



# Wireshark

**Jin Seek Choi**

[jinseek@hanyang.ac.kr](mailto:jinseek@hanyang.ac.kr)

**Ref: Hiral Chhaya, Anvita Priyam**

**By Fei Xu**

# About Wireshark

- It is a packet sniffer Computer application
- Functionality is very similar to tcpdump
- Has a GUI front-end and many more information sorting and filtering options

# About Wireshark

- Wireshark is a network **packet/protocol analyzer**.
  - A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible.
- Wireshark is perhaps **one of the best open source packet analyzers** available today for UNIX and Windows.
- It has two libraries:
  - WinPcap for Windows OS
  - LibPcap for Linux OS

# Some intended purposes

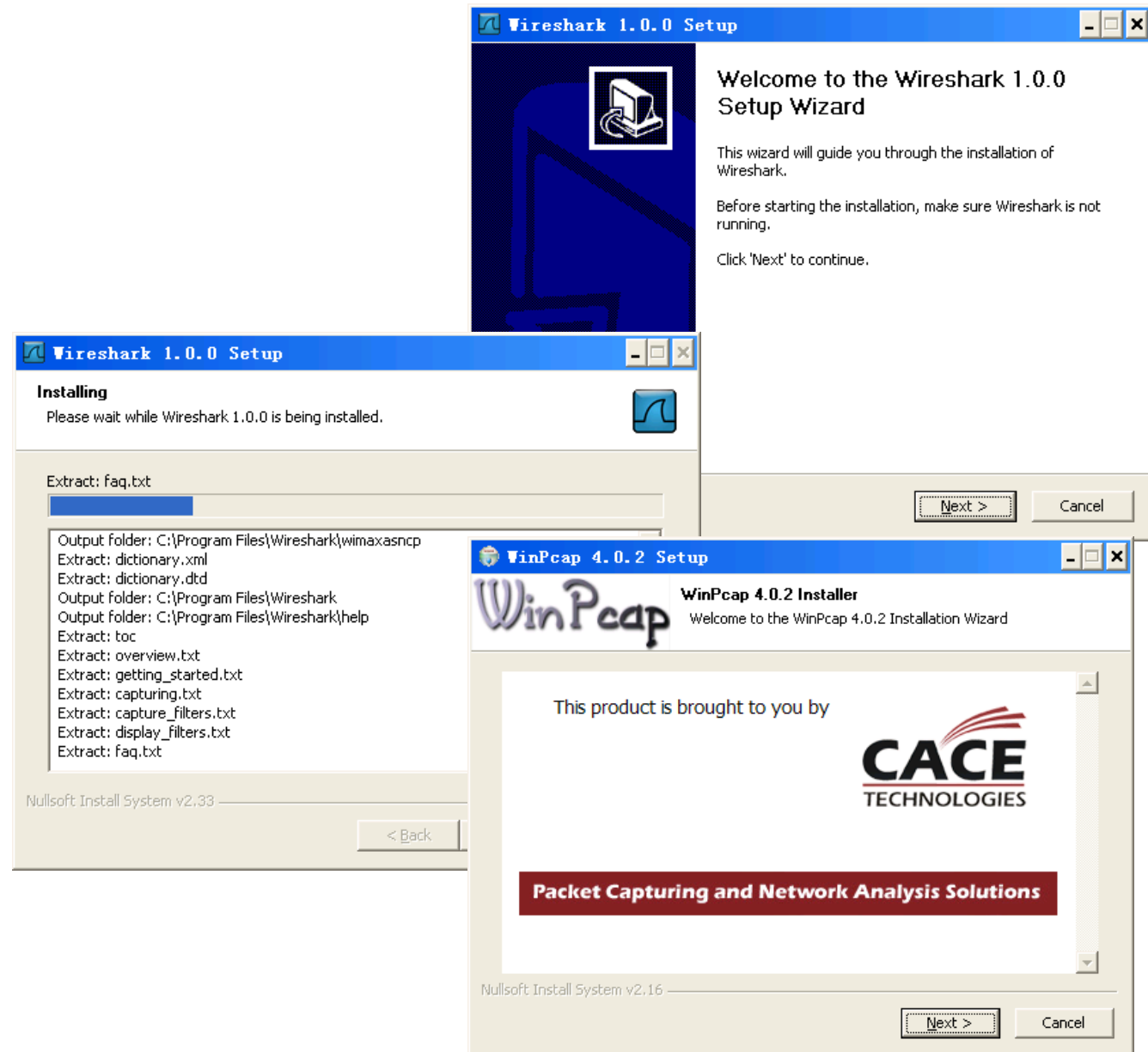
- Network administrators use it to **troubleshoot network problems**
- Network security engineers use it to examine security problems
- Developers use it to debug protocol implementations
- People use it to **learn network protocol internals**
- Wireshark isn't an intrusion detection system
- Wireshark will not manipulate things on the network, it will only "measure" things from it

# Install Wireshark

- Visit this site
  - <http://www.wireshark.org>
  - <http://wiki.wireshark.org>
- Windows
  - Download
    - wireshark-win32(or win64)-(version).exe
- Debian/Ubuntu Linux
  - `sudo apt-get install wireshark`
  - `sudo wireshark`

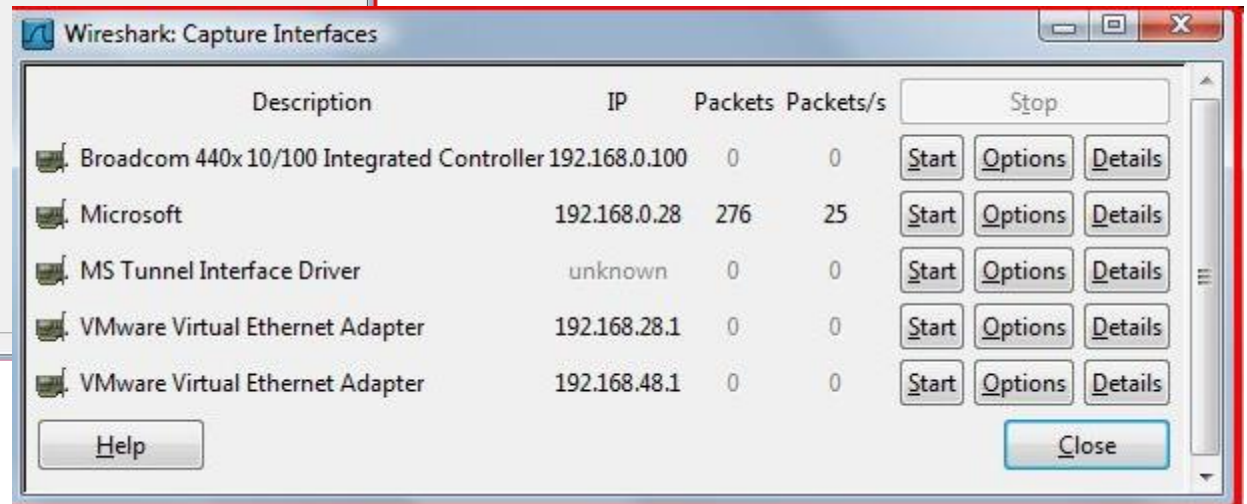
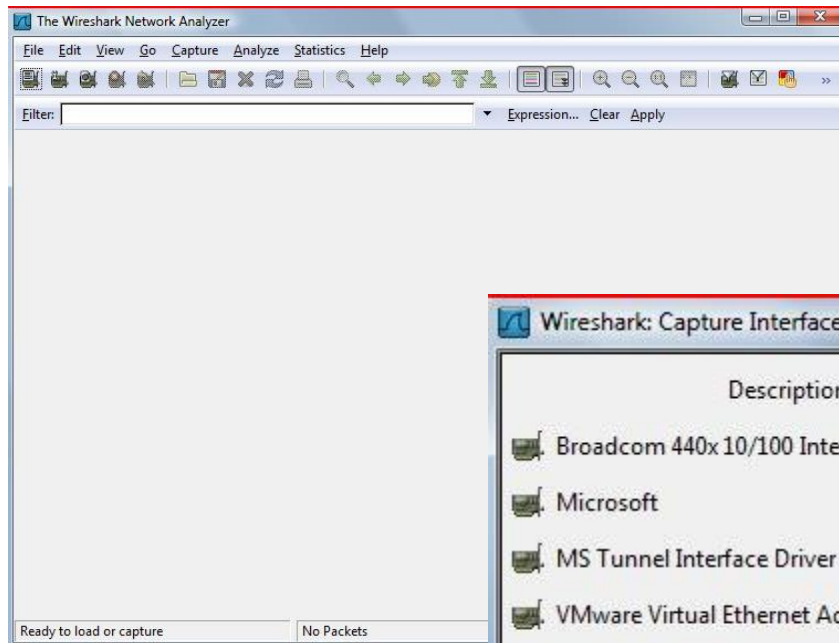
# Install under Windows

