CSCI4211: Introduction to Computer Networks

Homework Assignment III

Due 11:55PM Nov 18th, 2015

Help-hot-line: csci4211-help@cs.umn.edu

Name: Yanbang Liu   Student ID: 4446044

Important Notes:

Please submit your solutions as a single archive file (.zip or .tar or .tar.gz) on moodle. You may download the MS Word version of this assignment from moodle and edit it directly. Only online submissions are accepted for this assignment - do not attempt to submit hard copies. All textbook references pertain to the 6th edition.

You may discuss ideas and ask for clarifications freely with others on or off the class forum, and with Professor He or with the TAs. You must not provide or accept other assistance on the assignments. Feel free to post any queries you might have on the moodle discussion forum for this assignment.

Please do not write any thing other than name and student ID on this cover page

For TA only:

Problem	Points	Score
1	10	
2	10	
3	10	
4	10	
5	10	
6	50	
Total	**100**	

1. IP Address (10 pts)

1. Convert the IP address whose hexadecimal representation is CB44AFA2 to dotted decimal notation. (**2 pts**)
   0xCB = 0b203
   0x44 = 0b68
   0xAF = 0b175
   0xA2 = 0b162
   Thus: 203.68.175.162

2. What is the 32-bit binary equivalent of the IP address 100.114.20.107? (**2pts**)
   01100100 01110010 00010100 01101011

3. A network on the Internet has a subnet mask of 255.255.192.0. What is the maximum number of hosts it can handle (note: network address and local broadcast address are not assigned to individual hosts)? (**2 pts**)
   255.255.192.0 => 11111111 11111111 11000000 00000000
   This is equivalent to 255.255.192.0/18, and since there are 14 digits there can be 2^14 = 16,384 hosts.

4. Suppose an organization owns the block of addresses of the form 129.17.129.96/28. Suppose it wants to create four IP subnets from this block, with each block having the same number of IP addresses. What are the prefixes (of form xxx.xxx.xxx/y) for the four IP subnets? (**4pts**)
   129.17.129.96/28 => 10000001 00010001 10000001 0110 | 0000
   The last four digits range from 0000 to 1111 thus in total 16 addresses. Therefore each subnet would have 16/4 = 4 addresses, ends up in just two digits of binary number. Thus, the prefix is: 129.17.129.96/30

2. IP Datagram Forwarding (10 pts)

Considering a datagram network using 8-bit host addressing (totally 254 hosts), suppose a router use longest prefix matching and has following forwarding table. How many host addresses will the router route through interface 0, interface 1 respectively (5pt each)? (hint: when host id is all 0, the address is network address and when host id is all 1's, the address is local broadcast address. Both are not host addresses)

Prefix (binary)	*Interface*
1	*0*
101	*1*
101110	*2*
Otherwise	*3*

Consider Interface 0 (I0) first, every address that does not begin with 1 would be routed through Interface 3 (I3), since prefixes for I1 and I2 begin with 1 as well. Thus, addresses ranges from 1000 0000 to 1111 1110 would be routed through I0, I1, or I2. The rest will be routed through I3. Thus
N_I0+N_I1+N_I2 = 0d(0b11111110-0b10000000)+1 = 127, and
N_I3 = 0d(0b01111111-0b00000001)+1 = 127.
Then, since longest prefix matching, every address that does not have prefix 101110 will be matched to either I0 or I1, and every address that does will be matched to I2.
Therefore
N_I2 = 0d(0b10111011-0b10111000)+1 = 4, so
N_I0+N_I1 = 127-N_I2 = 123.
Then, everything that has prefix 101 will be matched to I1 or I2, and I0 otherwise, thus
N_I1+N_I2 = 0d(0b10111111-0b10100000)+1 = 32, so
N_I0 = 127-(N_I1+N_I2) = 95, and
N_I1 = 32-N_I2 = 28.

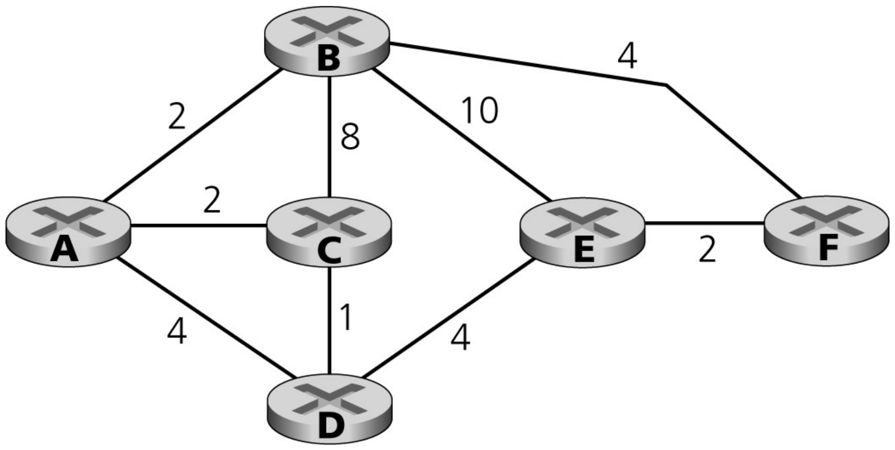
Thus

Interface	**Number of host addresses routed**
0	95
1	28
2	4
3	127

3. Link State Routing (10 pts)

Consider the network shown below. Show the operation of Dijkstra's (link-state) algorithm for computing the least cost path from A to all destinations using the table below (**8 pts**).
What is the shortest path from A to F, and what is the cost of this path (**2 pts**)?

Step	N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2, A	2, A	4, A	Infinity	Infinity
1	AB		2, A	4, A	12, B	6, B
2	ABC			4, A	12, B	6, B
3	ABCD				7, D	6, B
4	ABCDF				7, D	
5	ABCDFE					



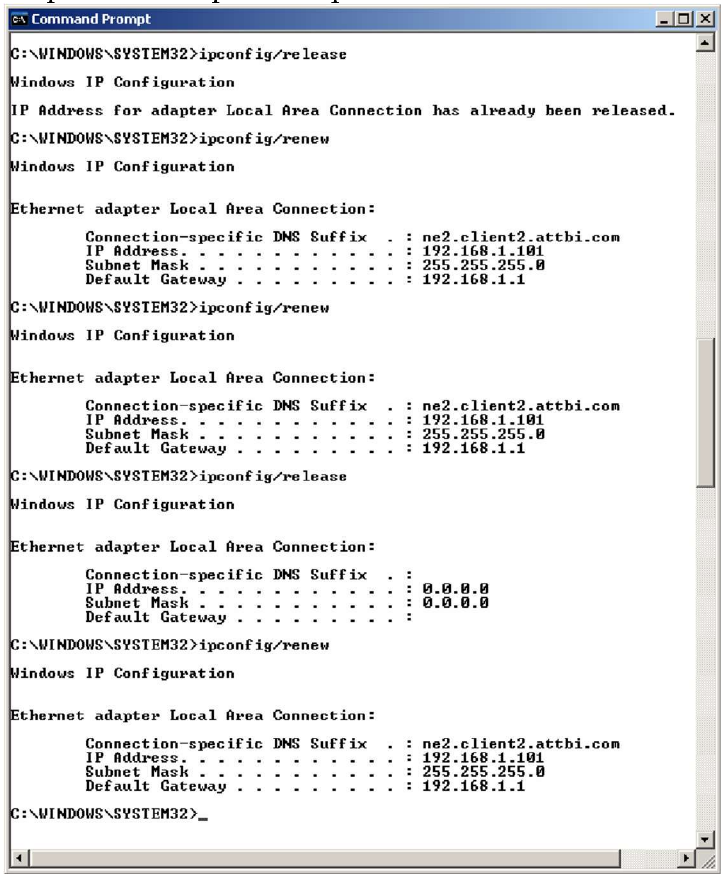
To calculate shortest path from A to F:
Access the last row in F column: 6, B => Shortest path cost A-F is 6, parent is B, then
Access the last row in B column: 2, A => Shortest path cost A-B is 2, parent is A, done.
Thus, the shortest path from A to F is ABF, and the cost is 6.

