EL5373

INTERNET ARCHITECTURE AND PROTOCOLS

Runze Dong

N10264442

rd1711@nyu.edu

Workstation: APAH Othello_I

MAC: f8:0f:41:c4:7f:aa

Lab Report 5

Due Nov 11, 2014

Exercise 1

Explain the operation of each command.

iperf3 -s -p 5555

Start server listening on port 5555

Iperf3 -c 192.168.1.17 -p 8911 -u

Run UDP with remote server (192.168.1.17 port 8911).

Iperf3 -c 192.168.1.17 -p 8911 -n 10K -l 200

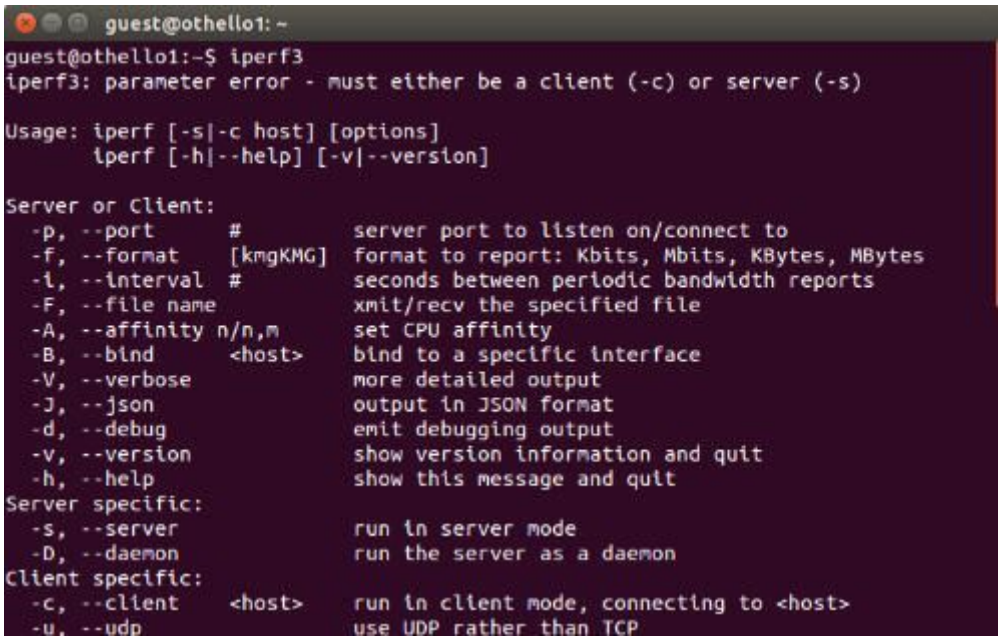
Use UDP with remote server (192.168.1.17 port 8911), to transmit 10K bytes data with the length of buffer, 200.

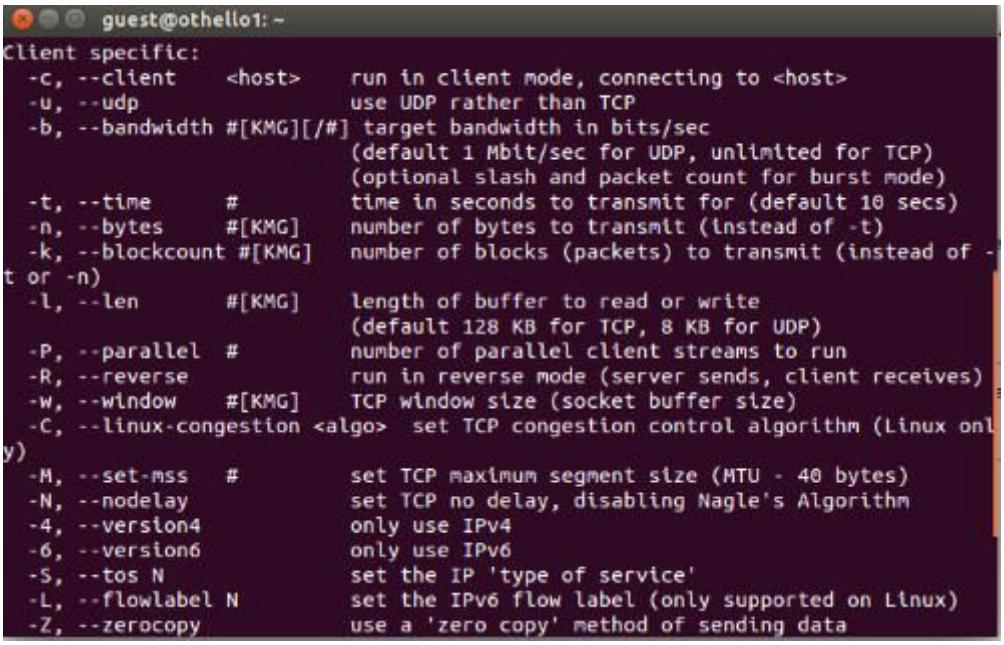
For the second and/or the third command, does it make a difference if there is a server running? Yes.

