

Exercise 1

Traffic capture

Selected enx0c37965f8a22 to capture from

Captured Packets:

- Domain Name System (Query)
- Address Resolution Protocol (request and reply)

The screenshot shows the Wireshark network protocol analyzer interface. The title bar indicates the capture file is 'enx0c37965f8a22 capture.pcapng'. The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for packet manipulation. A filter bar at the top of the packet list shows 'Apply a display filter ... <Ctrl-/>'. The packet list table has columns for No., Time, Source, Destination, Protocol, Length, and Info. It displays 16 packets, with packets 11 and 12 highlighted in yellow. Packet 11 is an ARP request from BizlinkT_5f:8a:22 to Raspberr_8d:d2:a6. Packet 12 is an ARP reply from Raspberr_8d:d2:a6 to BizlinkT_5f:8a:22. The packet details pane for packet 11 is expanded, showing Ethernet II, Internet Protocol Version 4, User Datagram Protocol, and Domain Name System (query) layers. The packet bytes pane shows the raw data in hexadecimal and ASCII. The status bar at the bottom indicates 'Packets: 32 · Displayed: 32 (100.0%) Profile: Default'.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.10.2	1.1.1.1	DNS	81	Standard query 0xeaf1
2	0.000000273	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x2f80
3	5.005597318	192.168.10.2	1.1.1.1	DNS	81	Standard query 0xaa33
4	5.005597681	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x33ba
5	10.010683306	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x8e82
6	10.010683610	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x33ba
7	15.016349574	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x4bae
8	15.016349891	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x8efb
9	20.021286634	192.168.10.2	1.1.1.1	DNS	81	Standard query 0xa545
10	20.021286967	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x8efb
11	20.079066423	Raspberr_8d:d2:a6	BizlinkT_5f:8a:22	ARP	60	Who has 192.168.10.1?
12	20.079090043	BizlinkT_5f:8a:22	Raspberr_8d:d2:a6	ARP	42	192.168.10.1 is at 0c
13	25.026979968	192.168.10.2	1.1.1.1	DNS	81	Standard query 0xaae9
14	25.026980289	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x8256
15	30.032043982	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x0501
16	30.032044278	192.168.10.2	1.1.1.1	DNS	81	Standard query 0x0256

Frame 1: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface enx0c37965f8a22, id 0
Ethernet II, Src: Raspberr_8d:d2:a6 (e4:5f:01:8d:d2:a6), Dst: BizlinkT_5f:8a:22 (0c:37:96:5f:8a:22)
Internet Protocol Version 4, Src: 192.168.10.2, Dst: 1.1.1.1
User Datagram Protocol, Src Port: 41014, Dst Port: 53
Domain Name System (query)

```
0000  0c 37 96 5f 8a 22 e4 5f 01 8d d2 a6 08 00 45 00  .7._."_ .....E.
0010  00 43 81 8d 40 00 40 11 ec 70 c0 a8 0a 02 01 01  .C..@.@. .p.....
0020  01 01 a0 36 00 35 00 2f 92 96 ea f1 01 00 00 01  ...6.5./ .....
0030  00 00 00 00 00 00 01 32 06 64 65 62 69 61 6e 04  ....2 .debian.
0040  70 6f 6f 6c 03 6e 74 70 03 6f 72 67 00 00 01 00  pool.ntp .org...
0050  01
```

enx0c37965f8a22 capture.pcapng Packets: 32 · Displayed: 32 (100.0%) Profile: Default

Selected enp0s31f6

Apply display filter HTTP

start capture and stop

http capture.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
112	1.085599333	10.200.18.81	10.200.18.105	HTTP	879	HTTP/1.1 401 Unauthorized
115	1.088313009	10.200.18.82	10.200.18.105	HTTP	879	HTTP/1.1 401 Unauthorized
121	1.123496403	10.200.18.81	10.200.18.105	HTTP	634	HTTP/1.1 200 OK (text/plai
122	1.123999658	10.200.18.82	10.200.18.105	HTTP	637	HTTP/1.1 200 OK (text/plai
618	4.204400399	10.200.18.81	10.200.18.105	HTTP	879	HTTP/1.1 401 Unauthorized
623	4.207539714	10.200.18.82	10.200.18.105	HTTP	879	HTTP/1.1 401 Unauthorized
635	4.243907595	10.200.18.81	10.200.18.105	HTTP	634	HTTP/1.1 200 OK (text/plai
637	4.246870938	10.200.18.82	10.200.18.105	HTTP	637	HTTP/1.1 200 OK (text/plai

