

Wireshark : UDP

Aim

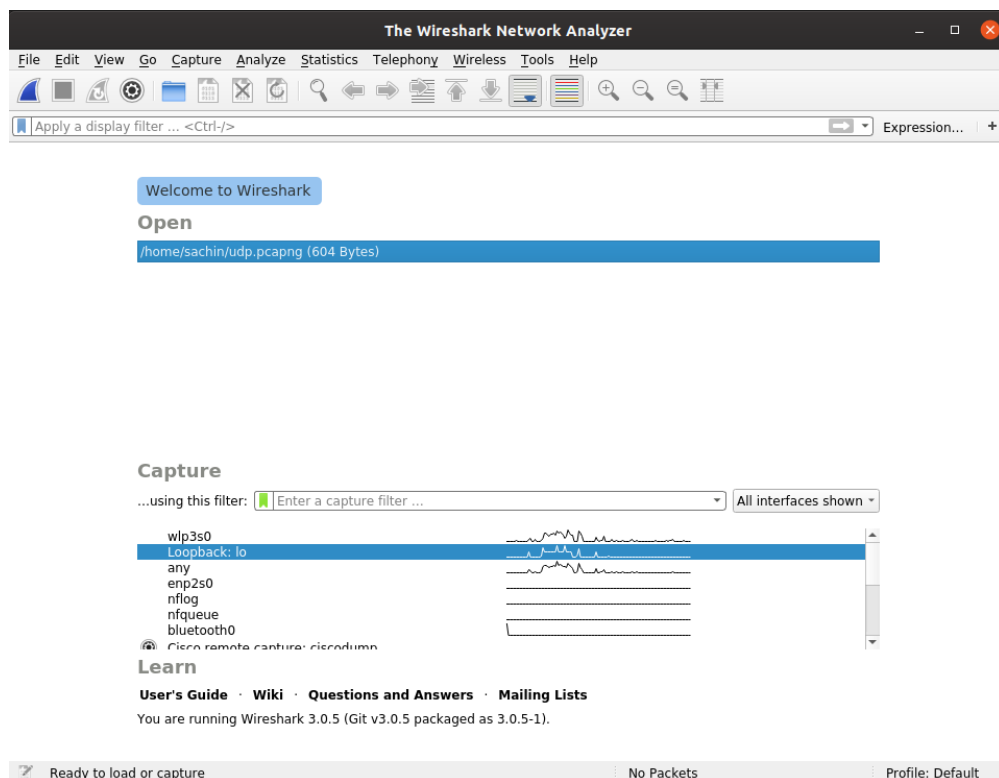
Using Wireshark observe data transferred in client server communication using UDP and identify the UDP datagram.

Theory

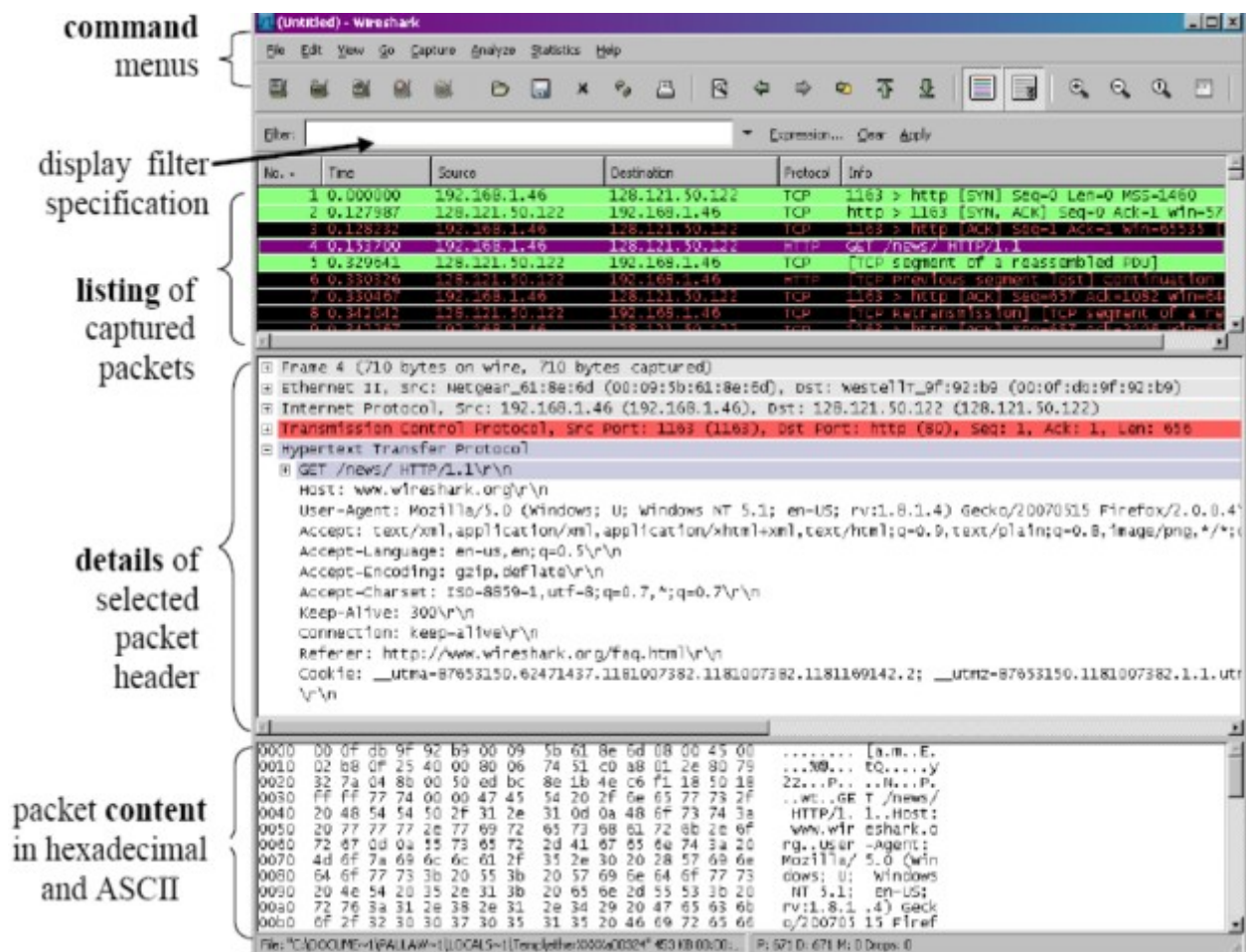
Wireshark

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible. Some of the main uses include:

- Network administrators use it to troubleshoot network problems
- Developers use it to debug protocol implementations
- QA engineers use it to verify network applications
- Network security engineers use it to examine security problems
- People use it to learn network protocol internals



Wireshark Interface



Wireshark components

Capturing Packets: After downloading and installing Wireshark, one can launch and select a network interface under Capture to start capturing packets on that interface. That is, if you want to capture traffic on your bluetooth interface, select the bluetooth interface. Advanced features can be configured by clicking Options in Capture tab. 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. If you have promiscuous mode enabled(default), you'll also see all the other packets on the network instead of only packets addressed to your network adapter.

Filtering Packets: For inspecting something specific, filtering comes in handy. 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.

Output

The screenshot shows the Wireshark interface with the title bar "*Loopback: lo". The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The toolbar contains various icons for file operations, capture control, and analysis. The display filter bar shows "Apply a display filter ... <Ctrl-/>". The packet list pane shows three captured packets:

Time	Source	Destination	Protocol	Length	Info
0.000000000	127.0.0.1	127.0.0.1	UDP	44	41420 → 5005 Len=2
3.543359469	127.0.0.1	127.0.0.1	UDP	47	59019 → 5005 Len=5
9.671771571	127.0.0.1	127.0.0.1	UDP	53	43550 → 5005 Len=11

The packet details pane for the selected packet (Time 9.671771571) shows:

- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
- User Datagram Protocol, Src Port: 41420, Dst Port: 5005
- Data (2 bytes)
 - Data: 6869

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

```
0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00  .....E:
0010  00 1e 16 3b 40 00 40 11 26 92 7f 00 00 01 7f 00  ...;@. &.....
0020  00 01 a1 cc 13 8d 00 0a fe 1d 68 69                .....hi
```

The screenshot shows the Wireshark interface with the title bar "*Loopback: lo". The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The display filter bar shows "Apply a display filter ... <Ctrl-/>". The packet list pane shows three captured packets:

Time	Source	Destination	Protocol	Length	Info
0.000000000	127.0.0.1	127.0.0.1	UDP	44	41420 → 5005 Len=2
3.543359469	127.0.0.1	127.0.0.1	UDP	47	59019 → 5005 Len=5
9.671771571	127.0.0.1	127.0.0.1	UDP	53	43550 → 5005 Len=11

The packet details pane for the selected packet (Time 9.671771571) shows:

- Frame 2: 47 bytes on wire (376 bits), 47 bytes captured (376 bits) on interface 0
- Interface 0 (lo)
 - Interface name: lo
 - Encapsulation type: Ethernet (1)
 - Arrival Time: Apr 14, 2020 13:06:05.713697470 IST
 - [Time shift for this packet: 0.000000000 seconds]
 - Epoch Time: 1586849765.713697470 seconds
 - [Time delta from previous captured frame: 3.543359469 seconds]
 - [Time delta from previous displayed frame: 3.543359469 seconds]
 - [Time since reference or first frame: 3.543359469 seconds]
 - Frame Number: 2
 - Frame Length: 47 bytes (376 bits)
 - Capture Length: 47 bytes (376 bits)
 - [Frame is marked: False]
 - [Frame is ignored: False]
 - [Protocols in frame: eth:ethertype:ip:udp:data]
 - [Coloring Rule Name: UDP]
- Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
 - Destination: 00:00:00:00:00:00 (00:00:00:00:00:00)
 - Address: 00:00:00:00:00:00 (00:00:00:00:00:00)
 - ...0. = LG bit: Globally unique address (factory default)
 - ...0. = IG bit: Individual address (unicast)
 - Source: 00:00:00:00:00:00 (00:00:00:00:00:00)
 - Address: 00:00:00:00:00:00 (00:00:00:00:00:00)
 - ...0. = LG bit: Globally unique address (factory default)
 - ...0. = IG bit: Individual address (unicast)
 - Type: IPv4 (0x0800)
- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 - 0100 = Version: 4
 - ...0101 = Header Length: 20 bytes (5)
 - Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

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

```
0000  00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00  .....E:
0010  00 21 16 da 40 00 40 11 25 f9 7f 00 00 01 7f 00  ...!@. %.....
0020  00 01 e6 8b 13 8d 00 0d fe 20 68 65 6c 6c 6f      .....hello
```

On the right, two terminal windows are shown. The top terminal window shows the output of a Python script named "udclient.py":

```
[sachin@sachin ~]$ python udclient.py
UDP target IP: 127.0.0.1
UDP target port: 5005
hi
message: hi
hello
message: hello
how are you
message: how are you
^CTraceback (most recent call last):
  File "udclient.py", line 7, in <module>
    MESSAGE= raw_input()
KeyboardInterrupt
[sachin@sachin ~]$
```

The bottom terminal window shows the output of a Python script named "udpserver.py":

```
[sachin@sachin ~]$ python udpserver.py
received message: hi
received message: hello
received message: how are you
^X^CTraceback (most recent call last):
  File "udpserver.py", line 7, in <module>
    data, addr = sock.recvfrom(1024) # buffer size is 1024 bytes
KeyboardInterrupt
[sachin@sachin ~]$
```

The top screenshot shows the Wireshark interface with a capture on the loopback interface 'lo'. Three packets are captured, all of type UDP. The selected packet (No. 3) is expanded to show its details: Ethernet II, Internet Protocol Version 4, and User Datagram Protocol (Source Port: 59019, Destination Port: 5005). The packet data is 5 bytes long. To the right, a terminal window shows the execution of a Python script 'udclient.py'. The script sends three messages: 'hi', 'hello', and 'how are you'. The terminal output shows these messages being sent successfully.

The bottom screenshot shows the same Wireshark interface, but the selected packet (No. 3) is expanded further to show the 'Data' field. The data is 11 bytes long and contains the ASCII string 'hello'. The terminal window on the right shows the execution of a Python script 'udpserver.py'. The script receives three messages: 'hi', 'hello', and 'how are you'. The terminal output shows these messages being received successfully.

Result

Wireshark was installed and Packet transfer using UDP was observed .

