|  |
| --- |
| CSCI4211: Introduction to Computer Networks |

Homework Assignment III

**Due 11:55PM April 9th, 2014**

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

Name: \_\_\_Russell Bunch \_\_ Student ID: \_\_\_4352554\_\_\_

**Important Notes:**

Please submit your homework in either PDF or MS Word format using the online electronic submission system on the class moodle website. The MS Word version of this assignment can be downloaded at the moodle website, which you can edit directly. Only online submission accepted. No hard copy. All text book references pertain to the 6rd Edition. For details check the course website.

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.

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

|  |
| --- |
| For Teaching Assistant Only |
| Score of this homework: |

|  |  |  |
| --- | --- | --- |
| **Problem** | **Points** | **Score** |
| 1 | 20 |  |
| 2 | 10 |  |
| 3 | 10 |  |
| 4 | 10 |  |
| 5 | 10 |  |
| 6 | 10 |  |
| 7 | 30 |  |
| **Total** | **100** |  |

**1. IP Address (20 pts.)**

1. Convert the IP address whose hexadecimal representation is CA23AFA2 to dotted decimal notation. (2 pts.)

CA23AFA2 = 11001010 00100011 10101111 10100010 = 202.35.175.162

1. What is the 32-bit binary equivalent of the IP address 200.104.30.127? (2pts.)

200.104.30.127 = C8681E7F

1. A network on the Internet has a subnet mask of 255.255.224.0. What is the maximum number of hosts it can handle (note: network address and local broadcast address are not assigned to individual hosts)? (4 pts.)

This subnet has a first part of address of 19. This gives us 13 bits to use in the subnet, therefore 2^13-2 = 8190. Then removing the network address 255.255.224.0

1. Suppose an organization owns the block of addresses of the form 129.17.129.97/27. 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)

Since the block of IPs has a y of 27 and starts at a.b.c.96, we have a range from a.b.c.96 to a.b.c.127. Therefore we can support 32 hosts, which gives us 8 hosts per group. This gives us the prefixes of:

Subnet 1: 129.17.129.96/29

Subnet 2: 129.17.129.104/29

Subnet 3: 129.17.129.112/29

Subnet 4: 129.17.129.120/29

1. Consider a router which interconnects three subnets: Subnet1, 2 and 3. Suppose all of the interfaces in each of the three subnets are required to have the prefix 211.2.112/23. Also, the Subnet 1 is required to support 250 interfaces, and Subnets 2 and 3 are each required to support up to 126 interfaces. Provide three network addresses which satisfy these constraints. (6pts) and justify (2pts).

Subnet 1: 211.2.112.0/24

Subnet 2: 211.2.113.0/25

Subnet 3: 211.2.113.128/25

Subnet 1 gives us 2^8-2 hosts (254). This takes up a.b.112.0 through a.b.112.255. Subnet 2 starts at a.b.113.0 which follows subnet 1. Here we have 2&7-2 hosts (126). That takes up a.b.113.0 through a.b.113.127. Finally subnet 3 will be identical to subnet 2 however the range will instead be a.b.113.128 through a.b.113.255 (another 126 hosts).

**2. IP datagram (10 pts)**

Suppose a host has a file consisting of 2 million bytes. The host is going to send this file over a link with an MTU of 1,500 bytes. Assuming the data is encapsulated with TCP and IP headers. How many datagrams are required to send this file? (5pts) what is the size of the last datagrams (with TCP and IP header). (5 pts)

The IP and TCP header combined gives us 40 bytes. Therefore each datagram can only hold 1500 – 40 = 1460 bytes of data. This gives us datagrams with all but the last datagram having a size of 1500. The last one is has bytes, adding the headers gives us a final answer of 1300 for the last datagram.