Frame 112: 879 bytes on wire (7032 bits), 879 bytes captured (7032 bits) on interface enp0s31f6, id 0

- Ethernet II, Src: AxisComm_b0:2f:b8 (ac:cc:8e:b0:2f:b8), Dst: ExtronEl_15:a7:06 (00:05:a6:15:a7:06)
- Internet Protocol Version 4, Src: 10.200.18.81, Dst: 10.200.18.105
- Transmission Control Protocol, Src Port: 80, Dst Port: 42024, Seq: 1, Ack: 154, Len: 813
- Hypertext Transfer Protocol
- Line-based text data: text/html (12 lines)

```
0000 00 05 a6 15 a7 06 ac cc 8e b0 2f b8 08 00 45 00 ..... /...E.
0010 03 61 27 7f 40 00 40 06 d5 ce 0a c8 12 51 0a c8 .a'..@.. ....Q..
0020 12 69 00 50 a4 28 ec 07 65 83 20 d3 c9 b7 80 18 .i.P.(.. e.....
0030 03 ab 69 a7 00 00 01 01 08 0a 03 0c 42 fd 49 1e ..i.....B.I.
0040 3c ff 48 54 54 50 2f 31 2e 31 20 34 30 31 20 55 <.HTTP/1 .1 401 U
0050 6e 61 75 74 68 6f 72 69 7a 65 64 0d 0a 44 61 74 nauthori zed..Dat
0060 65 3a 20 4d 6f 6e 2c 20 30 39 20 4a 75 6e 20 32 e: Mon, 09 Jun 2
0070 30 32 35 20 31 30 3a 33 32 3a 34 30 20 47 4d 54 025 10:3 2:40 GMT
0080 0d 0a 53 65 72 76 65 72 3a 20 41 70 61 63 68 65 ..Server : Apache
0090 2f 32 2e 34 2e 35 34 20 28 55 6e 69 78 29 20 4f /2.4.54 (Unix) 0
00a0 70 65 6e 53 53 4c 2f 31 2e 31 2e 31 73 0d 0a 58 penSSL/1 .1.1s.X
00b0 2d 43 6f 6e 74 65 6e 74 2d 54 79 70 65 2d 4f 70 -Content -Type-Op
```

Hypertext Transfer Protocol: Protocol Packets: 975 · Displayed: 8 (0.8%) Profile: Default