4. Hand-on Practice: DHCP (10 pts)

In this practice, we'll take a quick look at DHCP. In order to observe DHCP in action, we'll perform several DHCP-related commands and capture the DHCP messages exchanged as a result of executing these commands. Do the following as in Figure:

1. Enter "*ipconfig /release*" to releases your current IP address, so that your host's IP address becomes 0.0.0.0. (sudo dhclient –r is used in linux)

2. Start up the Wireshark packet sniffer, with *"bootp"* as the filter (Note to see DHCP packets in the current version of Wireshark, you need to enter *"bootp"* and not "dhcp" in the filter.)
3. Enter "*ipconfig /renew*". This instructs your host to obtain a network configuration, including a new IP address.
1. Stop Wireshark packet capture.



```
C:\WINDOWS\SYSTEM32>ipconfig/release

Windows IP Configuration

IP Address for adapter Local Area Connection has already been released.

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>ipconfig/release

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 0.0.0.0
        Subnet Mask . . . . . . . . . . . : 0.0.0.0
        Default Gateway . . . . . . . . . :

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>_
```

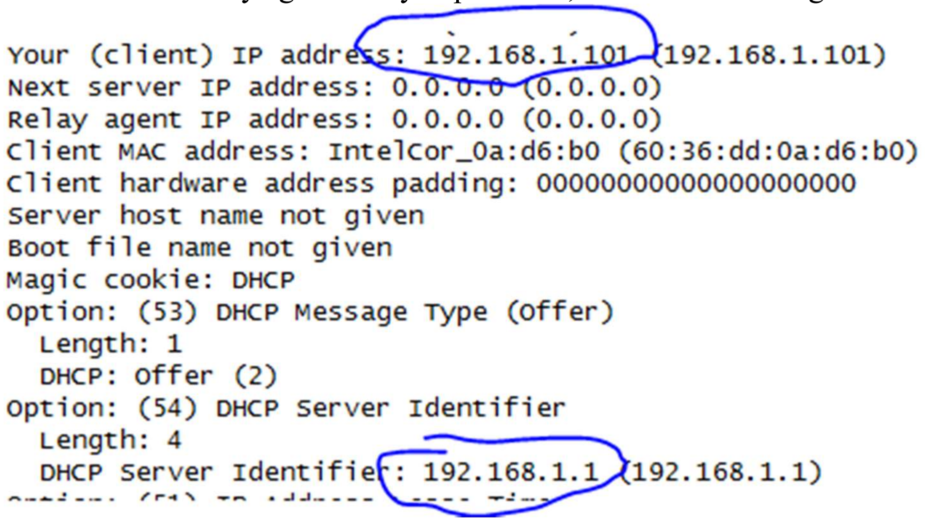
Provide two screen shots: command line screen similar to the figure above and a wireshark screen that captures the DHCP interaction (you can have similar screenshots in unix)

Based on the screenshots, answer the following questions:

1. Are DHCP messages sent over UDP or TCP? (**2pts**)
   UDP, as in the IPv4 header protocol field, it says UDP.
2. Explain the purpose of the router and subnet mask lines in the DHCP offer message. (**2pts**)
   Subnet mask masks the prefix of my IP address and the 0's correspond to my IP address field for that position. It also serves a purpose of telling how many subnet hosts are available. In my case I got 255.255.255.0, thus it means that I can have 128 different subnet host addresses.

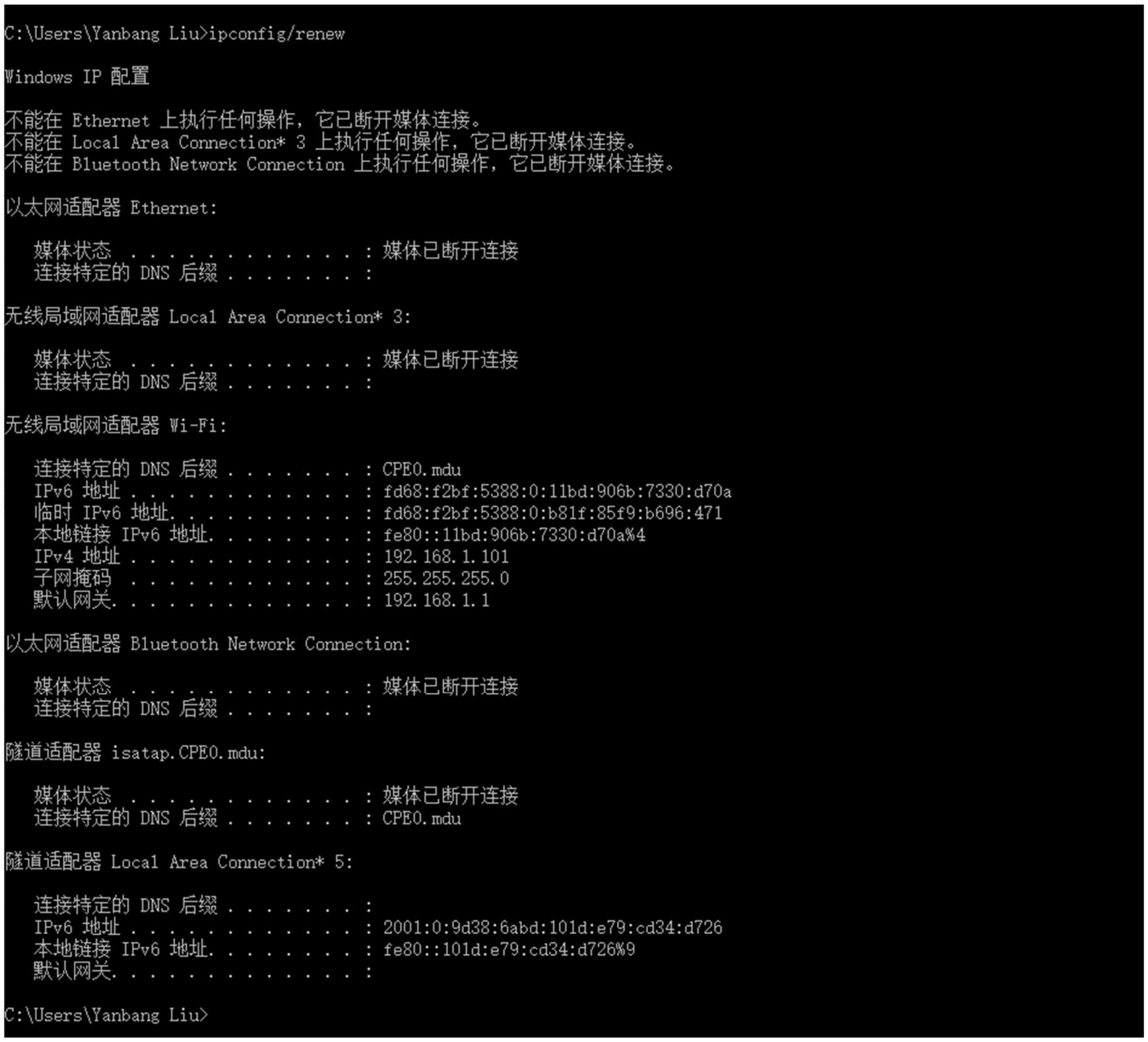
The router line just tells which router interface is this host (which is my computer) is connected to, and the router line is the address of that router interface.