Exercise 2

Study various options associated with the iperf3 program.

We can learn some more detailed commands from man page of iperf3.

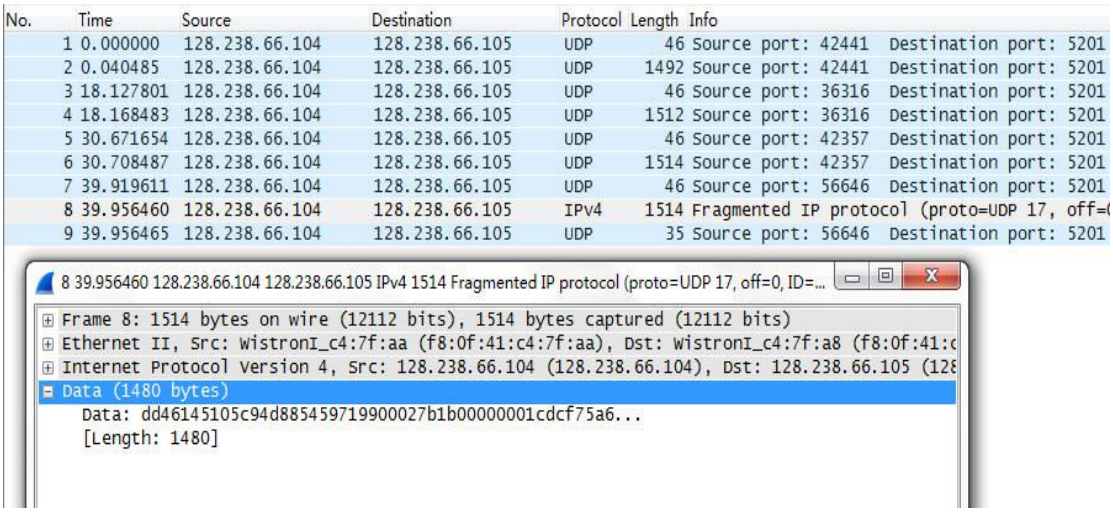




Exercise 3

We find the maximum value of size of one UDP data is **1480 bytes**, which can be sent without IP fragmentation.

Justify: Firstly, we use ifconfig command to find out **MTU of Ethernet interface is 1500 bytes**. In order to avoid IP fragmentation, this IP datagram should be equal or less than MTU. And also UDP is encapsulated into IP datagram. So there are 1472 bytes data left since a **8 bytes UDP header and 20 bytes IP header** should be contained in Ethernet frame. However, when IP fragmentation occurs, only the first fragment has the UDP header. So the maximum is 1480 bytes.



Exercise 4

Explain the tcpdump output in terms of the IP header fields that are used in fragmentation. When IP fragmentation occurs, only the first fragment has the UDP header. How do you verify this fact from the tcpdump output?

From exe4.txt output we dump, the **10000 bytes UDP data** are fragmented into **7 IP fragmentation**. And only the first fragmentation has the UDP header.

UDP header: Source port 53150.
Destination port 5201.
Length 10000
Checksum

```
19:58:17.868285 f8:0f:41:c4:7f:aa > f8:0f:41:c4:7f:a8, ethertype IPv4 (0x0800), length 46: (tos
0x0, ttl 64, id 56833, offset 0, flags [DF], proto UDP (17), length 32)
    128.238.66.104.53150 > 128.238.66.105.5201: [bad udp cksum 0x86cb -> 0x2464!] UDP,
length 4
    0x0000:  4500 0020 de01 4000 4011 d61d 80ee 4268
    0x0010:  80ee 4269 cf9e 1451 000c 86cb 15cd 5b07
19:58:17.906588 f8:0f:41:c4:7f:aa > f8:0f:41:c4:7f:a8, ethertype IPv4 (0x0800), length 1514: (tos
0x0, ttl 64, id 56834, offset 0, flags [+], proto UDP (17), length 1500)
    128.238.66.104.53150 > 128.238.66.105.5201: UDP, length 10000
    0x0000:  4500 05dc de02 2000 4011 f060 80ee 4268
```

For IP headers of 7 fragmentation, they share the exactly same identification, different fragment offsets and flags bits.