**3. 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 |

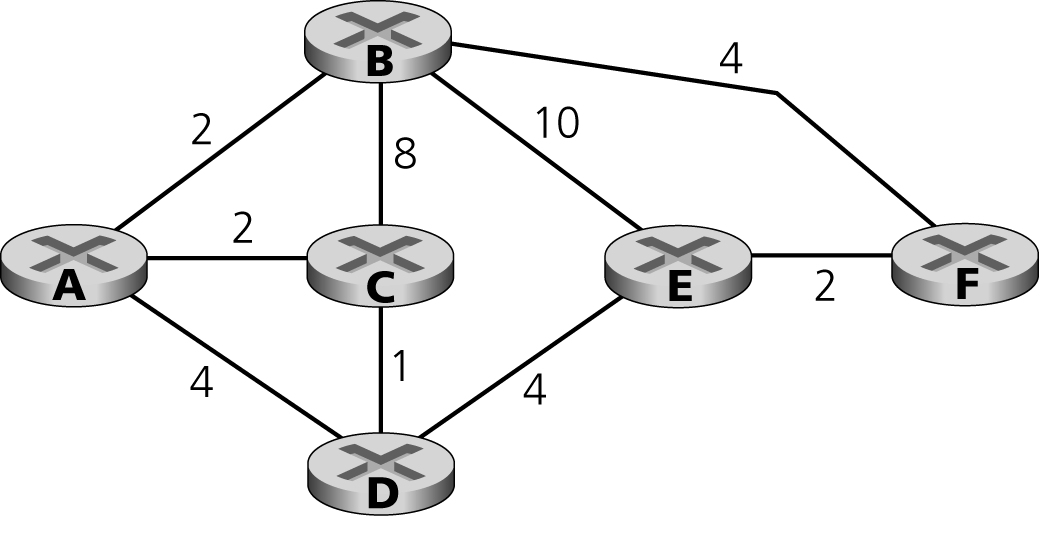
* Interface 0 has ranges 10000000-10011111 and 11000000-11111111 which gives us 32 + 63 (because 11111111 can not be used) hosts respectively to get 95 hosts
* Interface 1 has ranges 10100000-10110111 and 10111100-10111111 which give 24 + 4 hosts respectively to get 28 hosts
* Interface 2 has range 10111000-10111011 giving us a total of 4 hosts
* Finally the remaining 128 hosts from 00000000-01111111 is 127 (since 00000000 can not be a host)

**4. 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 | INF | INF |
| 1 | AB |  | 2,A | 4,A | 12,B | 6,B |
| 2 | ABC |  |  | 3,C | 12,B | 6,B |
| 3 | ABCD |  |  |  | 7,D | 6,B |
| 4 | ABCDE |  |  |  |  | 6,B |
| 5 | ABCDEF |  |  |  |  |  |

Shortest path from A to F is A-B-F which costs 2 + 4 = 6.



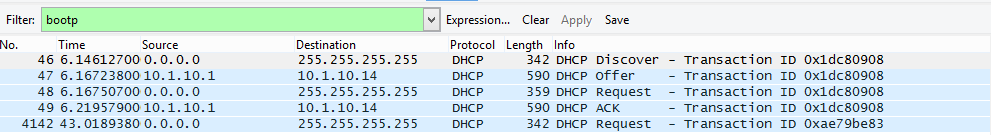
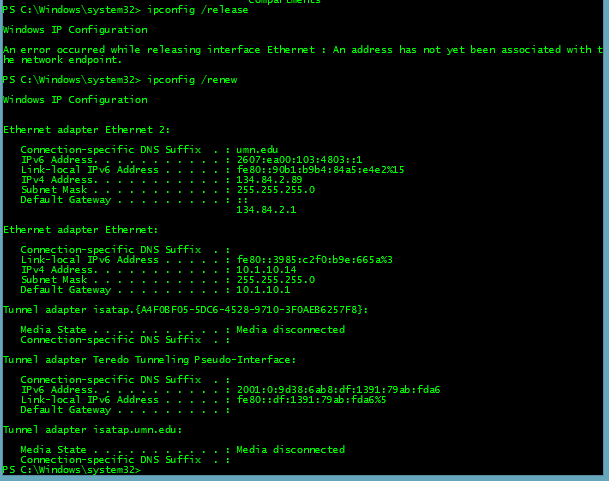
**5. Hand-on Practice: DHCP (10 pts.)**

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.
4. Stop Wireshark packet capture.



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)

DHCP messages are sent over UDP

1. Explain the purpose of the router and subnet mask lines in the DHCP offer message. (2pts)

The subnet mask identifies which subnet should be used, the router identifies which IP the router is behind so the requestor can find the DHCP server.

1. Explain the purpose of the lease time. How long is the lease time in your experiment? (2pts)

The lease time is how long the given IP is good for. Mine was 7 days (604800s).

1. Explain the purpose of the DHCP release message? What would happen if the client’s DHCP release message is lost? (2pts)

The release message removes the assigned IP from the specified client. This does not entirely matter if it is lost as the client simply won’t care what is sent to that address anymore on the router side.

1. 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)

There is no relay agent in my experiment, the relay agent IP address is 0.0.0.0.

**6. 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.

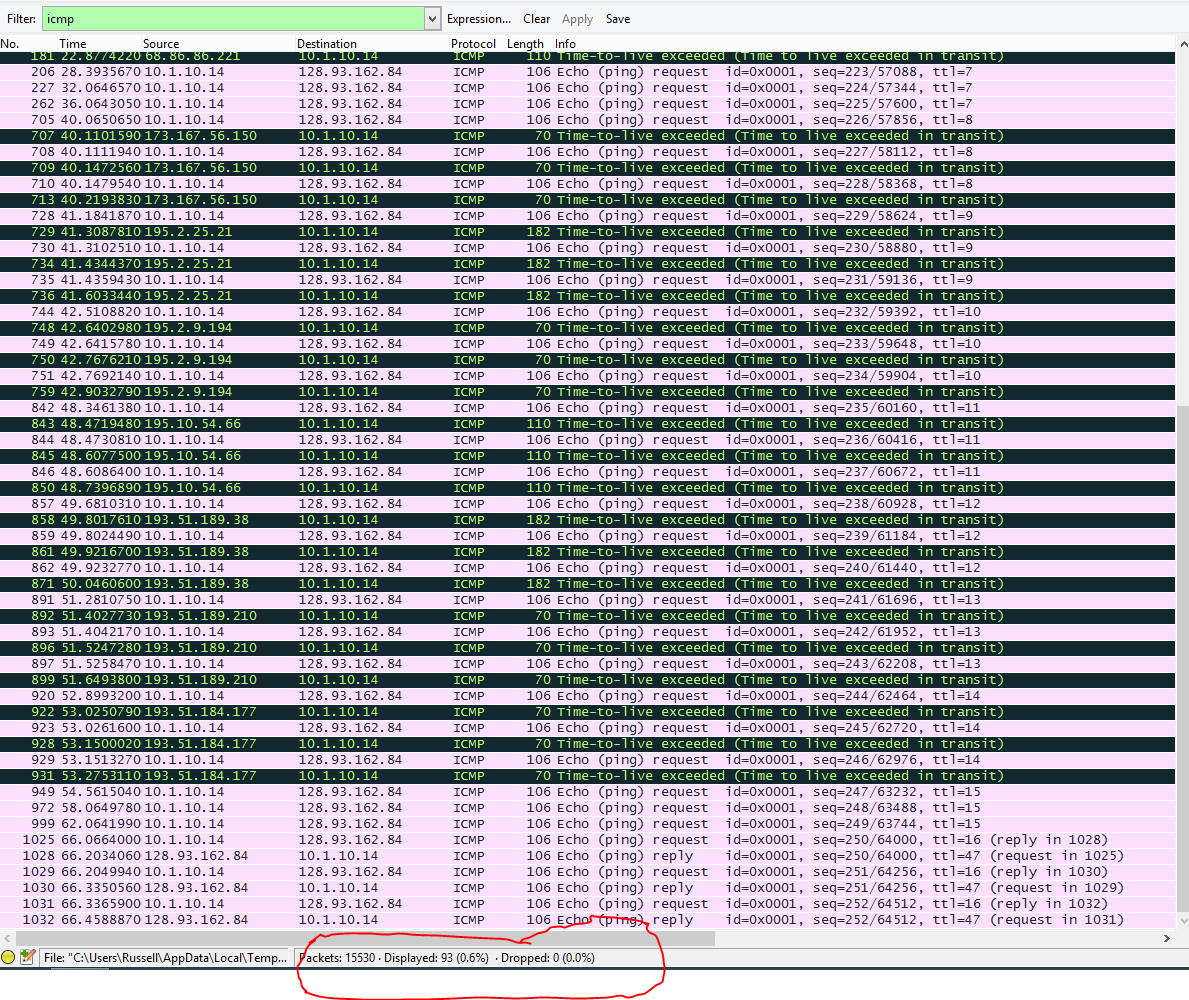
Do the following:

1. Start up the Wireshark packet sniffer, and begin Wireshark packet capture.
2. Type “tracert hostname”. Choose a host outside of north Americ such as [www.inria.fr](http://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 were 93 ICMP packets captured:



1. 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)

These replies are requests instead of time outs. This is a signal that we have completed our tracert, it is repeated 3 times since we test each hop three times.

**7.Programming: Host lookup and network performance measurement (30 pts)**

**Implementation requirements and suggestions (must read!)**

1. This problem requires building one server and one client application, which are extensions of the programs wrote for problem 7 in assignment 2. Please feel free to reuse those codes wherever possible.
2. You are free to use any language and your applications can have any function, as long as they can provide the information needed.
3. During implementation, it is recommended to run both the server and client on one machine using *loopback*, as it greatly helps the debugging process.
4. When it comes to actual measurement, server and client needs to be at least **one hop away** (i.e., running on different machines)
5. This problem requires access to the directory server and DB running on a CSELABS machine (apollo.cselabs.umn.edu). Therefore, **you need to run your client and server within the university or use VPN to connect to the department.**
6. **For your own server please use only the port number between 9000 and 10000. These ports are currently allowed by the system staff.**
7. In order for your client and server applications to properly communicate with given servers, it is important that the three applications agree upon the message format. Any message exchange with directory server must use **‘\r’ to indicate end of line**,while **‘\r\n’ is used to delimit the end of a single communication message**.

For example, a server registration message (which will be described in step a.) should be formatted like this:

*Register kimx2490 134.84.228.64 56789\r\n*

Directory server’s response stream to serverlist (which will be described in step b.) command would look like this:

*200 Success\rkimx2490 134.84.228.64 56789\rhexxx071 134.84.228.21 65000\r\n*

Which should be shown as below on your command prompt:

*200 Success*

*kimx2490 134.84.228.64 56789*

*hexxx071 134.84.228.21 65000*

1. This is an individual work. All the programming and measurement should be done independently.
2. Network performance measurement should be done on the client side.
3. Make sure your firewall is set accordingly.

**Problem overview**



Similarly to problem 7 in assignment 2, you will build one server and one client to measure the performance of the network path between them. However, this time the client looks for the server address from the directory server, instead of assuming the server address is known in advance. Also, measurement result has to be uploaded to DB. Directory server and DB are set up on a CSELABS machine, Apollo. Your task is to build one client and one server that can interact with given servers. Specifically, you need to do the followings, in the order described:

#1: Your server registers its IP address, port number, and X500 id of the author to the directory server. Registration should be the first thing done after server startup.

#2: Client looks up server list in directory server. Note that the directory server is shared among students, and the list shows all the servers currently available, including yours.

#3: Within the obtained server list, your client search by your X500 to find the address of your server. This information is used to establish a TCP connection for network performance measurements.

#4: Client uploads the network performance records onto DB.

