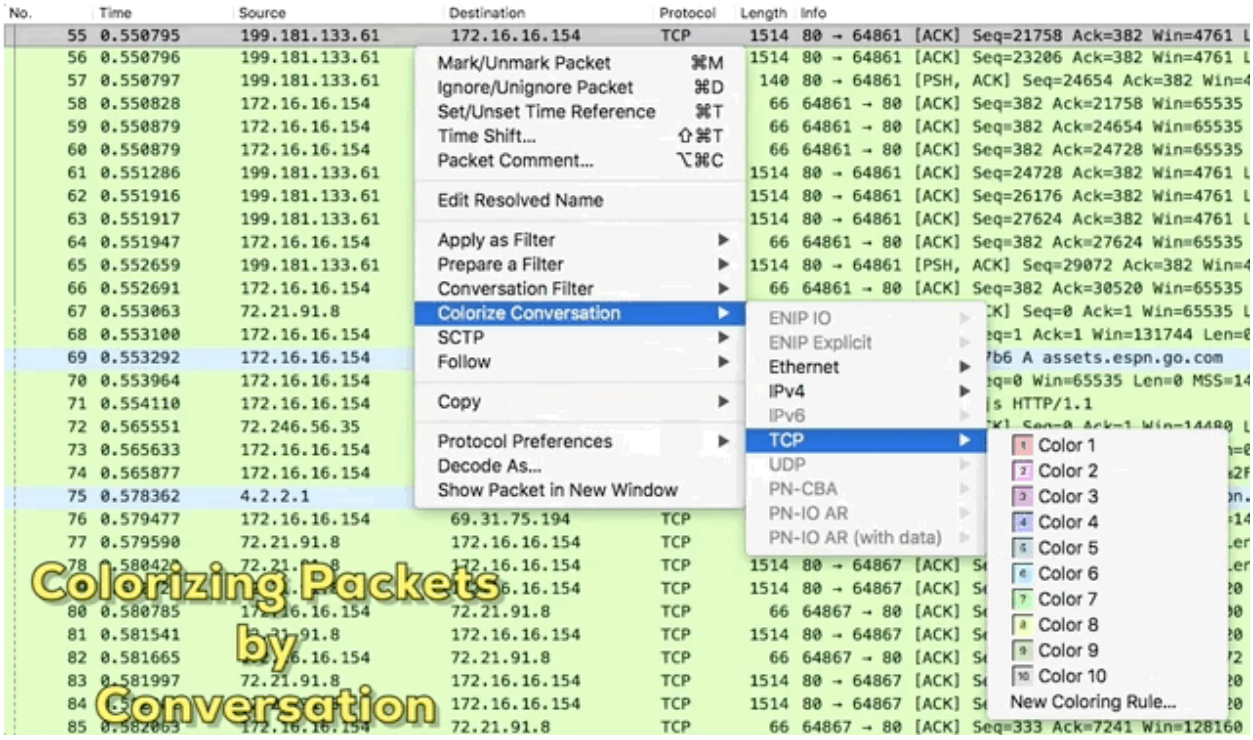
2)Color Coding:

You’ll probably see packets highlighted in a variety of different colors. Wireshark uses colors to help you identify the types of traffic at a glance. By default, light purple is TCP traffic, light blue is UDP traffic, and black identifies packets with errors—for example, they could have been delivered out of order.

To view exactly what the color codes mean, click View > Coloring Rules. You can also customize and modify the coloring rules from here, if you like.



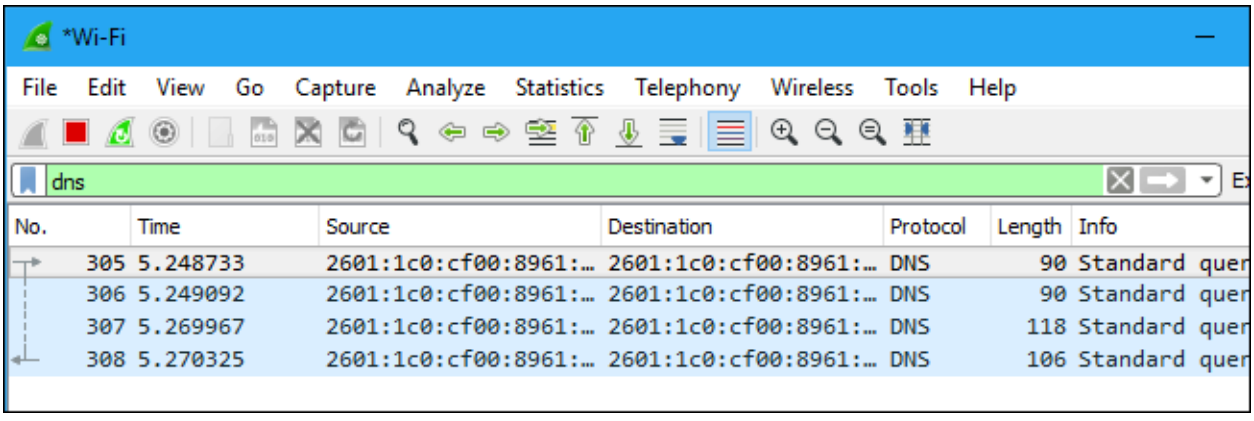
In most cases, I apply coloring rules to individual conversations. **That's even easier. Pick a packet in a capture file, right-click it, and hover over Colorize Conversation. The slide-out menu will reveal options that allow you to define the type of conversation (IPv4, IPv6, TCP, etc).**