- Download
- Install



# Install under Debian/Ubuntu

- *# apt-get install wireshark*



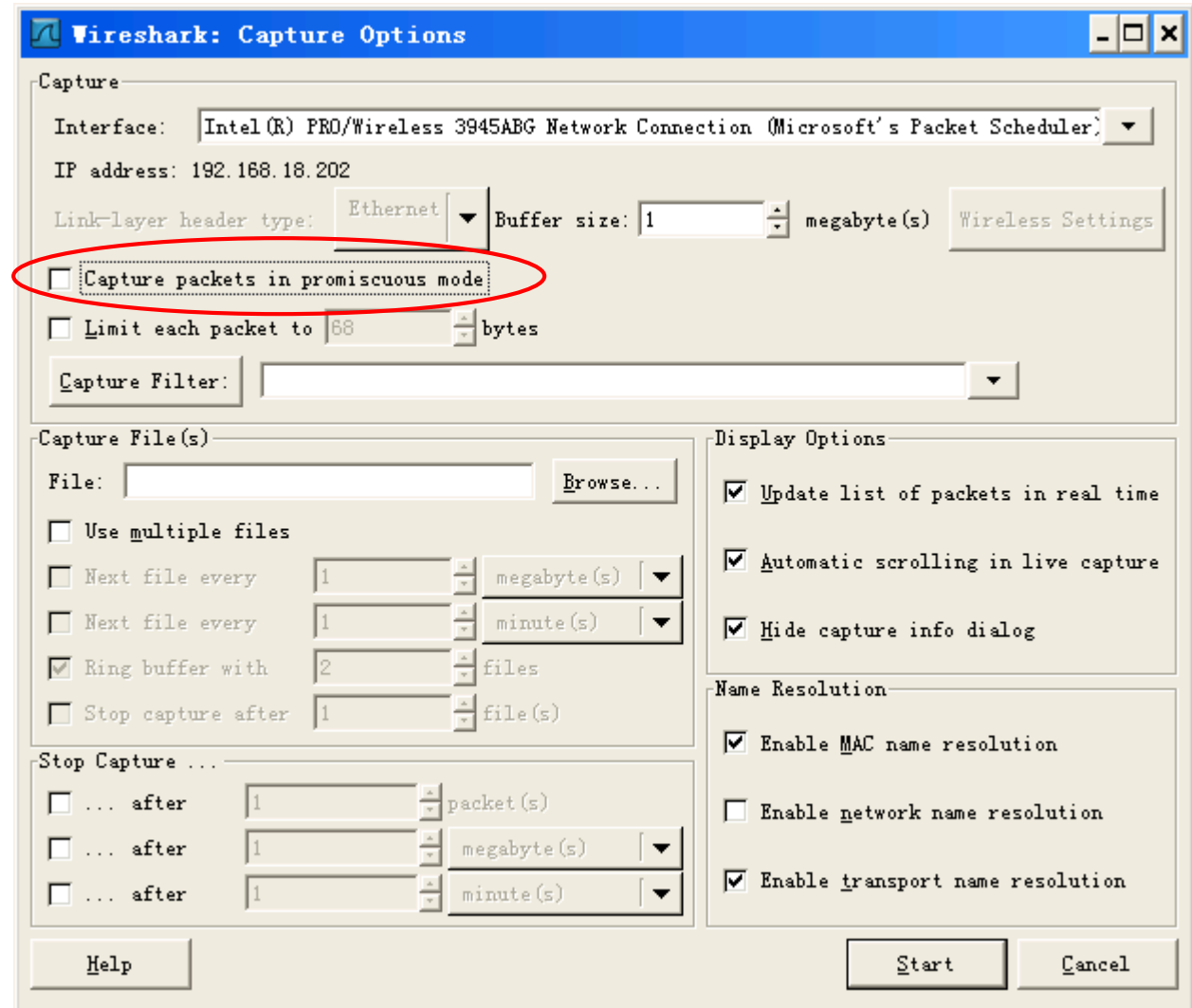
# Configuration

- Turn the **PROMISCUOUS MODE** off!
- If you are at work, your Network Administrator may see you running in **PROMISCUOUS MODE**: a feature normally used for packet sniffing
- You maybe considered as a sniffer and somebody may decide to fire you for that



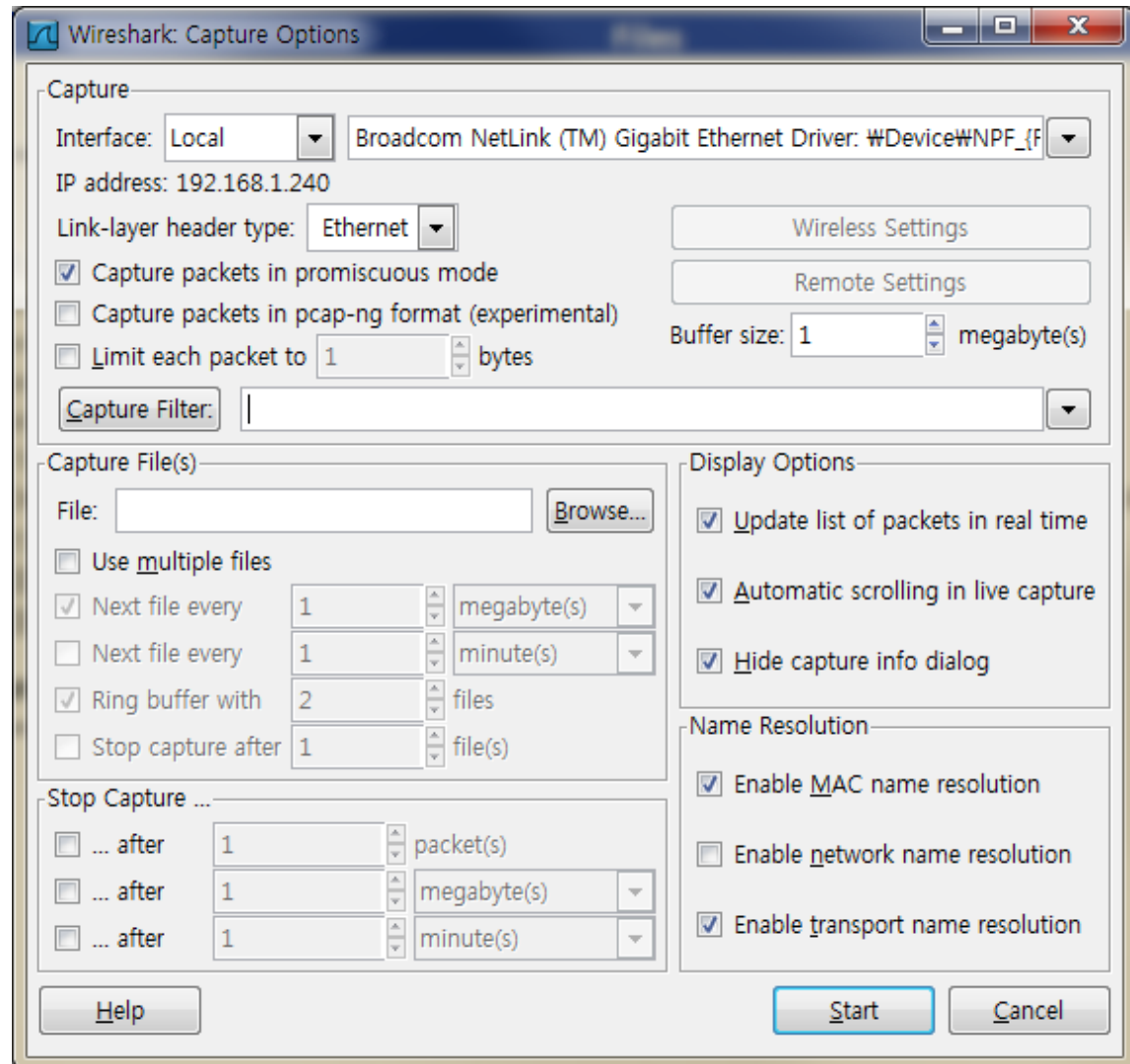
# Configuration

This checkbox allows you to specify that Wireshark should put the interface in **promiscuous** mode when capturing. If you do not specify this, Wireshark will only capture the packets going to or from your computer (not all packets on your LAN segment)