1472+5*1480+1128=10000

id 56834, offset 0, flags [+], proto UDP (17), length 1500)
id 56834, offset 1480, flags [+], proto UDP (17), length 1500)
id 56834, offset 2960, flags [+], proto UDP (17), length 1500)
id 56834, offset 4440, flags [+], proto UDP (17), length 1500)
id 56834, offset 5920, flags [+], proto UDP (17), length 1500)
id 56834, offset 7400, flags [+], proto UDP (17), length 1500)
id 56834, offset 8880, flags [+], proto UDP (17), length 1148)

Exercise 5

What is the maximum size of user data in a UDP datagram that the system can send or receive, even when fragmentation is allowed? Why?

Since UDP header has a **16bits** length field (**0xFFFF = 65535**),the field size sets a maximum limit of **65,535 bytes (8 byte header + 65,527 bytes of data)** for a UDP datagram. Considering 20 bytes header of IP, there are only **65,507 bytes (65,535 − 8 byte UDP header − 20 byte IP header)**

Exercise 7

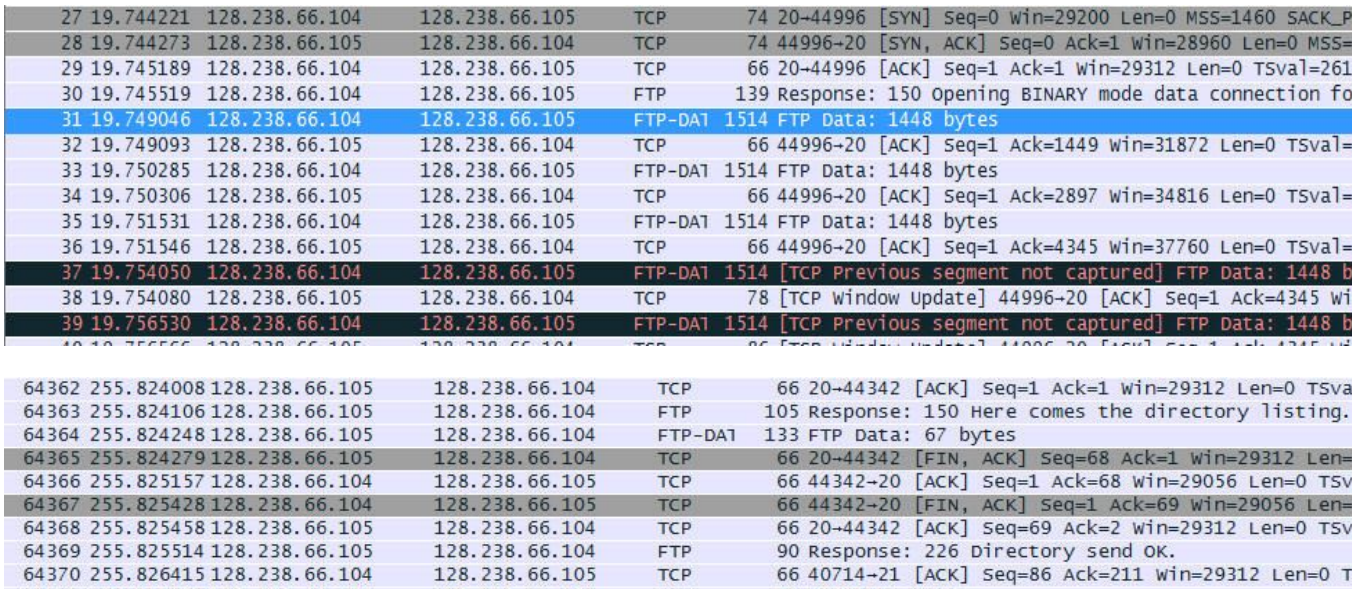
For FTP

Calculate the time spent for actual data transfer.

FTP data transfer start at 19.749, finish at 255.824. So the actual time spent on data transmission is **236.07 s** (255.824-19.749).

The value displayed in FTP window is **296.41 s**.