3. Explain the purpose of the lease time. How long is the lease time in your experiment? (**2pts**)
   The lease time indicates how long I am allowed to be connected to the network for this session. In my case, my lease time is 1 day.
4. Explain the purpose of the DHCP release message? What would happen if the client's DHCP release message is lost? (**2pts**)
   The release message releases the client's assigned address from the DHCP server. If the release message is lost then client may have disconnected from the network, but the DHCP would still retain that address for that client, and not assign it to other client until the lease ends.
5. In a certain network configuration, the DHCP server might not be located at the same network as your machine. In this case, DHCP request are relayed by a relay agent. Is there a relay agent in your experiment? Justify your answer. (**2pts**)
   No there is no relay agent in my experiment, as shown in the figure:

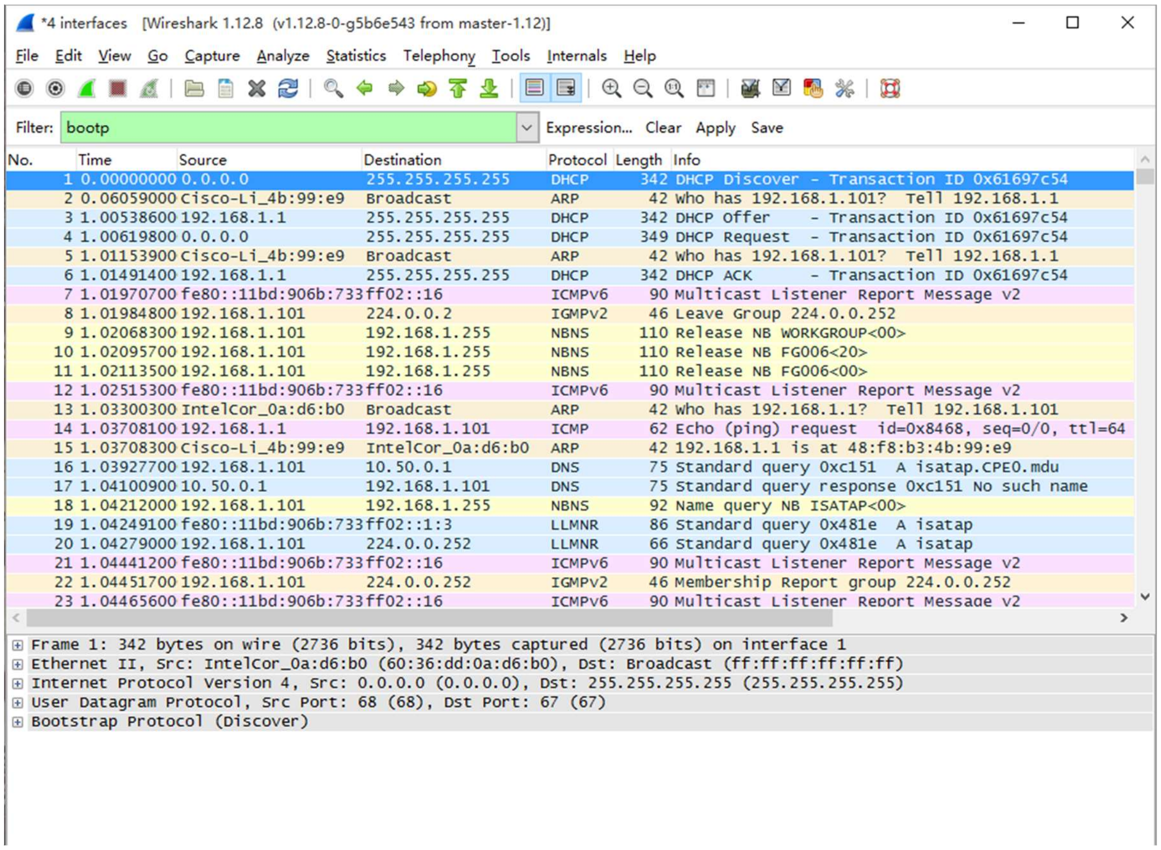


```
Your (client) IP address: 192.168.1.101 (192.168.1.101)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: IntelCor_0a:d6:b0 (60:36:dd:0a:d6:b0)
Client hardware address padding: 00000000000000000000
Server host name not given
Boot file name not given
Magic cookie: DHCP
Option: (53) DHCP Message Type (Offer)
  Length: 1
  DHCP: Offer (2)
Option: (54) DHCP Server Identifier
  Length: 4
  DHCP Server Identifier: 192.168.1.1 (192.168.1.1)
```

And consider the subnet mask is 255.255.255.0, the DHCP server and my computer are in the same subnet.



```
C:\Users\Yanbang Liu>ipconfig/renew

Windows IP 配置

不能在 Ethernet 上执行任何操作，它已断开媒体连接。
不能在 Local Area Connection* 3 上执行任何操作，它已断开媒体连接。
不能在 Bluetooth Network Connection 上执行任何操作，它已断开媒体连接。

以太网适配器 Ethernet:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

无线局域网适配器 Local Area Connection* 3:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

无线局域网适配器 Wi-Fi:

   连接特定的 DNS 后缀 . . . . . . . : CPE0.mdu
   IPv6 地址 . . . . . . . . . . . . : fd68:f2bf:5388:0:11bd:906b:7330:d70a
   临时 IPv6 地址. . . . . . . . . . : fd68:f2bf:5388:0:b81f:85f9:b696:471
   本地链接 IPv6 地址. . . . . . . . : fe80::11bd:906b:7330:d70a%4
   IPv4 地址 . . . . . . . . . . . . : 192.168.1.101
   子网掩码  . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . : 192.168.1.1

以太网适配器 Bluetooth Network Connection:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

隧道适配器 isatap.CPE0.mdu:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . : CPE0.mdu

隧道适配器 Local Area Connection* 5:

   连接特定的 DNS 后缀 . . . . . . . :
   IPv6 地址 . . . . . . . . . . . . : 2001:0:9d38:6abd:101d:e79:cd34:d726
   本地链接 IPv6 地址. . . . . . . . : fe80::101d:e79:cd34:d726%9
   默认网关. . . . . . . . . . . . . :

C:\Users\Yanbang Liu>
```