# Configuration

- **Promiscuous mode**
  - promiscuous mode is a configuration of a network card that makes the card pass all traffic it receives to the central processing unit rather than just frames addressed to it



# Configuration: Capture Options

**Ethereal: Capture Options**

**Capture**

Interface: Realtek RTL8139/810x Family Fast Ethernet NIC (Microsoft: ▼)

IP address: 10.3.100.231

Link-layer header type: Ethernet ▼ Buffer size: 1 megabyte(s)

☒ Capture packets in promiscuous mode

☐ Limit each packet to 68 bytes

Capture Filter: ▼

**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) ▼

Help Start Cancel

**Display Options**

☒ Update list of packets in real time

☒ Automatic scrolling in live capture

☐ Hide capture info dialog

**Name Resolution**

☒ Enable MAC name resolution

☒ Enable network name resolution

☒ Enable transport name resolution

**Ethereal: Capture from Realtek RT...**

**Captured Packets**

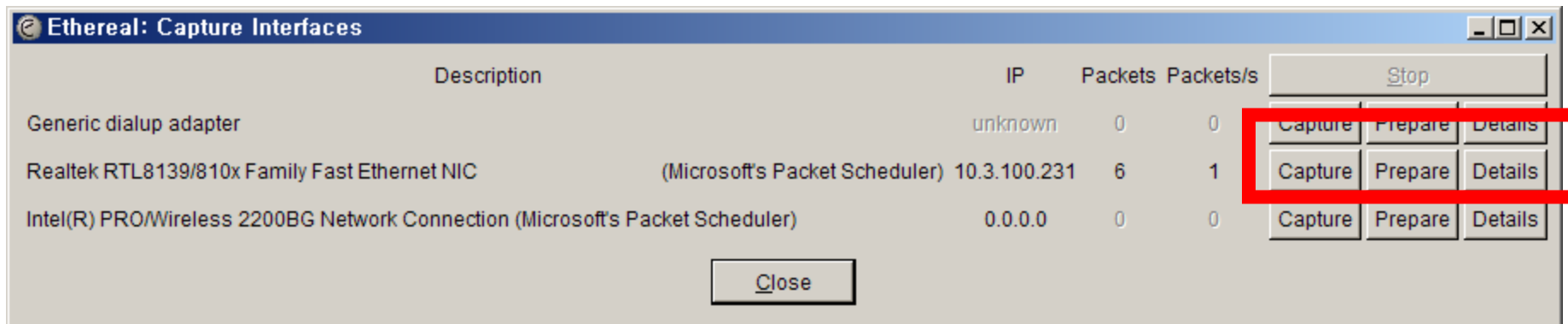
	Total	% of total
SCTP	0	0.0%
TCP	0	0.0%
UDP	4	19.0%
ICMP	0	0.0%
ARP	15	71.4%
OSPF	0	0.0%
GRE	0	0.0%
NetBIOS	1	4.8%
IPX	0	0.0%
VINES	0	0.0%
Other	1	4.8%

Running 00:00:03

Stop

# Capturing Live Network Data

- Start Capturing
- "Capture Interfaces" dialog box
- "Capture Options" dialog box



# Demo

✓ HTTP

• TCP

• DNS

• ARP



Photo credit: Jeff Kubina

# Demo: HTTP Packet Capture

- When you browse Internet, you can capture HTTP (TCP) packets

Microsoft - Wireshark

File Edit View Go Capture Analyze Statistics Telephony Tools Help

Filter: Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
35	0.008972	192.168.1.195	175.158.11.52	TCP	53566 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
36	0.009213	192.168.1.195	175.158.11.52	TCP	53568 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
37	0.009571	192.168.1.195	175.158.11.52	TCP	53565 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
38	0.009823	192.168.1.195	175.158.11.52	TCP	53567 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
39	0.010454	192.168.1.195	1.226.51.70	TCP	53577 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
40	0.010697	192.168.1.195	1.226.51.70	TCP	53575 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
41	0.010977	192.168.1.195	1.226.51.70	TCP	53578 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
42	0.011211	192.168.1.195	1.226.51.70	TCP	53576 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
43	0.011444	192.168.1.195	1.226.51.70	TCP	53579 > http [FIN, ACK] Seq=1 Ack=1 win=4380 Len=0
44	0.018145	119.205.216.217	192.168.1.195	TCP	http > 53539 [FIN, ACK] Seq=1 Ack=2 win=5840 Len=0
45	0.018230	192.168.1.195	119.205.216.217	TCP	53539 > http [ACK] Seq=2 Ack=2 win=4380 Len=0
46	0.018434	119.205.216.217	192.168.1.195	TCP	http > 53541 [FIN, ACK] Seq=1 Ack=2 win=5840 Len=0
47	0.018495	192.168.1.195	119.205.216.217	TCP	53541 > http [ACK] Seq=2 Ack=2 win=4380 Len=0
48	0.021689	202.131.25.79	192.168.1.195	TCP	http > 53544 [RST] Seq=1 win=0 Len=0
49	0.021877	202.131.25.79	192.168.1.195	TCP	http > 53542 [RST] Seq=1 win=0 Len=0
50	0.030471	202.131.25.79	192.168.1.195	TCP	http > 53545 [RST] Seq=1 win=0 Len=0
51	0.030560	1.226.51.181	192.168.1.195	TCP	http > 53580 [FIN, ACK] Seq=1 Ack=2 win=46 Len=0
52	0.030616	192.168.1.195	1.226.51.181	TCP	53580 > http [ACK] Seq=2 Ack=2 win=4380 Len=0
53	0.034939	202.131.25.79	192.168.1.195	TCP	http > 53546 [RST] Seq=1 win=0 Len=0
54	0.035065	1.226.51.181	192.168.1.195	TCP	http > 53583 [FIN, ACK] Seq=1 Ack=2 win=46 Len=0
55	0.035142	192.168.1.195	1.226.51.181	TCP	53583 > http [ACK] Seq=2 Ack=2 win=4380 Len=0
56	0.038849	1.226.51.181	192.168.1.195	TCP	http > 53582 [FIN, ACK] Seq=1 Ack=2 win=46 Len=0
57	0.038937	192.168.1.195	1.226.51.181	TCP	53582 > http [ACK] Seq=2 Ack=2 win=4380 Len=0

Frame 66: 54 bytes on wire (432 bits), 54 bytes captured (432 bits)

Ethernet II, Src: Buffalo\_b1:fd:50 (00:1d:73:b1:fd:50), Dst: IntelCor\_29:5f:9c (8c:a9:82:29:5f:9c)

Internet Protocol, Src: 202.131.28.38 (202.131.28.38), Dst: 192.168.1.195 (192.168.1.195)