The reason why there exists time difference is that client and server need to establish TCP connection before actual data transferring, acknowledge during the data transfer process and quit after transfer completion. The time displayed in FTP window also include all of this procedure. So it's larger than time actual spent on data transfer.



For TFTP

Calculate the time spent for actual data transfer.

TFTP data transfer start at 0.000, finish at 1263.930. So the actual time spent on data transmission is **1263.93 s** (1263.930-0).

The value displayed in TFTP window is **1263.9 s.**

There are no significant difference between these two time. Because there are no TCP control and acknowledge during TFTP transfer.

	Time	Source	Destination	Protocol	Length	Info
1	0.000000	128.238.66.105	128.238.66.104	TFTP	63	Read Request, File: large.dum,
2	0.008332	128.238.66.104	128.238.66.105	TFTP	558	Data Packet, Block: 1
3	0.008377	128.238.66.105	128.238.66.104	TFTP	46	Acknowledgement, Block: 1
4	0.010415	128.238.66.104	128.238.66.105	TFTP	558	Data Packet, Block: 2
5	0.010450	128.238.66.105	128.238.66.104	TFTP	46	Acknowledgement, Block: 2
188160	1263.92583	128.238.66.104	128.238.66.105	TFTP	558	Data Packet, Block: 26129
188161	1263.92596	128.238.66.105	128.238.66.104	TFTP	46	Acknowledgement, Block: 26129
188162	1263.92826	128.238.66.104	128.238.66.105	TFTP	558	Data Packet, Block: 26130
188163	1263.92839	128.238.66.105	128.238.66.104	TFTP	46	Acknowledgement, Block: 26130
188164	1263.93006	128.238.66.104	128.238.66.105	TFTP	338	Data Packet, Block: 26131 (last)
188165	1263.93016	128.238.66.105	128.238.66.104	TFTP	46	Acknowledgement, Block: 26131
188166	1267.22298	WistronI_c4:7f:a8	WistronI_c4:7f:aa	ARP	42	Who has 128.238.66.104? Tell 128.238.66.
188167	1267.22393	WistronI_c4:7f:aa	WistronI_c4:7f:a8	ARP	60	128.238.66.104 is at f8:0f:41:c4:7f:aa

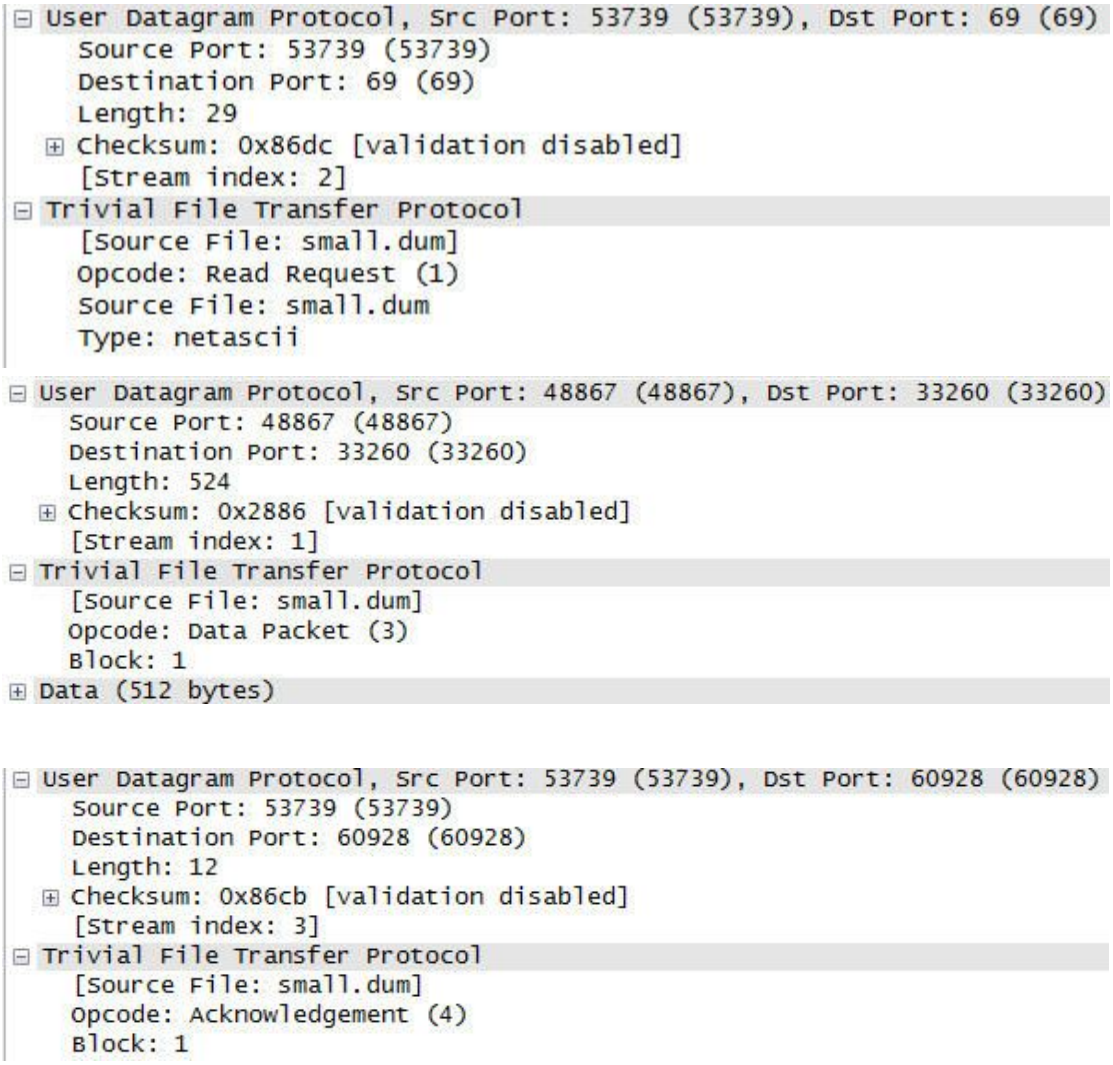
By comparing the actual transfer data time for FTP and TFPT, we find FTP is much more fast than TFTP, especially for large file, such as our large.dum file with 46MB.