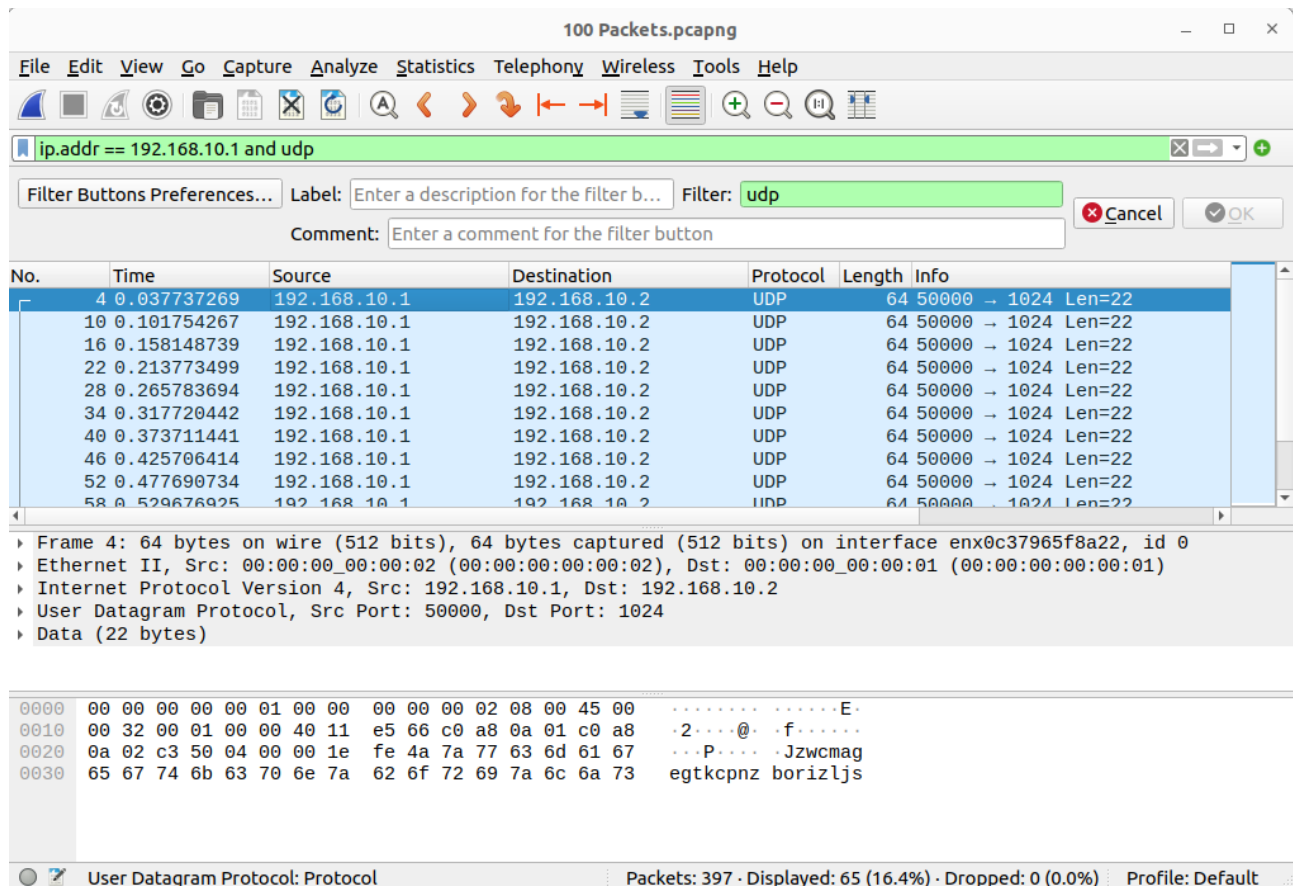
Capture Raspberry Pi using tcp dump

```
pi@p4pi:~$ sudo tcpdump -i eth0 -c 10 -w captured.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10 packets captured
10 packets received by filter
0 packets dropped by kernel
pi@p4pi:~$ tcpdump -r captured.pcap
reading from file captured.pcap, link-type EN10MB (Ethernet), snapshot length 262144
18:00:09.423744 IP 192.168.10.2.ssh > 192.168.10.1.44982: Flags [P.], seq 1023742234:1023742278, ack 2952574732, win 501, options [nop,nop,TS val 3883994997 ecr 2034911232], length 44
18:00:09.423859 IP 192.168.10.2.ssh > 192.168.10.1.44982: Flags [P.], seq 44:96, ack 1, win 501, options [nop,nop,TS val 3883994998 ecr 2034911232], length 52
18:00:09.423948 IP 192.168.10.2.ssh > 192.168.10.1.44982: Flags [P.], seq 96:164, ack 1, win 501, options [nop,nop,TS val 3883994998 ecr 2034911232], length 68
18:00:09.424039 IP 192.168.10.2.ssh > 192.168.10.1.44982: Flags [P.], seq 164:232, ack 1, win 501, options [nop,nop,TS val 3883994998 ecr 2034911232], length 68
18:00:09.424303 IP 192.168.10.1.44982 > 192.168.10.2.ssh: Flags [.], ack 44, win 1632, options [nop,nop,TS val 2034911284 ecr 3883994997], length 0
18:00:09.424304 IP 192.168.10.1.44982 > 192.168.10.2.ssh: Flags [.], ack 96, win 1632, options [nop,nop,TS val 2034911284 ecr 3883994998], length 0
18:00:09.424304 IP 192.168.10.1.44982 > 192.168.10.2.ssh: Flags [.], ack 164, win 1632, options [nop,nop,TS val 2034911284 ecr 3883994998], length 0
18:00:09.424357 IP 192.168.10.1.44982 > 192.168.10.2.ssh: Flags [.], ack 232, win 1632, options [nop,nop,TS val 2034911284 ecr 3883994998], length 0
18:00:11.051235 IP 192.168.10.2.57856 > 1.1.1.1.domain: 3372+ A? 0.debian.pool.ntp.org. (39)
18:00:11.051363 IP 192.168.10.2.36585 > 1.1.1.1.domain: 28013+ AAAA? 0.debian.pool.ntp.org. (39)
pi@p4pi:~$
```