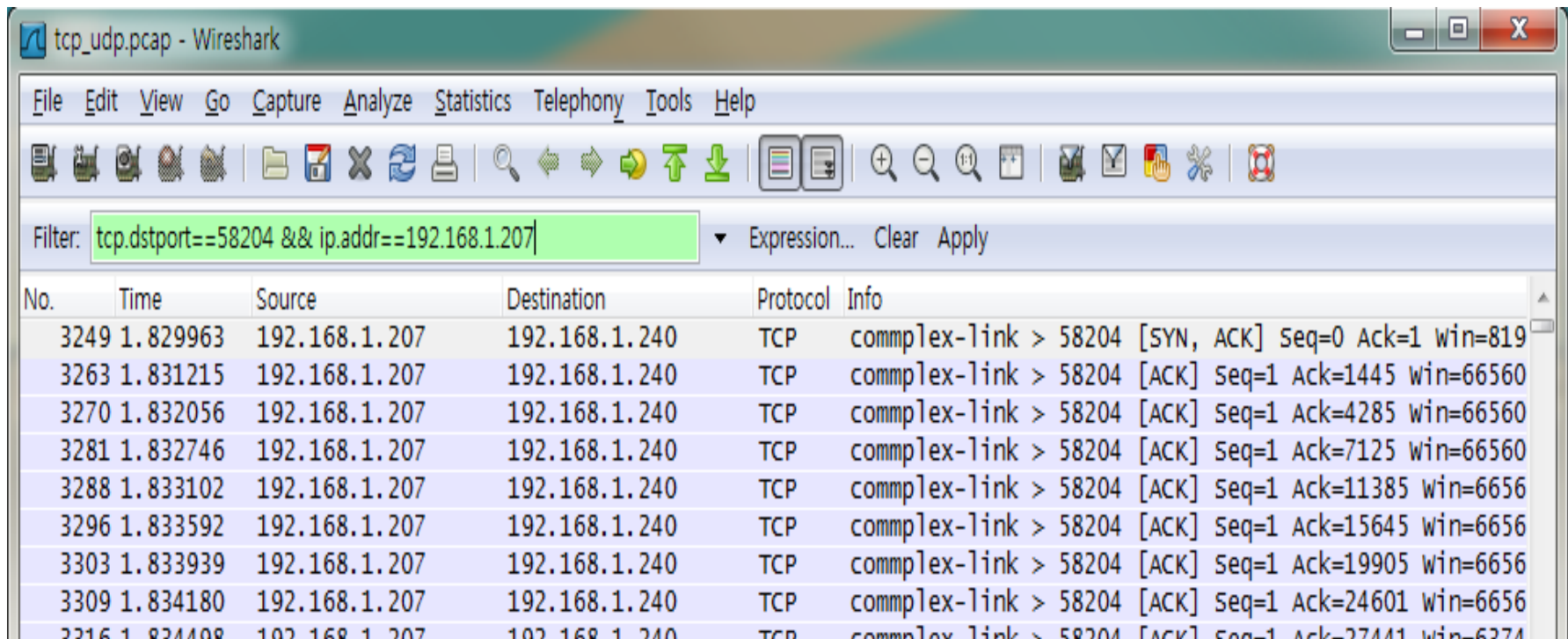
Version: 4  
Header length: 20 bytes  
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)  
Total Length: 40  
Identification: 0xe1e4 (57828)  
Flags: 0x02 (Don't Fragment)  
Fragment offset: 0  
Time to live: 47  
Protocol: TCP (6)  
Header checksum: 0xc0d6 [correct]  
Source: 202.131.28.38 (202.131.28.38)  
Destination: 192.168.1.195 (192.168.1.195)

Transmission Control Protocol, Src Port: http (80), Dst Port: 53556 (53556), Seq: 1, Ack: 2, Len: 0

Source port: http (80)  
Destination port: 53556 (53556)  
[Stream index: 15]  
Sequence number: 1 (relative sequence number)  
Acknowledgement number: 2 (relative ack number)  
Header length: 20 bytes  
Flags: 0x11 (FIN, ACK)  
window size: 5840  
Checksum: 0xb291 [validation disabled]  
[SEQ/ACK analysis]

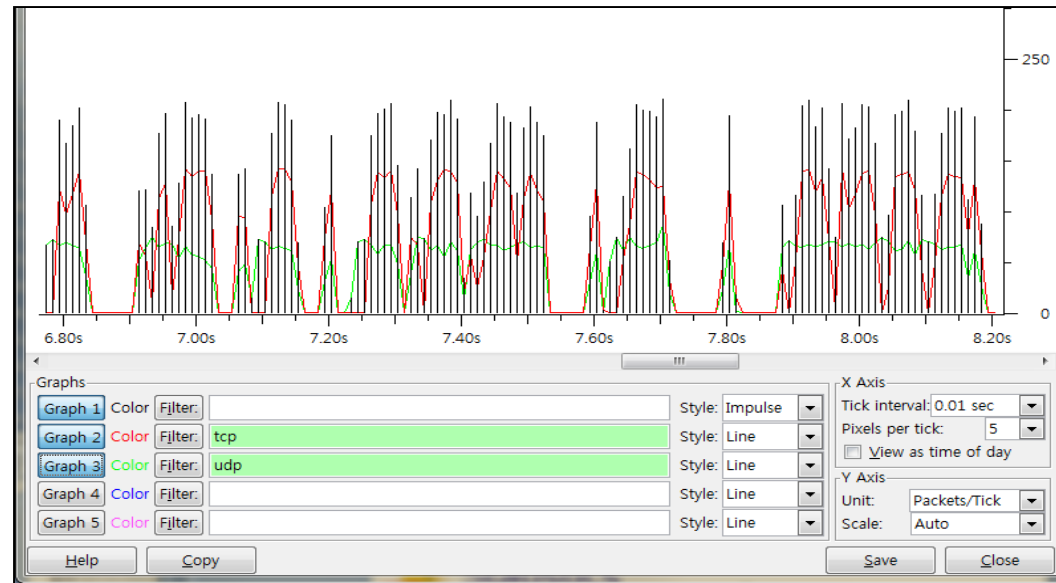
# Demo: Filtering Packets

- Use “Filter”
  - Rule : write field name and relation
    - Check the Expression button → this button shows filtering rules



# Demo: Statistics

- **IO graphs**
  - Visualizing the number of packets (or similar) in time
- **Protocol Hierarchy**
  - The protocol hierarchy of the captured packets
- **Summary**
  - General statistics about the current capture file
- **Flow graph**
  - Show the flow of packets