*4 interfaces [Wireshark 1.12.8 (v1.12.8-0-g5b6e543 from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: bootp
Expression... Clear Apply Save

```
No.  Time       Source                Destination       Protocol Length Info
   1 0.00000000 0.0.0.0               255.255.255.255   DHCP      342 DHCP Discover - Transaction ID 0x61697c54
   2 0.06059000 Cisco-Li_4b:99:e9     Broadcast         ARP        42 Who has 192.168.1.101?  Tell 192.168.1.1
   3 1.00538600 192.168.1.1           255.255.255.255   DHCP      342 DHCP Offer    - Transaction ID 0x61697c54
   4 1.00619800 0.0.0.0               255.255.255.255   DHCP      349 DHCP Request  - Transaction ID 0x61697c54
   5 1.01153900 Cisco-Li_4b:99:e9     Broadcast         ARP        42 Who has 192.168.1.101?  Tell 192.168.1.1
   6 1.01491400 192.168.1.1           255.255.255.255   DHCP      342 DHCP ACK      - Transaction ID 0x61697c54
   7 1.01970700 fe80::11bd:906b:733   ff02::16          ICMPv6     90 Multicast Listener Report Message v2
   8 1.01984800 192.168.1.101         224.0.0.2         IGMPv2     46 Leave Group 224.0.0.252
   9 1.02068300 192.168.1.101         192.168.1.255     NBNS      110 Release NB WORKGROUP<00>
  10 1.02095700 192.168.1.101         192.168.1.255     NBNS      110 Release NB FG006<20>
  11 1.02113500 192.168.1.101         192.168.1.255     NBNS      110 Release NB FG006<00>
  12 1.02515300 fe80::11bd:906b:733   ff02::16          ICMPv6     90 Multicast Listener Report Message v2
  13 1.03300300 IntelCor_0a:d6:b0     Broadcast         ARP        42 Who has 192.168.1.1?  Tell 192.168.1.101
  14 1.03708100 192.168.1.1           192.168.1.101     ICMP       62 Echo (ping) request  id=0x8468, seq=0/0, ttl=64
  15 1.03708300 Cisco-Li_4b:99:e9     IntelCor_0a:d6:b0 ARP        42 192.168.1.1 is at 48:f8:b3:4b:99:e9
  16 1.03927700 192.168.1.101         10.50.0.1         DNS        75 Standard query 0xc151  A isatap.CPE0.mdu
  17 1.04100900 10.50.0.1             192.168.1.101     DNS        75 Standard query response 0xc151 No such name
  18 1.04212000 192.168.1.101         192.168.1.255     NBNS       92 Name query NB ISATAP<00>
  19 1.04249100 fe80::11bd:906b:733   ff02::1:3         LLMNR      86 Standard query 0x481e  A isatap
  20 1.04279000 192.168.1.101         224.0.0.252       LLMNR      66 Standard query 0x481e  A isatap
  21 1.04441200 fe80::11bd:906b:733   ff02::16          ICMPv6     90 Multicast Listener Report Message v2
  22 1.04451700 192.168.1.101         224.0.0.252       IGMPv2     46 Membership Report group 224.0.0.252
  23 1.04465600 fe80::11bd:906b:733   ff02::16          ICMPv6     90 Multicast Listener Report Message v2
```

```
⊞ Frame 1: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 1
⊞ Ethernet II, Src: IntelCor_0a:d6:b0 (60:36:dd:0a:d6:b0), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
⊞ Internet Protocol Version 4, Src: 0.0.0.0 (0.0.0.0), Dst: 255.255.255.255 (255.255.255.255)
⊞ User Datagram Protocol, Src Port: 68 (68), Dst Port: 67 (67)
⊞ Bootstrap Protocol (Discover)
```

5. Hand-on Practice: ICMP (10 pts.)

In this practice, we capture the packets generated by the Traceroute program. You may recall that the Traceroute program can be used to figure out the path a packet takes from source to destination. Traceroute is discussed in Section 1.3 and in Section 4.4 of the text.

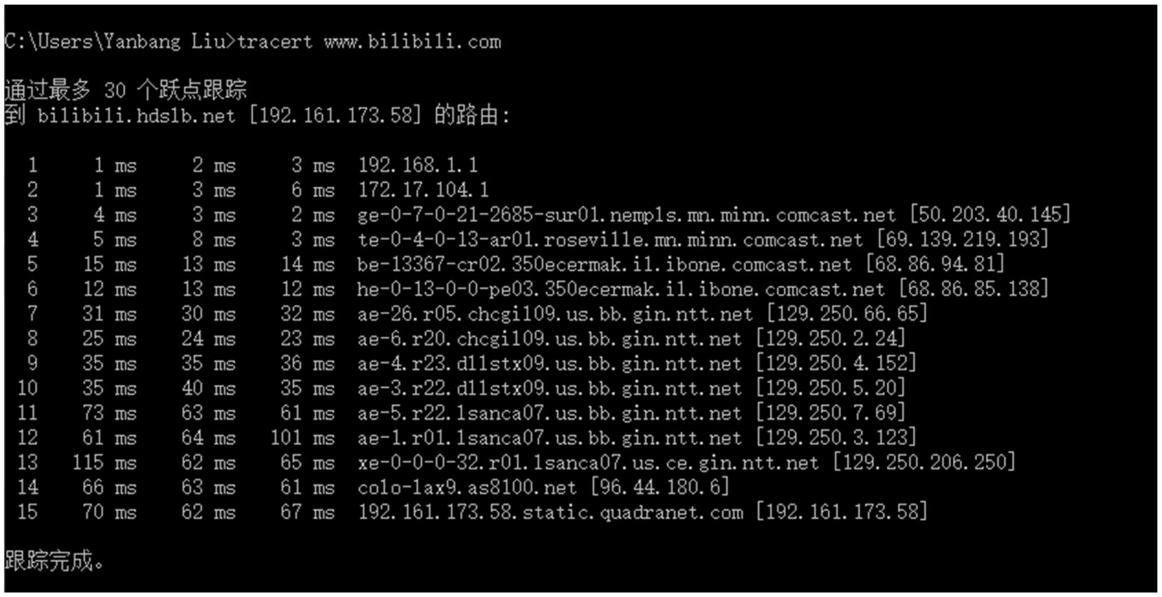
Traceroute is implemented in different ways in Unix/Linux and in Windows. In Unix/Linux, the source sends a series of UDP packets to the target destination using an unlikely destination port number; in Windows, the source sends a series of ICMP packets to the target destination. For both operating systems, the program sends the first packet with TTL=1, the second packet with TTL=2, and so on. Recall that a router will decrement a packet's TTL value as the packet passes through the router. When a packet arrives at a router with TTL=1, the router sends an ICMP error packet back to the source.

Do the following:

1. Start up the Wireshark packet sniffer, and begin Wireshark packet capture.
2. Type "tracert hostname". Choose a host outside of north America such as www.inria.fr , a computer science research institute in France.
3. When the Traceroute program terminates, stop packet capture in Wireshark.

You should hand in a screen shot of the output of tracert command, then answering the following question:

1. How many ICMP echo packets are sent? Justify the observed number from wireshark, based on the output of tracert command. (**4pts**)
   There are in total 15 echo packets sent.
2. Examine the last three ICMP error packets received by the source host. How are these packets different from the previous ICMP error packets? Why are they different? (**6pts**)
   In the last three error message, the ICMP echo messages inside each ICMP error message do not have the data field, and they do not have the ICMP Multi-Part Extensions as the previous ones either. The reason for this might be that the last three routers are connected with Ethernet with the destination host of traceroute. Since Ethernet has larger maximum data size, segmentation did not happen. Thus these three packets does not need to be reassembled in these routers.



```
C:\Users\Yanbang Liu>tracert www.bilibili.com

通过最多 30 个跃点跟踪
到 bilibili.hdslb.net [192.161.173.58] 的路由:

  1     1 ms     2 ms     3 ms  192.168.1.1
  2     1 ms     3 ms     6 ms  172.17.104.1
  3     4 ms     3 ms     2 ms  ge-0-7-0-21-2685-sur01.nempls.mn.minn.comcast.net [50.203.40.145]
  4     5 ms     8 ms     3 ms  te-0-4-0-13-ar01.roseville.mn.minn.comcast.net [69.139.219.193]
  5    15 ms    13 ms    14 ms  be-13367-cr02.350ecermak.il.ibone.comcast.net [68.86.94.81]
  6    12 ms    13 ms    12 ms  he-0-13-0-0-pe03.350ecermak.il.ibone.comcast.net [68.86.85.138]
  7    31 ms    30 ms    32 ms  ae-26.r05.chcgil09.us.bb.gin.ntt.net [129.250.66.65]
  8    25 ms    24 ms    23 ms  ae-6.r20.chcgil09.us.bb.gin.ntt.net [129.250.2.24]
  9    35 ms    35 ms    36 ms  ae-4.r23.dllstx09.us.bb.gin.ntt.net [129.250.4.152]
 10    35 ms    40 ms    35 ms  ae-3.r22.dllstx09.us.bb.gin.ntt.net [129.250.5.20]
 11    73 ms    63 ms    61 ms  ae-5.r22.lsanca07.us.bb.gin.ntt.net [129.250.7.69]
 12    61 ms    64 ms   101 ms  ae-1.r01.lsanca07.us.bb.gin.ntt.net [129.250.3.123]
 13   115 ms    62 ms    65 ms  xe-0-0-0-32.r01.lsanca07.us.ce.gin.ntt.net [129.250.206.250]
 14    66 ms    63 ms    61 ms  colo-lax9.as8100.net [96.44.180.6]
 15    70 ms    62 ms    67 ms  192.161.173.58.static.quadranet.com [192.161.173.58]

跟踪完成。
```

6. <u>Programming assignment: Host lookup and performance measurement (50 points)</u>

Implementation requirements and suggestions (must read)

1. This problem requires building a directory server, an app-server and an app-client. You can use the provided db-server and do not have to develop it.

2. You may implement the programs in the language of your choice. Provide a short description about compiling/running your program in the README file.

3. During development, it may be simpler to run all the programs on the same machine using the loopback interface. However, for actual measurement, please follow the instructions provided.

4. Please use port numbers between 9000 and 10000 when running your programs.

5. In order for these servers and clients to properly communicate with each other, it is important that all of them agree upon the message format. For this assignment, you can assume that:
   '\r' (Carriage return) indicates end of a line
   '\r\n' (Carriage return + newline) indicates end of a message
   When print messages on the screen, print each part of the message on a new line. See the example below for clarity.

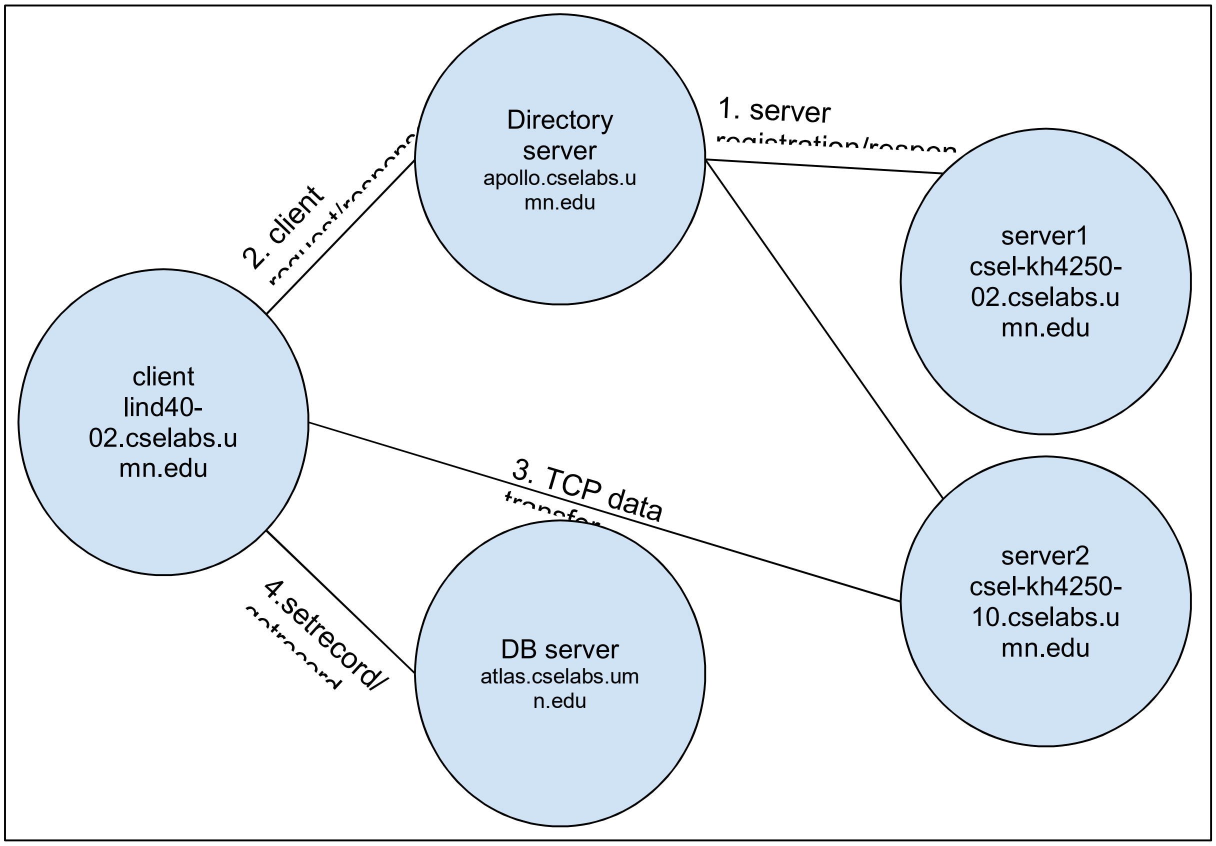
   For example:
   an app-server's registration message should be formatted like this:
   "register 128.101.37.1 9123\r\n"

   an app-client's list-servers message should be formatted like this:
   "list-servers\r\n"

   The directory server's response to the list-servers message will be like this:
   "success\r128.101.37.1 9123\r128.101.37.2 9321\r128.101.38.1 9321\r\n"

   Print the response on the console like this:
   ```
   success
   128.101.37.1 9123
   128.101.37.2 9321
   ```

6. All performance measurement must be performed on the client side.


Directory server
apollo.cselabs.umn.edu
1. server registration/respon
server1
csel-kh4250-02.cselabs.umn.edu
2. client request/respons
client
lind40-02.cselabs.umn.edu
3. TCP data transfer
server2
csel-kh4250-10.cselabs.umn.edu
4.setrecord/ getrecord
DB server
atlas.cselabs.umn.edu

For this assignment, you will build a dir-server, an app-server and an app-client. You will transfer data from the app-client to the app-server and measure the performance of the network path between them. You will then upload the performance results to the provided db-server.

Here is the sequence of operations that must be followed:

1. Start the dir-server on a specified port on **`apollo.cselabs.umn.edu`** and the db-server on a specified port on **`atlas.cselabs.umn.edu`**

2. Start two or more app-servers on different UNIX machines in Keller Hall. Each app-server must register itself with the directory server by providing its IP address and port number.

3. Start the app-client on one of the UNIX machines in Lind Hall. The app-client must query the directory server for the for the list of app servers.

4. The app-client must then connect to one of the app-servers and upload some data over a TCP connection and measure the time taken.

5. The app-client must then connect to the provided db-server and upload the results.

6. Finally, the app-client must download the results from the db-server and display them on the console.

Directory server

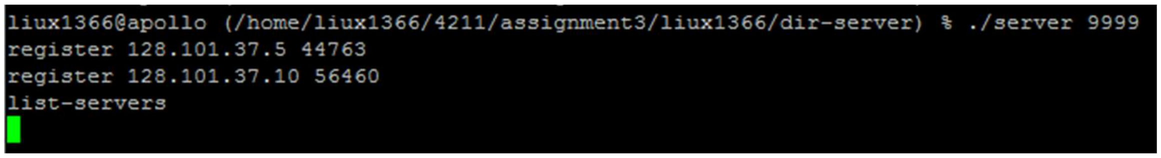
You must run the dir-server on `apollo.cselabs.umn.edu`.

1. The dir-server will be invoked as follows:
   `$ ./dir-server <ds_port>`
   The dir-server must start listening for TCP connections on the port specified by `ds_port`.

2. The app-server will try to register itself by sending the **register message**. The dir-server must respond and indicate success or failure.

3. The app-client will query the dir-server for a list of available servers by sending the **list-servers message**. The dir-server must respond and indicate success or failure. If successful, the list of servers needs to be returned to the client.

4. Print all received messages on the console and provide a screenshot of the same.



```
liux1366@apollo (/home/liux1366/4211/assignment3/liux1366/dir-server) % ./server 9999
register 128.101.37.5 44763
register 128.101.37.10 56460
list-servers
```

app-server

You must run at least two app-servers. Each app-server must run on one of the UNIX machines in Keller Hall.

The app-server will be invoked as follows:

`$ ./app-server <ds_port>`

The app-server must:

1. Create a TCP socket bound to a random port assigned by the operating system.
   (Hint: Set `sin_port = htons(0);` before calling `bind()`)

2. Print the following message on the console (without the quotes):
   "`<ip_address>, <port>`"
   `<ip_address>` is the address of the machine on which the app-server is running, and `<port>` is the port number which has been assigned to this socket by the operating system.
   (Hint: Use `getsockname()` or equivalent)

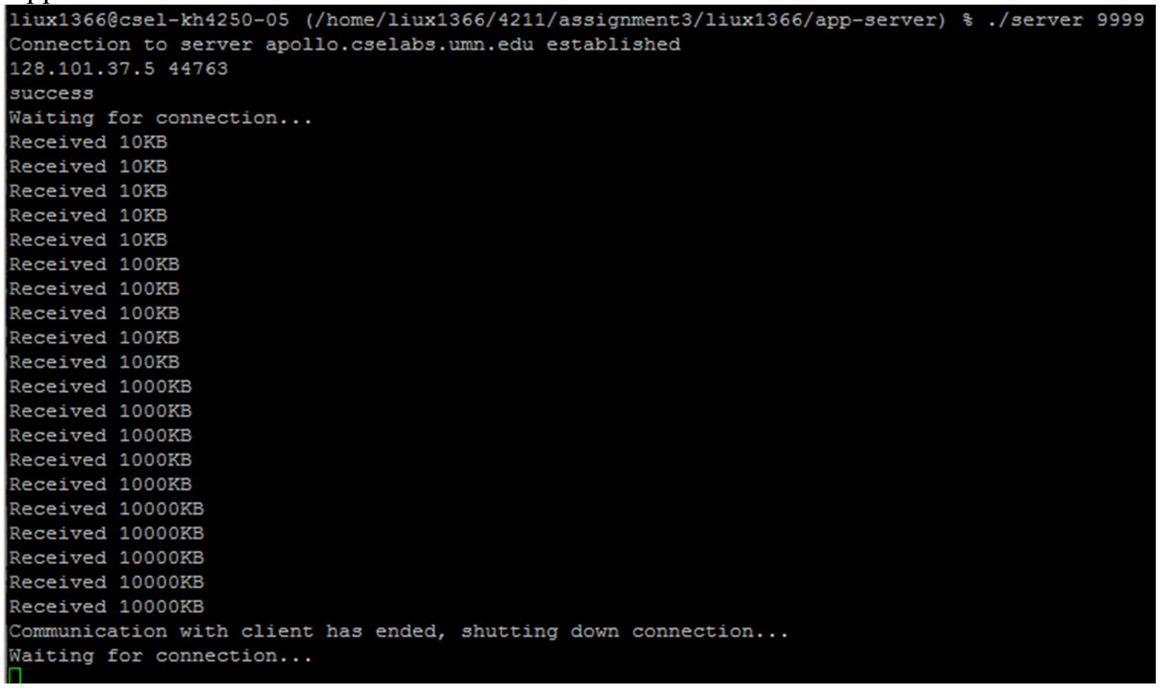
3. Establish a TCP connection with the dir-server, which should be running on `apollo.cselabs.umn.edu` on the port specified by `ds_port`.

4. Register with the dir-server, by sending the **register message**, using the IP address and port number determined in step 2. Print the response from the dir-server on the console.

5. Listen for app-clients on the socket created in step 1. An app-client will connect to this app-server and upload some data. The app-server must respond with an acknowledgement after all the data has been uploaded.

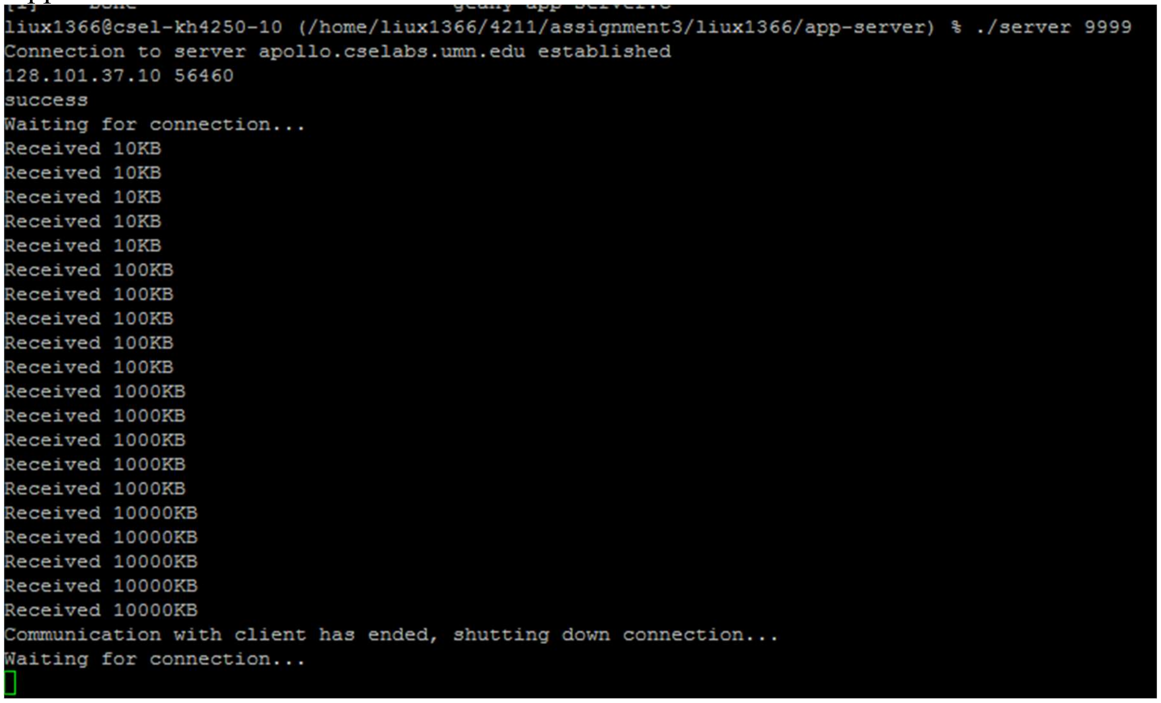
6. Print all received messages on the console and provide a screenshot of the same.

App-server 1



```
liux1366@csel-kh4250-05 (/home/liux1366/4211/assignment3/liux1366/app-server) % ./server 9999
Connection to server apollo.cselabs.umn.edu established
128.101.37.5 44763
success
Waiting for connection...
Received 10KB
Received 10KB
Received 10KB
Received 10KB
Received 10KB
Received 100KB
Received 100KB
Received 100KB
Received 100KB
Received 100KB
Received 1000KB
Received 1000KB
Received 1000KB
Received 1000KB
Received 1000KB
Received 10000KB
Received 10000KB
Received 10000KB
Received 10000KB
Received 10000KB
Communication with client has ended, shutting down connection...
Waiting for connection...
```

App-server 2



```
liux1366@csel-kh4250-10 (/home/liux1366/4211/assignment3/liux1366/app-server) % ./server 9999
Connection to server apollo.cselabs.umn.edu established
128.101.37.10 56460
success
Waiting for connection...
Received 10KB
Received 10KB
Received 10KB
Received 10KB
Received 10KB
Received 100KB
Received 100KB
Received 100KB
Received 100KB
Received 100KB
Received 1000KB
Received 1000KB
Received 1000KB
Received 1000KB
Received 1000KB
Received 10000KB
Received 10000KB
Received 10000KB
Received 10000KB
Received 10000KB
Communication with client has ended, shutting down connection...
Waiting for connection...
```

app-client

You must run the app-client on one of the machines in Lind Hall. The app-client will be invoked as follows:

`$ ./app-client <ds_port> <db_port>`

The app-client must:

1. Connect to the dir-server and obtain a list of available app-servers by sending the **list-servers message.** Print the response from the dir-server on the console.

2. Establish a TCP connection to one of the app-servers in the list by using the provided IP address and port number. Upload **`10KB`** of data to the app-server. The app-server will respond with an acknowledgement after all the data has been uploaded. Measure the time taken for transmission (Time from start of transmission till reception of application level acknowledgement; Do not include time for I/O). Repeat this operation at least five (5) times and compute the average time taken for upload.

3. Upload the performance measurements to the db-server by sending the **set-record message.** Print the response from the db-server on the console.

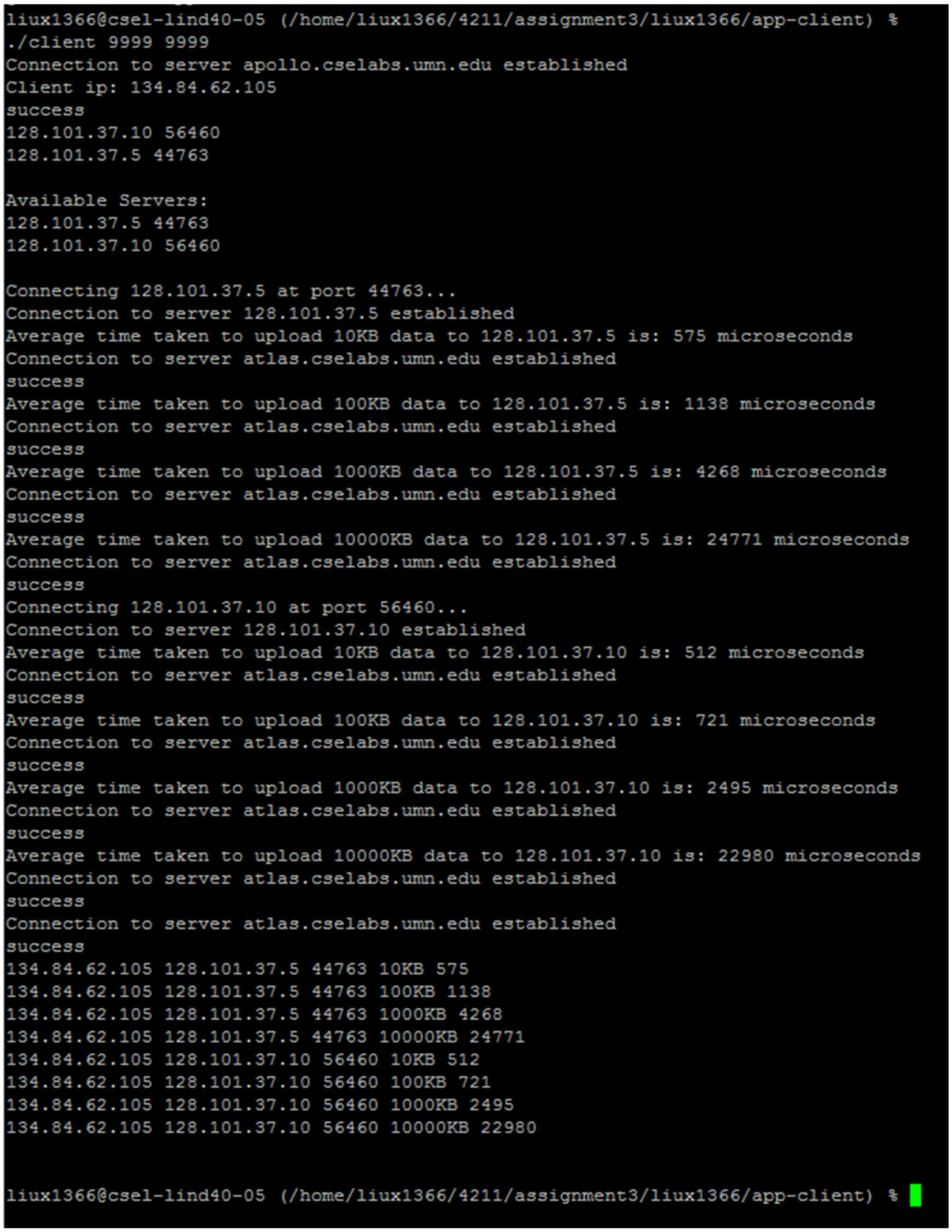
4. Repeat step 3 for different data sizes **`(10KB, 100KB, 1000KB and 10000KB)`**

5. Connect to the other app-server and repeat steps (2), (3) and (4).

6. Connect to the db-server which should be running on **`atlas.cselabs.umn.edu`** on the port specified by **`db_port`**.

7. Fetch records from the db-server by sending a **get-records message** to the db-server. Print the response from the db-server on the console.

8. Print all received messages on the console. Attach one or more screenshots showing the results of steps (1), (3) and (7).



```
liux1366@csel-lind40-05 (/home/liux1366/4211/assignment3/liux1366/app-client) %
./client 9999 9999
Connection to server apollo.cselabs.umn.edu established
Client ip: 134.84.62.105
success
128.101.37.10 56460
128.101.37.5 44763

Available Servers:
128.101.37.5 44763
128.101.37.10 56460

Connecting 128.101.37.5 at port 44763...
Connection to server 128.101.37.5 established
Average time taken to upload 10KB data to 128.101.37.5 is: 575 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Average time taken to upload 100KB data to 128.101.37.5 is: 1138 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Average time taken to upload 1000KB data to 128.101.37.5 is: 4268 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Average time taken to upload 10000KB data to 128.101.37.5 is: 24771 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Connecting 128.101.37.10 at port 56460...
Connection to server 128.101.37.10 established
Average time taken to upload 10KB data to 128.101.37.10 is: 512 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Average time taken to upload 100KB data to 128.101.37.10 is: 721 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Average time taken to upload 1000KB data to 128.101.37.10 is: 2495 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Average time taken to upload 10000KB data to 128.101.37.10 is: 22980 microseconds
Connection to server atlas.cselabs.umn.edu established
success
Connection to server atlas.cselabs.umn.edu established
success
134.84.62.105 128.101.37.5 44763 10KB 575
134.84.62.105 128.101.37.5 44763 100KB 1138
134.84.62.105 128.101.37.5 44763 1000KB 4268
134.84.62.105 128.101.37.5 44763 10000KB 24771
134.84.62.105 128.101.37.10 56460 10KB 512
134.84.62.105 128.101.37.10 56460 100KB 721
134.84.62.105 128.101.37.10 56460 1000KB 2495
134.84.62.105 128.101.37.10 56460 10000KB 22980


liux1366@csel-lind40-05 (/home/liux1366/4211/assignment3/liux1366/app-client) %
```

db-server
You must run the db-server on `atlas.cselabs.umn.edu`.
You may use the provided db-server - You **DO NOT** have to develop your own.
You can start the db-server by executing:

```
$ ./db-server <db_port>
```

1. An app-client can query the db-server for a list of available records by sending the **get-records message**.

2. An app-client can try to set a record in the db-server by sending the **set-record message.**

3. The db-server will store all the records in a plaintext file called **`client_records.dat`**. Delete this file if you want to clear the records in the db-server and start again.

Message type	register message
Description	This message is sent by the app-server to the dir-server to register itself. The dir-server must respond with a **“`success\r\n`”** or **“`failure\r\n`”** **`<ip_address>`** is the IP address of the app-server **`<port>`** is the port on which the app-server is listening
Sender	app-server
Receiver	dir-server
Message format	`“register <ip_address> <port>\r\n”`
Message response	`“success\r\n” or “failure\r\n”`
Example request	`“register 128.101.37.1 9123\r\n”`
Example response	`“success\r\n”`

Message type	list-servers message
Description	This message is sent by the app-client to the dir-server. The dir-server must respond with failure if no app-servers are available or indicate success and send a list of app-server IP addresses and port numbers back to the app-client.
Sender	app-client
Receiver	dir-server
Message format	`“list-servers\r\n”`
Message response	`“success\r<server1_IP> <server1_port>\r<server2_IP> <server2_port>\r\n” or “failure\r\n”`
Example request	`“list-servers\r\n”`
Example response	`“success\r128.101.37.1 9123\r128.101.38.1 9321\r\n”`

Message type	set-record
Description	This message is sent by the app-client to the db-server. The db-server will respond with failure if it is unable to save the record or will respond with success.
Sender	app-client
Receiver	db-server
Message format	`“set-record <client_IP> <server_IP> <port> <data_size> <upload_time>\r\n”`
Message response	`“success\r\n” or “failure\r\n”`
Example request	`“set-record 10.10.10.10 20.20.20.20 9123 10 0.05\r\n”`
Example response	`“success\r\n”`

Message type	get-records
Description	This message is sent by the app-client to the db-server. The db-server will respond with failure if it is unable to fetch the records or will respond with success and a list of available records.
Sender	app-client
Receiver	db-server
Message format	`“get-records\r\n”`
Message response	`“success\r<client_IP> <server_IP> <port> <data_len> <time>\r\n” or “failure\r\n”`
Example request	`“get-records\r\n”`
Example response	`“success\r10.10.10.10 20.20.20.20 9123 10 0.05\r10.10.10.10 20.20.20.20 9123 100 0.5\r\n”`

Deliverables:

1. You must upload a single archive file (.zip or .tar or .tar.gz) on moodle. When extracted, the archive file must be a single folder. The name of the folder should be your **student ID**. The folder should contain the following files:
   - Readme
   - app-server source files
   - app-client source files
   - MS Word/PDF document
2. DO NOT include the test files

For example, here is a sample submission:

```
1234567/
    Readme
    PA3.doc
    app-server/
        app-server.c
         Makefile
    app-client/
        app-client.c
         Makefile
    dir-server/
        dir-server.c
         Makefile
```

You can create an archive file from the contents of the above directory as follows:

```
$ tar cvf 1234567.tar.gz 1234567/
```

Grading:

README, Makefiles, comments, readability: **3 points**
Packaging the submission as specified: **2 points**
app-server registration & screenshots: **5 points**
app-client list-servers screenshot: **10 points**
app-client data upload and performance measurement: **10 points**
app-client set-records: **10 points**
app-client get-records screenshot: **10 points**