For the grading purpose, **all the processes described above should be done automatically**. When grading, TA will run your server application first, followed by client application. **Your server and client should show all issued commands as well as received responses on the command prompt. (For screen capturing)**

Please note that your server and client should run on any machine; that is, **no hard coding of the client/server addresses is allowed.** (Please note thatIP and port numbers for directory server and DB should be hard-coded)

1. **Server registration (4 pts)**

As soon as the server starts, it shall establish a TCP connection to the directory server, which can be reached at *apollo.cselabs.umn.edu:9080*

When connected, the server shall issue a single communication message to the directory server. Watch out for blanks between fields and the delimiter at the end, which is not visible in command prompt:

* *Register <X500 id> <IP address> <port number>*

If the registration is successful, directory server will reply with “200 Success”.

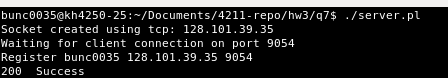
A**bove message needs to be sent periodically (at least once a minute) to directory server** in order to keep your server within server list. It is a process to inform the directory server that your server is still alive. Otherwise, directory server will assume your server is not in service and will eliminate it from the list.

|  |  |
| --- | --- |
| **DataType** | <X500 id> = String, <IP address> = String, <port number> = String |
| **Returns** | “200 Success” or “<failure code> <failure reason>” |
| **Example** | Command:  > *Register* kimx2490 134.84.228.64 9999 |
| Response:  >200 Success |

**Provide a screenshot of command prompt showing this process.** It can simply be command issued to directory server as well as response received from it, as shown below:

> *Register* kimx2490134.84.228.649999

> 200 Success



1. **Server address lookup (4 pts)**

Upon client startup, it establishes TCP connection to directory server. This connection is used to check for availability and address of your server. Specifically, it looks up current server list by sending the following communication message:

* *serverlist*

Directory server will respond with the addresses of all the servers available at the moment. Among them, client chooses the one with your X500 id to extract IP and port number.

|  |  |  |
| --- | --- | --- |
| **DataType** | None | |
| **Returns** | 200 Success” or “<failure code> <failure reason>”  If success, followed by a server list | |
| **Example** | Command:  >serverlist | Response:  >200 Success  >kimx2490 134.84.228.64 9999  >hexxx071 134.84.228.21 10000 |

**Provide a screenshot of this process.** It can simply be command issued to directory server as well as response received from it, as shown below:

> serverlist

> 200 Success

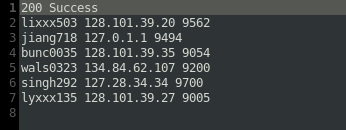
> kimx2490 134.84.228.64 9999

> hexxx071 134.84.228.21 10000

**NOTE: I had to print the records to a file because I was having difficulty with the carriage returns. Therefore there are two screen shots. However in order for my code to function it does not output the serverlist (it is still retrieved). Specific detail can be found in the readme.**

Client asking for servers:



Servers returned: 

1. **Measurement upload (6 pts)**

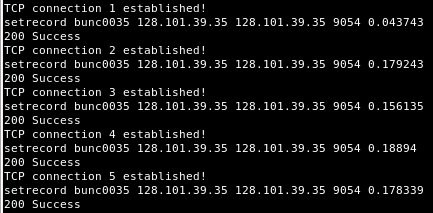
Make a TCP connection from your client to your server (not to somebody else’s server) using the address obtained from directory server. Repeat the measurement in prob. 7 (d) of assignment 2. That is, use your client to send 10,000,000 bytes of application data to server, where server responses with an application-level ACK upon receiving all data. From the client side, measure the time taken for the transmission. (i.e., the time from start of transmission until application-level ACK reception) Repeat this for at least five times.