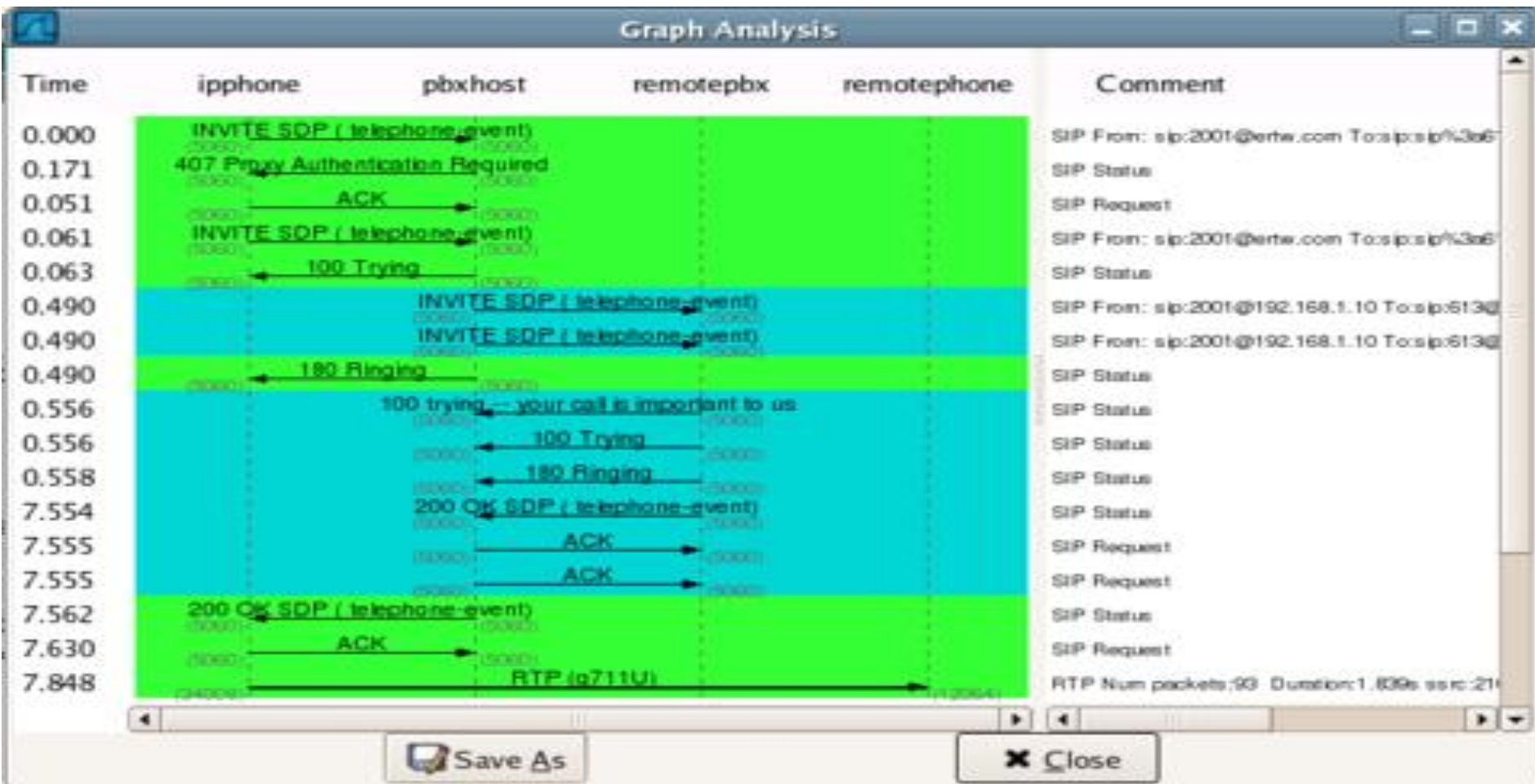


The Protocol Hierarchy Statistics window in Wireshark displays a tree view of the protocol hierarchy for the current capture file. The display filter is set to 'http or dns'. The table below shows the statistics for each protocol in the hierarchy.

Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
Frame	100.00%	367	211211	0.086	0	0	0.000
Ethernet	100.00%	367	211211	0.086	0	0	0.000
Internet Protocol	100.00%	367	211211	0.086	0	0	0.000
Transmission Control Protocol	93.46%	343	207029	0.084	113	82553	0.034
Hypertext Transfer Protocol	62.67%	230	124476	0.051	189	93393	0.038
Compuserve GIF	7.36%	27	17114	0.007	27	17114	0.007
Line-based text data	3.27%	12	12265	0.005	12	12265	0.005
JPEG File Interchange Format	0.27%	1	990	0.000	1	990	0.000
eXtensible Markup Language	0.27%	1	714	0.000	1	714	0.000
User Datagram Protocol	6.54%	24	4182	0.002	0	0	0.000
Domain Name Service	6.54%	24	4182	0.002	24	4182	0.002



# Demo: Graphical Interpretation



# Demo: Dump Data

- Copy & Paste raw data

The screenshot displays a network protocol analyzer interface. The top pane shows a tree view of an SNMP packet structure:

- Simple Network Management Protocol
  - version: version-1 (0)
  - community: public
  - data: get-next-request (1)
    - get-next-request
      - request-id: 750369244
      - error-status: noError (0)
      - error-index: 0
      - variable-bindings: 1 item
        - 1.0.8802.1.1.2.1.4.1.1.5: v... (2.1.4.1.1.5)
          - Object Name: 1.0.8802.1.1.2.1.4.1.1.5
          - value (Null)

The bottom pane shows a hex dump of the selected data. The selected bytes are highlighted in blue:

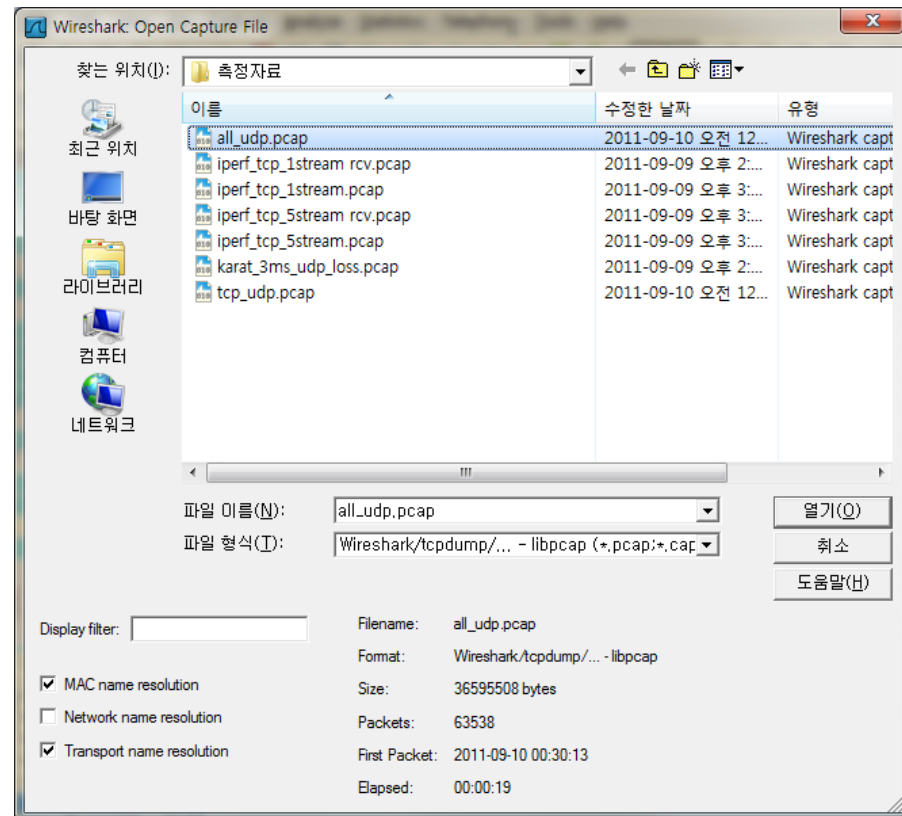
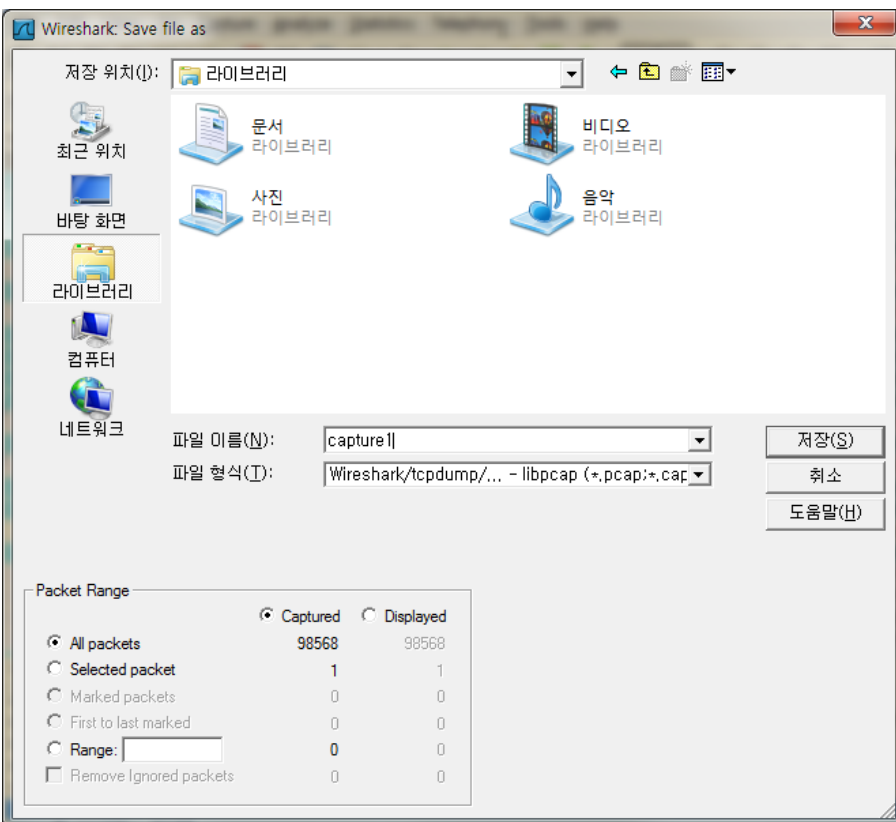
```
0000 00 19 6d 00 97 a3 60 eb 69 de
0010 00 4a 1d 09 00 00 80 11 99 52
0020 01 f1 fd 2d 00 a1 00 36 46 78
0030 06 70 75 62 6c 69 63 a1 1f 02
0040 01 00 02 01 00 30 11 30 0f 06
0050 02 01 04 01 01 05 05 00
```

A context menu is open over the hex dump, with the 'Copy' option selected. The sub-menu shows the following options:

- Description
- Fieldname
- Value
- As Filter
- Bytes (Offset Hex Text)
- Bytes (Offset Hex)
- Bytes (Printable Text Only)
- Bytes (Hex Stream)
- Bytes (Binary Stream)

# Demo: Save & Load

- File → Save as : save captured file
- File → Open \*.pcap file



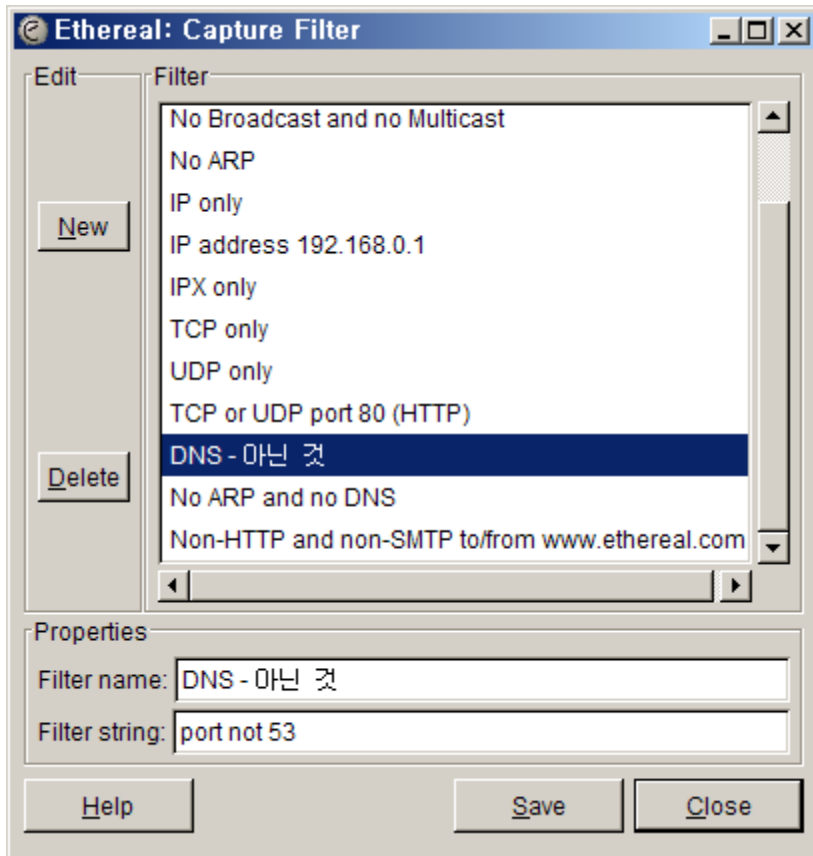
# Capture filter (**libpcap** 방식)

- 모든 HTTP 패킷
  - **tcp port 80**
- Non-HTTP 패킷:
  - not tcp port 80 또는 !tcp port 80 또는 tcp port not 80
- www.inzen.com HTTP browsing
  - tcp port 80 || dst www.inzen.com
- www.inzen.com 아닌 HTTP browsing
  - tcp port 80 and not dst www.inzen.com
- TCP 패킷:
  - tcp 또는 ip proto 6
- TCP SYN 패킷
  - **tcp[tcpflag] & tcp-syn == tcp-syn**

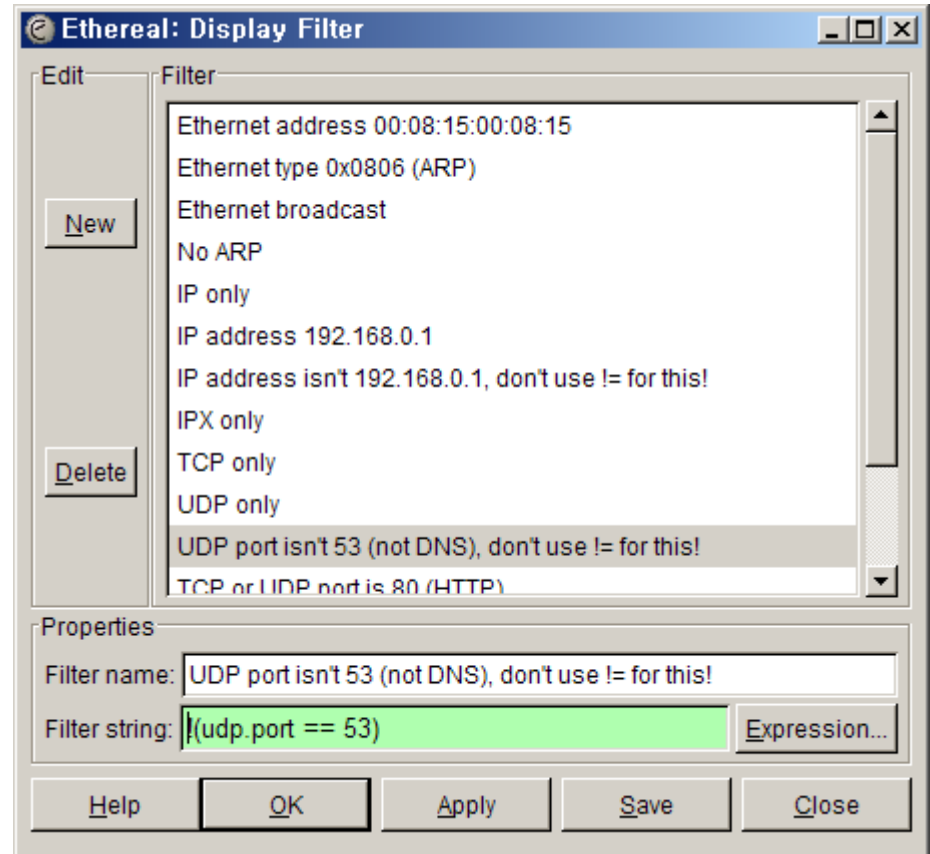
# Capture filter (libpcap 방식)

- IP 패킷 with total length > 255
  - `ip[2:2] > 0xff`
- ICMP echo request and reply
  - `icmp[icmptype] == icmp-echo` or `icmp[icmptype] == icmp-echoreply`
- IP 또는 IPX 패킷
  - `ip` or `ipx`
- IPX 패킷
  - `ipx`
- IPX 패킷 destined for IPX network 00:01:F0:EE
  - `ipx`는 일부 주소만 지정 불가능

# Capture/Display Filter – 주의 필요



Libpcap 입력 방식  
다중 조건 경우 OR 확장



입력과 동시에 자동 문법 체크  
다중 조건 경우 OR로 확장

# 캡처 및 캡처 필터 데모

# Working with captured 패킷

- 패킷 보기
- Display Filter
- Finding 패킷
  - "Find Packet", "Find Next", "Find Previous"
- Go to a specific packet
  - "Go Back", "Go Forward", "Go to Packet", "Go to Corresponding Packet", "Go to First Packet", "Go to Last Packet"
- Marking 패킷 표시
- 시간 표시 형식 and 시간 참조
  - 패킷 시간 참조



