Assignment 1

CSL 374/672

Due date: 4th August, 2013 (Sunday)

Note: *Solve all problems on your own.* Approach the instructor for clarifications.

1. The aim of this part is to make you familiar with basic network tools. Read the man pages of *ifconfig, ping, traceroute, arp, dig* and *nslookup*. Write the solution of the following problems into the report and also paste the screenshots of Terminal into the report.

   (a) Find the IP address of your machine using '*ifconfig*' command.

   (b) Use '*ping www.iitk.ac.in*' command and find out

      i. the average RTT(round trip time).

      ii. the %packet loss.

      iii. size of packet that is sent to www.iitk.ac.in server.

      iv. size of packet that is received by your machine.

   (c) Use '*dig www.iitk.ac.in*' command and find out

      i. the ip address of iitk.ac.in.

      ii. the ip addresses of local DNS servers of IITD.

   (d) Use '*traceroute www.iitk.ac.in*' and find out

      i. number of hops in between your machine and *iitk.ac.in* server.

      ii. the ip address of your network gateway of your subnet.

   (e) Use '*arp -an*' command to find out the MAC address of the device that is performing as your network gateway.

2. This part is about socket programming. You can learn socket programming basics and use python source code from python documentation to implement this assignment. Write two socket programs in python, client.py and server.py, that together communicate using Datagram Sockets. Each datagram generated by client.py must contain

   (a) a sequence number which identifies the packet,

   (b) a timestamp with microsecond level precision which indicates the time at which the packet is first transmitted,

   (c) an even non-negative integer called the time-to-live field (TTL) with initial value T. Let us assume that the packet is of size P bytes.

   When server.py receives a datagram from client.py, it immediately decrements the TTL value in the datagram and sends the same datagram (with the new TTL) back to client.py. The client.py program on receiving a datagram from server.py, decrements the TTL value, and checks if this new value is zero. If the new TTL is greater than zero, then client.py sends the datagram (with the new TTL) back to server.py.

   However, if TTL is zero, then client.py prints to a file (on a new line) the difference between the current time and the timestamp field in the datagram. Call this time

the cumulative RTT. A new datagram is then generated by client.py with TTL set to T. The value of P and T , and the output file name for storing the cumulative RTT should be entered on the command line when executing client.py. P should be within the range 100 to 1300 bytes, and T between 2 and 20 (and must be even). The client.py program totally sends out 50 datagrams and then quits. Run client.py and server.py on two different machines.

For T = 2 and different values of P = 100, 200, . . . , 1000, run client.py. Plot a scatter- plot (using any suitable software, such as matlab, gnuplot etc.) of cumulative RTT for all 50 datagrams vs. P for the different values of P when T = 2. What do you observe? What information does the slope of the graph contain? Repeat when T = 8 and 16.

Add the plots, and your observations for each plot, to the pdf report.

Download assignment1.zip and read README for further information.