Sending Traffic

1. Can you define a filter to capture only the packets you sent?

Using `ip.addr == 192.168.10.1` and `udp`



The image shows the Wireshark network protocol analyzer interface. At the top, the title bar reads "100 Packets.pcapng". The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons. A filter bar at the top displays the active filter: `ip.addr == 192.168.10.1 and udp`. Below the filter bar, there is a "Filter Buttons Preferences..." button, a "Label:" field with the text "Enter a description for the filter b...", a "Filter:" field with the text "udp", and "Cancel" and "OK" buttons. A "Comment:" field with the text "Enter a comment for the filter button" is also present. The main packet list table shows the following data:

No.	Time	Source	Destination	Protocol	Length	Info
4	0.037737269	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
10	0.101754267	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
16	0.158148739	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
22	0.213773499	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
28	0.265783694	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
34	0.317720442	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
40	0.373711441	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
46	0.425706414	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
52	0.477690734	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22
58	0.529676925	192.168.10.1	192.168.10.2	UDP	64	50000 → 1024 Len=22

Below the packet list, the details pane shows the following information for the selected packet (Frame 4):

- Frame 4: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface `enx0c37965f8a22`, id 0
- Ethernet II, Src: 00:00:00_00:00:02 (00:00:00:00:00:02), Dst: 00:00:00_00:00:01 (00:00:00:00:00:01)
- Internet Protocol Version 4, Src: 192.168.10.1, Dst: 192.168.10.2
- User Datagram Protocol, Src Port: 50000, Dst Port: 1024
- Data (22 bytes)

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

```
0000  00 00 00 00 00 01 00 00 00 00 02 08 00 45 00  .....E.
0010  00 32 00 01 00 00 40 11 e5 66 c0 a8 0a 01 c0 a8  .2...@. .f.....
0020  0a 02 c3 50 04 00 00 1e fe 4a 7a 77 63 6d 61 67  ...P.... .Jzwcma
0030  65 67 74 6b 63 70 6e 7a 62 6f 72 69 7a 6c 6a 73  egtkcpnz borizljs
```

At the bottom, the status bar shows: "User Datagram Protocol: Protocol", "Packets: 397 · Displayed: 65 (16.4%) · Dropped: 0 (0.0%)", and "Profile: Default".

2. What is the packet size?

512bits

3. What is the protocol used (TCP or UDP)?

UDP

Modified Script `send_TCP.py`

```
Open  send_TPC.py  Save  -/CWM-Prog/NetL/assignment1

1#!/usr/bin/python
2
3from scapy.all import Ether, IP, sendp, get_if_hwaddr, get_if_list, TCP, Raw, UDP
4import sys
5import random, string
6
7
8def randomword(length):
9    return ''.join(random.choice(string.ascii_lowercase) for i in range(length))
10
11def send_random_traffic(num_packets, interface, src_ip, dst_ip):
12    dst_mac = "00:00:00:00:00:01"
13    src_mac = "CA:FE:CA:FE:CA:FE"
14    total_pkts = 0
15    port = 1024
16    for i in range(num_packets):
17        data = randomword(22)
18        p = Ether(dst=dst_mac,src=src_mac)/IP(dst=dst_ip,src=src_ip)
19        p = p/TCP(sport=5555,dport=port)/Raw(load=data)
20        sendp(p, iface = interface, inter = 0.01)
21        # If you want to see the contents of the packet, uncomment the line below
22        # print(p.show())
23        total_pkts += 1
24    print("Sent %s packets in total" % total_pkts)
25
26if __name__ == '__main__':
27    if len(sys.argv) < 5:
28        print("Usage: python send.py number_of_packets interface_name src_ip_address dst_ip_address")
29        sys.exit(1)
30    else:
31        num_packets = sys.argv[1]
32        interface = sys.argv[2]
33        src_ip = sys.argv[3]
34        dst_ip = sys.argv[4]
35        send_random_traffic(int(num_packets), interface, src_ip, dst_ip)
```