# 실제 예제 (Display 필터)

- 스캐닝 – TCP 커넥션
  - `tcp.flags.syn==1 && tcp.flags.ack==1` 또는 `tcp.flags==18`
- 트로이잔 – SubSeven
  - `tcp.port == 27374`
- 트로이잔 – Netbus
  - `tcp.port == 12345 || tcp.port == 12346`
- 웹 – SQL Slammer
  - `udp.port==1434 and ip.len < 384`
- Nmap 스캐닝 확인
  - `nmap -sS -O 10.3.100.231/24`

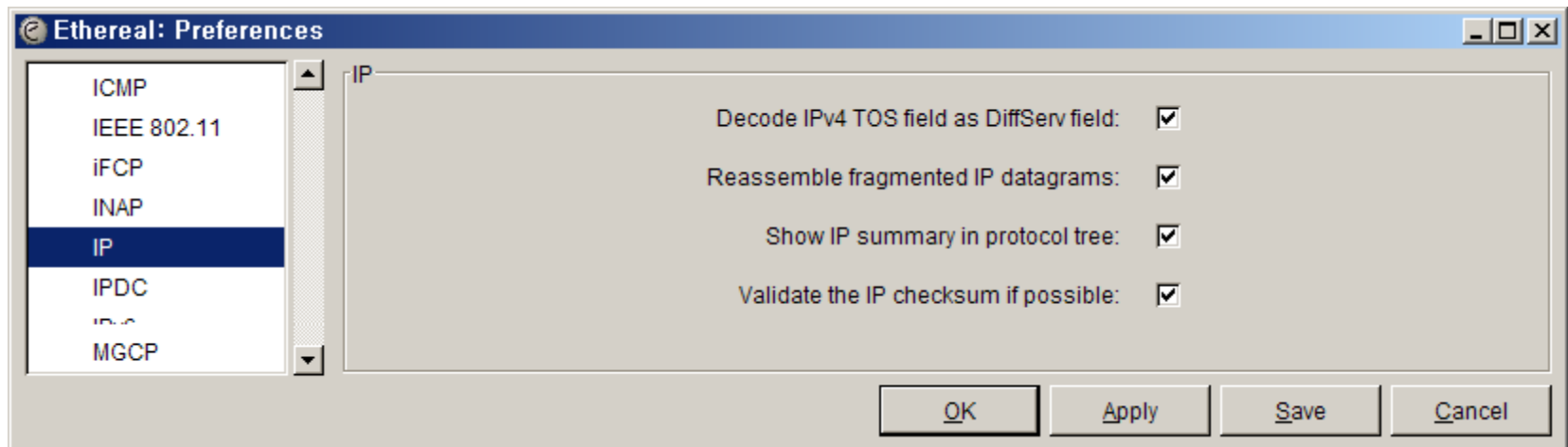
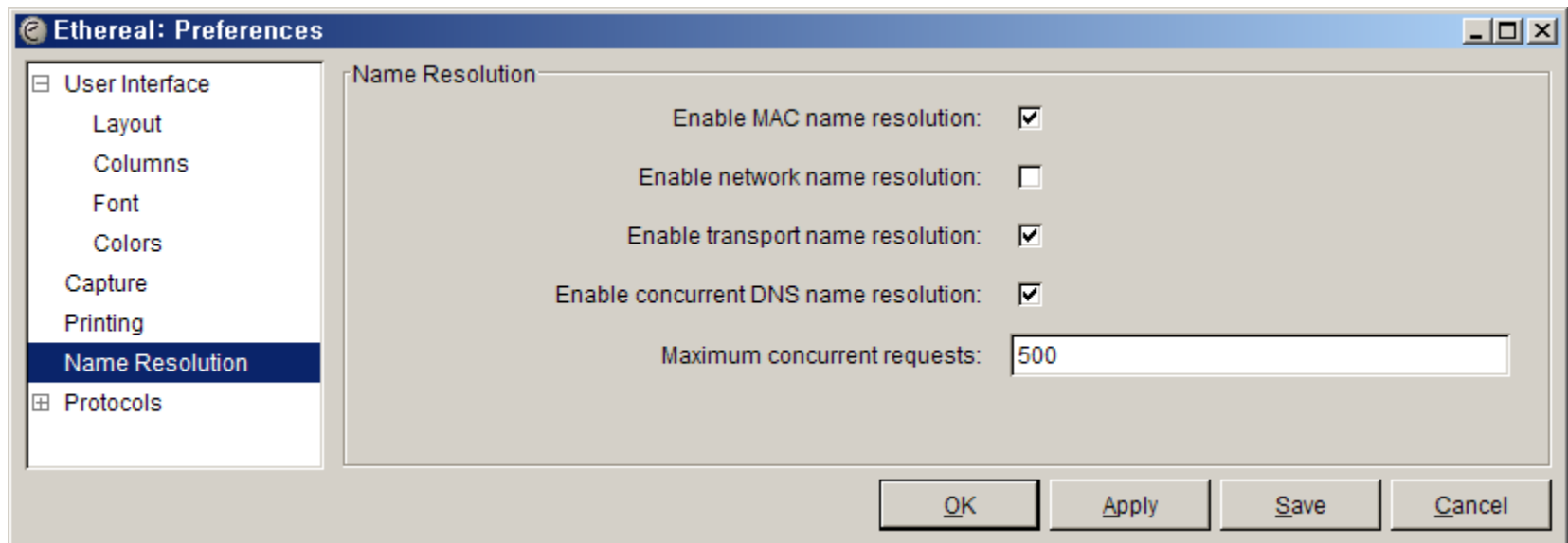
# 화면 필터 데모

# Advanced Features – 화면 표시 제어

- **Following TCP streams**
  - 특정 세션에 대한 프로토콜 흐름 분석
- **Name Resolution**
  - Ethernet name resolution (MAC)
  - IP name resolution (network)
  - TCP/UDP port name resolution (transport)
  - IPX name resolution (network)
- **Packet Reassembling - enabled by default ?**

소스 [epan/dissectors/](http://epan/dissectors/)

# Name resolution, Fragment Reassembly



# 캡처 화면 – 필터, 이름 변환, 재조함

(Untitled) - Ethereal

File Edit View Go Capture Analyze Statistics Help

Filter: (ip.addr eq 10.3.100.231 and ip.addr eq 61.106.27.55) and (tcp.port ... Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
43	4.937723	10.3.100.231	tech.realmedia.co.kr	TCP	1302 > http [SYN] Seq=0 Ack=0 win=65535 Len=0
46	4.948818	tech.realmedia.co.kr	10.3.100.231	TCP	http > 1302 [SYN, ACK] Seq=0 Ack=1 win=5840 Len=0
47	4.948877	10.3.100.231	tech.realmedia.co.kr	TCP	1302 > http [ACK] Seq=1 Ack=1 win=65535 Len=0
48	4.949038	10.3.100.231	tech.realmedia.co.kr	HTTP	GET /RMTECH/imp.ads/www.bccard.co.kr/member_a@
54	4.967822	tech.realmedia.co.kr	10.3.100.231	TCP	http > 1302 [ACK] Seq=1 Ack=378 win=6432 Len=0
55	4.970241	tech.realmedia.co.kr	10.3.100.231	HTTP	HTTP/1.1 200 OK (GIF89a)
56	4.970877	tech.realmedia.co.kr	10.3.100.231	TCP	http > 1302 [FIN, ACK] Seq=470 Ack=378 win=6432

Frame 43 (62 bytes on wire, 62 bytes captured)

Ethernet II, Src: 10.3.100.231 (00:03:0d:1b:ed:0c), Dst: 10.3.1.1 (00:48:54:6d:e6:4e)

Internet Protocol, Src: 10.3.100.231 (10.3.100.231), Dst: tech.realmedia.co.kr (61.106.27.55)

Transmission Control Protocol, Src Port: 1302 (1302), Dst Port: http (80), Seq: 0, Ack: 0, Len: 0

Source port: 1302 (1302)

Destination port: http (80)