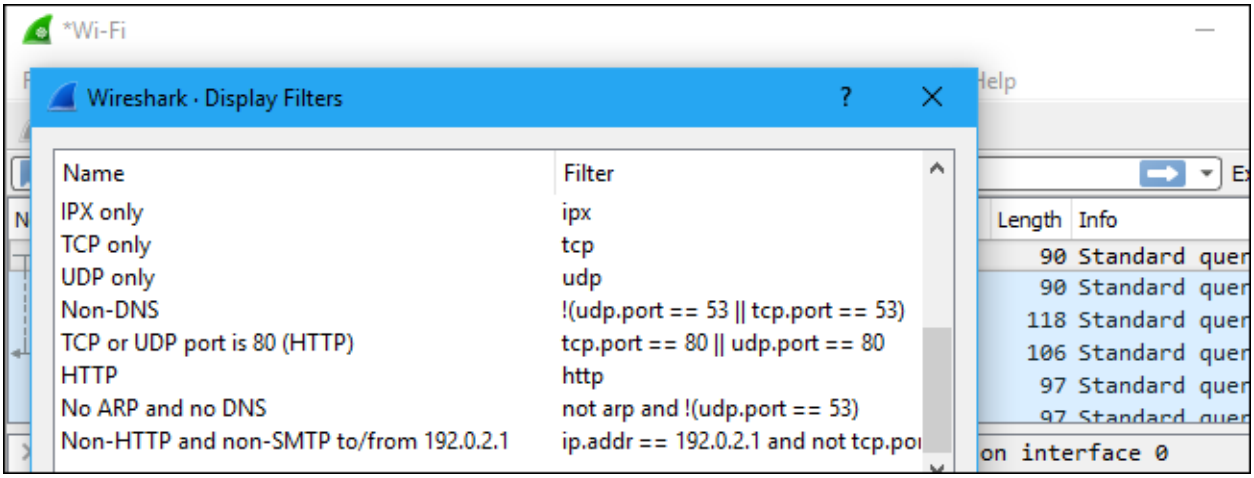
3)Filtering Packets:

If you're trying to inspect something specific, such as the traffic a program sends when phoning home, it helps to close down all other applications using the network so you can narrow down the traffic. Still, you'll likely have a large amount of packets to sift through. That's where Wireshark's filters come in.

The most basic way to apply a filter is by typing it into the filter box at the top of the window and clicking Apply (or pressing Enter). For example, type "dns" and you'll see only DNS packets. When you start typing, Wireshark will help you autocomplete your filter.



You can also click Analyze > Display Filters to choose a filter from among the default filters included in Wireshark. From here, you can add your own custom filters and save them to easily access them in the future.



Display Filter:

- Show only SMTP (port 25) and ICMP traffic:
  `tcp.port eq 25 or icmp`

- Show only traffic in the LAN (192.168.x.x), between workstations and servers -- no Internet:
  `ip.src==192.168.0.0/16 and ip.dst==192.168.0.0/16`

- TCP buffer full -- *Source is instructing Destination to stop sending data*
  `tcp.window_size == 0 && tcp.flags.reset != 1`

- Match HTTP requests where the last characters in the uri are the characters "gl=se":
  `http.request.uri matches "gl=se$"`

  Note: The $ character is a PCRE punctuation character that matches the end of a string, in this case the end of http.request.uri field.

- Filter by a protocol (e.g. SIP) and filter out unwanted IPs:
  `ip.src != xxx.xxx.xxx.xxx && ip.dst != xxx.xxx.xxx.xxx && sip`

- Capture only traffic to or from IP address 172.18.5.4:
  `host 172.18.5.4`

- Capture traffic from a range of IP addresses:

  `Src net 192.168.0.0/24 (OR)  src net 192.168.0.0 mask 255.255.255.0`

  `(OR) dst net 192.168.0.0/24` `(OR)` **`dst net 192.168.0.0 mask 255.255.255.0`**

- Capture only DNS (port 53) traffic:

  `port 53`

- Capture non-HTTP and non-SMTP traffic on your server (both are equivalent):

  `host www.example.com and not (port 80 or port 25)  (OR)`

  `host www.example.com and not port 80 and not port 25`

- Capture traffic within a range of ports
  `tcp portrange 1501-1549`

Display Filter comparison operators:

English	C-like	Description	Example
eq	==	Equal	`ip.src==10.0.0.5`
ne	!=	Not equal	`ip.src!=10.0.0.5`
gt	>	Greater than	`frame.len > 10`
lt	<	Less than	`frame.len < 128`
ge	>=	Greater than or equal to	`frame.len ge 0x100`
le	<=	Less than or equal to	`frame.len <= 0x20`
contains		Protocol, field or slice contains a value	`sip.To contains "a1762"`
matches	~	Protocol or text field matches a Perl-compatible regular expression	`http.host matches "acme\.(org|com|net)"`
bitwise_and	&	Bitwise AND is non-zero	`tcp.flags & 0x02`

Display Filter Logical Operations:

English	C-like	Description	Example
and	&&	Logical AND	`ip.src==10.0.0.5 and tcp.flags.fin`
or	||	Logical OR	`ip.scr==10.0.0.5 or ip.src==192.1.1.1`
xor	^^	Logical XOR	`tr.dst[0:3] == 0.6.29 xor tr.src[0:3] == 0.6.29`
not	!	Logical NOT	`not llc`
[...]		Subsequence	See “Slice Operator” below.
in		Set Membership	http.request.method in {"HEAD" "GET"}. See “Membership Operator” below.

NOTE: WIRESHARK CHEAT SHEET PDF FILE

4)Inspecting Packets: