EXPERIMENT - XV
WIRESHARK : UDP

April 12, 2020

ADITHYA D RAJAGOPAL
ROLL NO : 9
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COLLEGE OF ENGINEERING TRIVANDRUM

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

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.

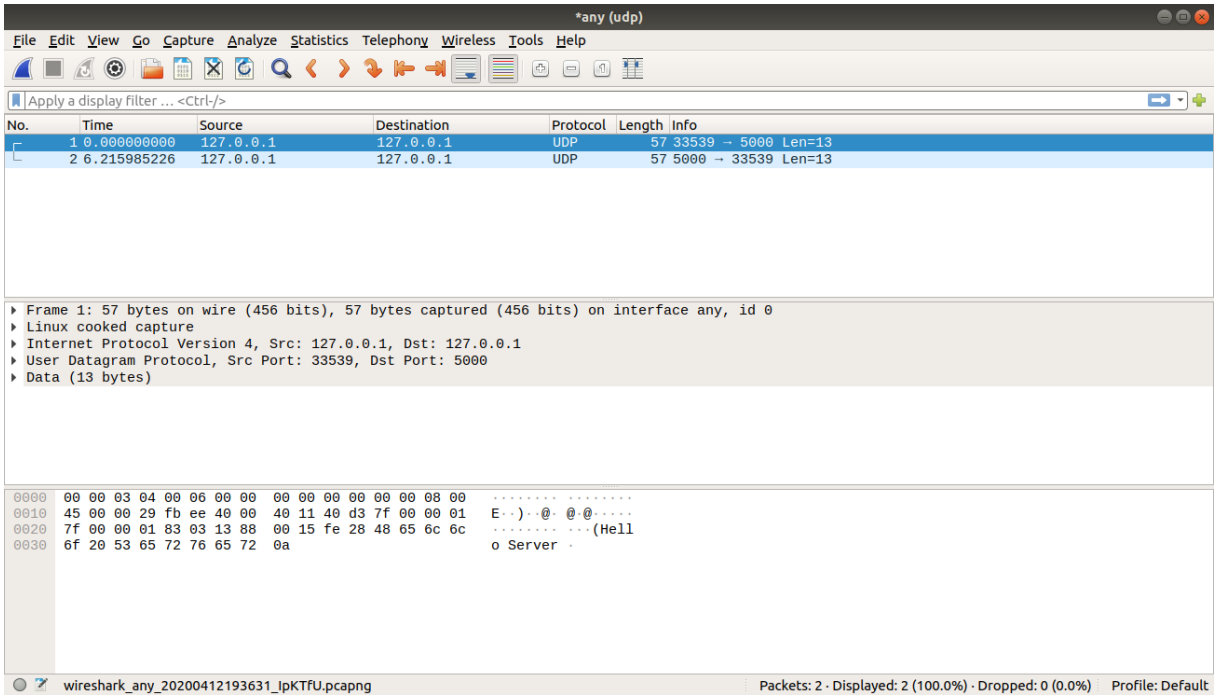
Filtering Packets

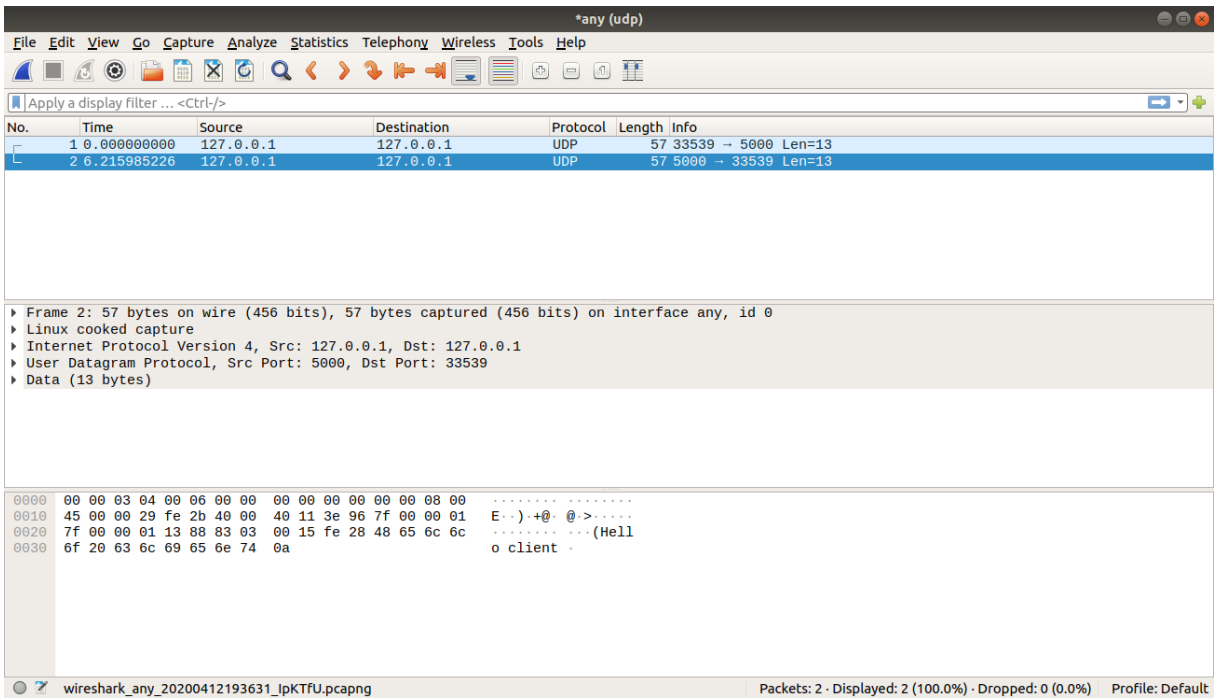
For inspecting something specific, filtering comes in handy. The most basic way to apply a filter is by typing it into the filter box at the top of the window and clicking Apply (or pressing Enter). For example, type "dns" and you'll see only DNS packets.

PROCEDURE

1. Run a basic UDP client-server program.
2. As a result of the process taking place, UDP packets will be sent or received.
3. These packets can be sniffed by using wireshark.
4. Upon applying the filter, we can specifically capture the packets which are a part of the ongoing udp packet transmission.

OUTPUT


*any (udp)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
1 0.000000000 127.0.0.1 127.0.0.1 UDP 57 33539 → 5000 Len=13
2 6.215985226 127.0.0.1 127.0.0.1 UDP 57 5000 → 33539 Len=13
Frame 1: 57 bytes on wire (456 bits), 57 bytes captured (456 bits) on interface any, id 0
Linux cooked capture
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 33539, Dst Port: 5000
Data (13 bytes)
0000 00 00 03 04 00 06 00 00 00 00 00 00 08 00
0010 45 00 00 29 fb ee 40 00 40 11 40 d3 7f 00 00 01
0020 7f 00 00 01 83 03 13 88 00 15 fe 28 48 65 6c 6c
0030 6f 20 53 65 72 76 65 72 0a
o Server
wireshark_any_20200412193631_IpKTfU.pcapng
Packets: 2 · Displayed: 2 (100.0%) · Dropped: 0 (0.0%)
Profile: Default


*any (udp)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
1 0.000000000 127.0.0.1 127.0.0.1 UDP 57 33539 → 5000 Len=13
2 6.215985226 127.0.0.1 127.0.0.1 UDP 57 5000 → 33539 Len=13
Frame 2: 57 bytes on wire (456 bits), 57 bytes captured (456 bits) on interface any, id 0
Linux cooked capture
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 5000, Dst Port: 33539
Data (13 bytes)
0000 00 00 03 04 00 06 00 00 00 00 00 00 08 00
0010 45 00 00 29 fe 2b 40 00 40 11 3e 96 7f 00 00 01
0020 7f 00 00 01 13 88 83 03 00 15 fe 28 48 65 6c 6c
0030 6f 20 63 6c 69 65 6e 74 0a
o client
wireshark_any_20200412193631_IpKTfU.pcapng
Packets: 2 · Displayed: 2 (100.0%) · Dropped: 0 (0.0%)
Profile: Default

RESULT

The data transferred in client-server communication have been observed using wireshark and the UDP datagram has been identified.