Sequence number: 0 (relative sequence number)

Header length: 28 bytes

Flags: 0x0002 (SYN)

window size: 65535

checksum: 0xa2f5 [correct]

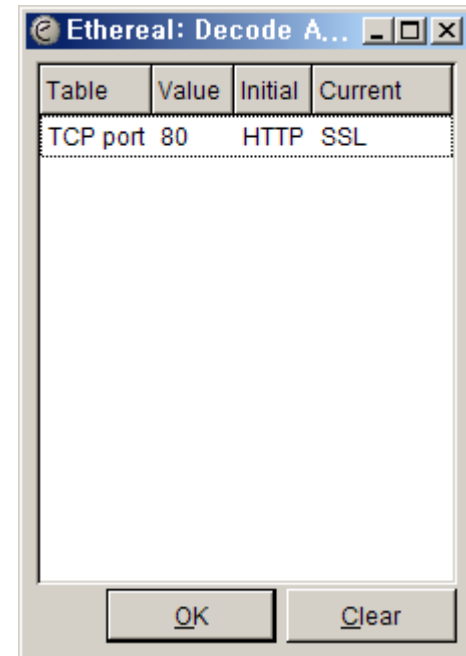
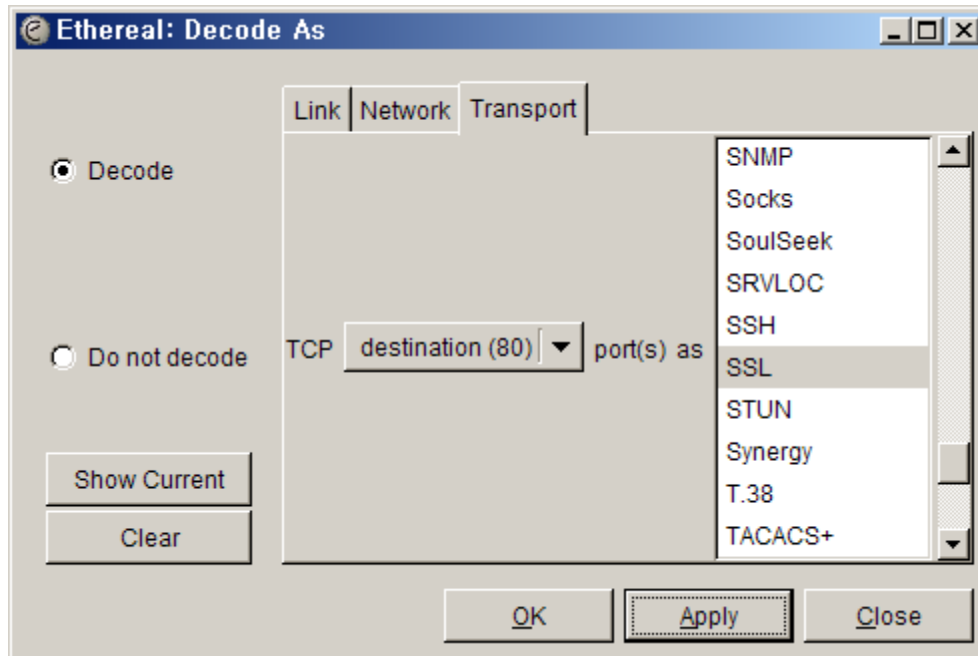
options: (8 bytes)

```

0000  00 48 54 6d e6 4e 00 03 0d 1b ed 0c 08 00 45 00  .HTm.N.. ....E.
0010  00 30 18 7b 40 00 80 06 1a c2 0a 03 64 e7 3d 6a  .0.{@... ....d.=j
0020  1b 37 05 16 00 50 02 3a 10 ff 00 00 00 00 70 02  .7...P.: .....p.
0030  ff ff a2 f5 00 00 02 04 05 b4 01 01 04 02      .....
    
```

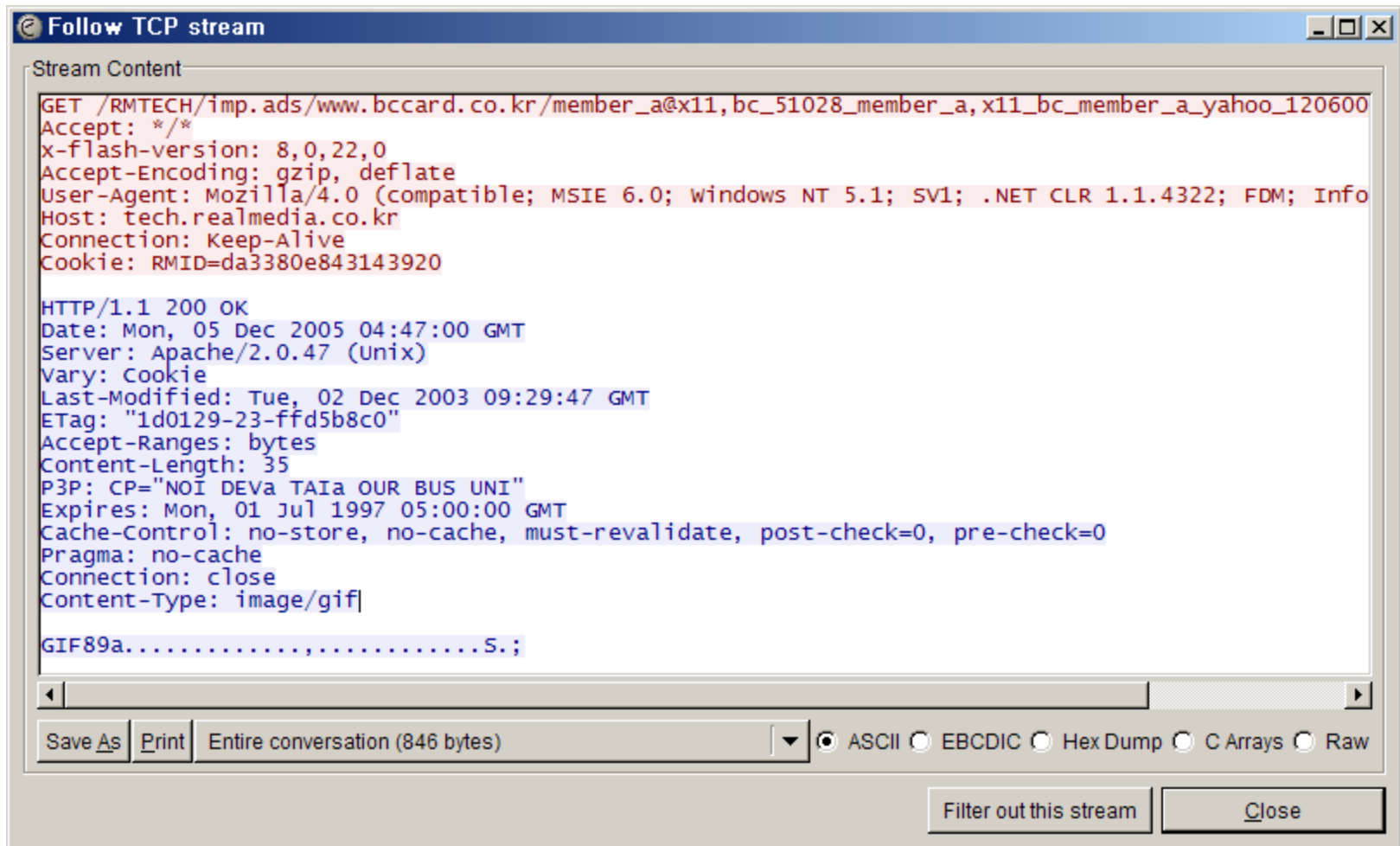
File: "t:\TempletherXXXXa0405" P: 282 D: 10 M: 3 Drops: 0

# 디코더 선택 (Dissector)



소스 [epan/dissectors/](http://epan/dissectors/)

# Follow stream



# More Resources

- **Search “wireshark tutorial”**
- **<http://wiki.wireshark.org>**
- **<http://wiki.wireshark.org/SampleCaptures>**