The one reason is FTP allow transfer 1448 byte data for once but TFTP only has 512 byte capacity because of different protocols they based on. Another reason is FTP can use ACK cumulative acknowledgment to send multiple packets at once to improve ACK efficiency.

Exercise 8

List all the different types of packets exchanged during the TFTP session.

There are three types of packets, read request (opcode=0x1); data packet (opcode=0x3) and acknowledgment (opcode=0x4).



Why does the server port number change?

Port 69 is used for TFTP request control. That means any read request should be sent to port 69 but the other ordinary port is used for data transfer. This ensure server won't miss another request when TFTP data transfer is processing.

Why is tftp service not generally available to user?

There is no any log-in procedure during TFTP connection. User can connect to port 69 and get data transfer with TFTP. It's not a safe design.

With tftp, which uses UDP, we transferred a file larger than the maximum UDP datagram size. How do you explain this?

For TFTP protocol, it will transfer one whole file by dividing it into many UDP datagram, no more than 512 byte.

The maximum UDP datagram size, we find in Ex5, is a size limitation in network layer.

Exercise 9

How many well-known port numbers were used? Which machine used the well-known port numbers? What were the other machine's port numbers?

Server uses two well-known port, 20 for data transfer, 21 TCP control. Host use two port, 40735 and 36219.

```
Internet Protocol Version 4, Src: 128.238.66.105 (128.238.66.105), Dst: 128.238.66.104 (128.238.66.1
Transmission Control Protocol, Src Port: 21 (21), Dst Port: 40735 (40735), Seq: 1, Ack: 9, Len: 31
  Source Port: 21 (21)
  Destination Port: 40735 (40735)
Internet Protocol Version 4, Src: 128.238.66.105 (128.238.66.105), Dst: 128.238.66.104 (12
Transmission Control Protocol, Src Port: 20 (20), Dst Port: 36219 (36219), Seq: 1, Ack: 1,
  Source Port: 20 (20)
  Destination Port: 36219 (36219)
```

As can be seen from the tcpdump output, FTP involves two different connections, Why are two different connections used, instead of one connection?

For server, it is easier to handle other request from clients when data transfer is processing. And for client, it can check TCP connection status without interrupting data transfer.

Exercise 10

Submit what you saved in this exercise, explaining each line of the output.

Explain how the PORT command works.

Firstly, client selects a port number and send it to server with PORT command by control connection. Then server issues acknowledgment to that port number from client and allows data transfer by data connection.

Which connection, the control connection or the data connection, did the server send the response (the LIST output) on?

The server sends response on control connection.