Establish TCP connection from client to DB. Upload measurements onto DB by sending below message:

* *setrecord <X500 id> <client IP> <server IP > <server port> <measured time>*

Where measured time should be in milliseconds. Since you have at least five measurements, **upload at least five different records.**

**My program sends a 10,000,000 byte file called 10MBfile.dat every connection. Looking in the submitted q7 folder will yield this file for verification. The floating point number is the RTT (referring to the screenshot below).**

****

|  |  |
| --- | --- |
| **DataType** | <X500 id> = String, <client/server IP> = String,  <server port> = String, <measured time> = String |
| **Returns** | “200 Success” or “<failure code> <failure reason>” |
| **Example** | Command:  > kimx2490 134.84.228.63 134.84.228.64 56789 500 |
| Response:  >200 Success |

**Provide a screenshot(s) of this process.** It can simply be command issued to DB as well as response received from it, as shown below:

> kimx2490 134.84.228.63 134.84.228.64 56789 500

> 200 Success

> …

(Repeated at least 5 times)

**.**

1. **Measurement download (6 pts)**

After the upload in step c, download the statistics by sending the following communication message to DB via TCP connection:

* *getrecord*

This will return up to 50 most recent records. (Not necessarily yours)

|  |  |
| --- | --- |
| **DataType** | None |
| **Returns** | “200 Success” or “<failure code> <failure reason>” |
| **Example** | Command:  > getrecord |
| Response:  >200 Success  > kimx2490 134.84.228.63 134.84.228.64 9999 500  > hexxx071 134.84.228.21 134.84.228.24 10000 450 |

**Provide screenshot(s).** It can simply be command issued to DB as well as response received from it,but **the screenshot(s) should include at least five records of your own, as well as five of other students.**

> getrecord

> 200 Success

> kimx2490 134.84.228.63 134.84.228.64 9999 500

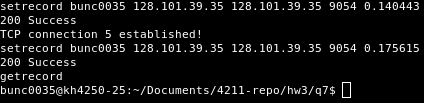
> hexxx071 134.84.228.21 134.84.228.24 10000 450

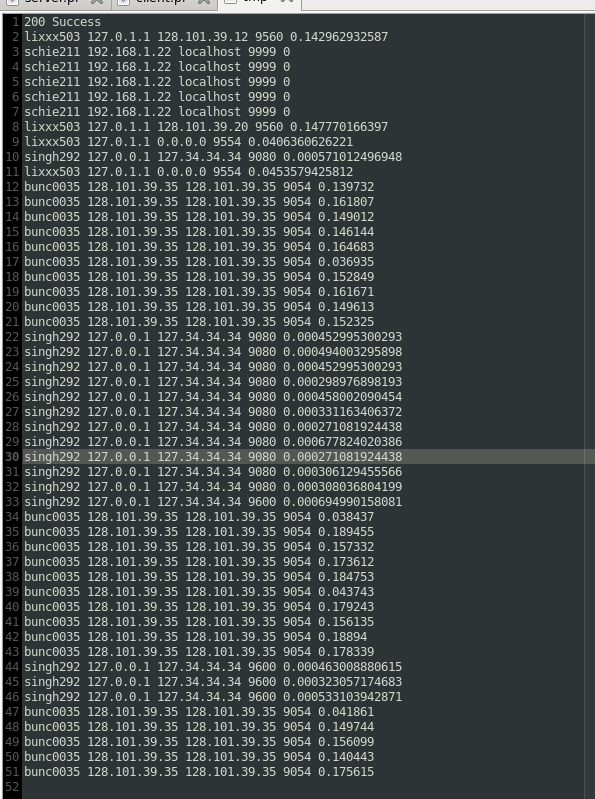
> …

(Total of 10 records)

**NOTE: I had to print the records to a file because I was having difficulty with the carriage returns. Therefore there are two screen shots.**

Client asking for records:



Records returned:

**Submission guidelines**

1. Zip and submit all source codes used in this problem along with your solution.
2. Please provide a brief readme file (need not be in detail) containing short descriptions on each source file, how your code works, and instructions on compiling and running.

**Grading guidelines**

1. 4 pts for screenshots in step a, b.
2. 6 pts for screenshots in step c, d.
3. 10 pts for the testable code and readme files

(Remember, your codes need to run **automatically** to print the results on command prompt)