CMPT 434
Computer Networks

Assignment One

Due: Wednesday February 3rd, 6:00pm – late submissions will not be accepted.

Total Marks: 74

Submission Instructions: All assignment submissions must use the Moodle online submission system. The assignment is in two parts.

For Part A your submission should consist of a separate C source file for each implemented server and proxy server, with names clearly indicating which server or proxy it is for, and a single documentation file in either plain text or PDF format. With respect to the documentation file – think carefully about what the marker needs to know to evaluate what you have done. In addition to a description of what you have done, you should carefully describe any limitations that your solution has. If the marker discovers limitations that you have not described in your documentation, the marker will conclude that your testing was insufficient and you will be docked marks accordingly.

For Part B your submission should consist of a single plain text or PDF file.

PART A

In this part you will implement several servers for a simple look-up service, in C. Solutions should work on the Department's Linux machines. The service provides storage and retrieval of (key, value) pairs. Keys are 10 character ascii strings. Values are ascii strings with length at most 200 characters. There can be at most one (key, value) pair stored in the system for any particular key. The following operations should be supported:

- **add** key value : add (key, value) pair, if no existing pair with same key value
- **getvalue** key : return value from matching (key, value) pair, if any
- **getall** : return all (key, value) pairs
- **remove** key : remove matching (key, value) pair, if any

Note that our focus is on the network communication, rather than how the (key, value) information is stored and retrieved, for which you can use any simple methods you wish. In particular, you can use an in-memory data structure, and can assume some fixed limit on the number of (key, value) pairs that will be added (if you want to use array storage). For simplicity, you don't need to support concurrent TCP connections, and connections can be non-persistent. However, you are encouraged to think about what complexities would arise in your implementations if you did need to support concurrent TCP connections, and/or persistent connections.

You must use the proper socket API functions for TCP and UDP servers and clients as described for example in Beej's guide to network programming. Note that with our lab configuration, only port numbers between 30000 and 40000 should be used. Program defensively, checking for possible error conditions.

1. (*20 marks*) Design and implement a TCP-based server. It should be possible to test your server by using `netcat` to connect to the appropriate port.
2. (*10 marks*) Design and implement a TCP-based "proxy" server. It should be possible to test your proxy by using `netcat` to connect to the appropriate port of your proxy server. Your proxy should take as command line arguments the host name and port number for your server from question 1, which it will need to connect with.
3. (*20 marks*) Design and implement a UDP-based server. Also implement a proxy server that uses TCP to communicate with clients, and UDP to communicate with your UDP-based server. For **getall**, each (key, value) pair should be returned in a separate UDP segment. It should be possible to test your solutions by using `netcat` to connect to the appropriate port of the proxy. Your proxy should take as command line arguments the host name and port number for your UDP-based server, which it will need to connect with. For simplicity, you can make the assumption that UDP segments are never lost.

PART B

1. (*4 marks*) Consider two nodes connected by a single dedicated link within an OCN (on-chip network), SAN (system/storage area network), LAN (local area network), or WAN (wide-area network), and suppose that we wish to transmit a single 100 byte (including header) packet from one node to the other. Calculate the **total delay** and the **percentage of the total delay that is propagation delay** for each network, assuming that:
   - the link data rate is 8 Gbps (i.e., 8×10^9 bps);
   - there is no queuing delay;
   - the total node processing delay (not overlapped with any other component of delay) is x + (0.5 nanoseconds/byte), where x (the portion of this delay that is independent of packet size) is 0 microseconds for the OCN, 0.3 microseconds for the SAN, 3 microseconds for the LAN, and 30 microseconds for the WAN; and
   - the link distances are 0.5 cm, 5m, 5000m, and 5000 km, for the OCN, SAN, LAN, and WAN, respectively, with the speed of signal propagation in each case equal to 200,000 km/s (approximately 2/3 of the speed of light in a vacuum).

   Be sure to show your work, both for this question and for subsequent questions, so that you can get part marks even if your numerical answers happen to be incorrect.

2. (*4 marks*) Consider a message consisting of *n* packets, each of 4000 bits (including header), that traverses *k* links in a store-and-forward packet-switched network. The

propagation delay on each link is 1 ms. The transmission rate on each link is 100 Mbps (i.e., 100 x 10^6 bps). Neglecting processing and queueing delays, and assuming that the packets are sent back-to-back without intervening delays, give an expression for the total delay from when the first bit is sent at the source until all packets have been completely received at the destination:

(a) for the special case of $n = 2$ and $k = 2$,

(b) for general n and k (i.e., as a function of these two variables).

3. (*4 marks*) Give the maximum data rate, as measured in both symbols/second and bps, for transmissions on a channel with bandwidth H = 4 KHz, assuming that the transmitted symbols have 8 possible values, for each of the following two values of the signal to noise ratio. (Hint: use whichever of Nyquist's theorem and Shannon's theorem imposes the tightest constraint.)

   (a) Channel has a signal to noise ratio S/N = 127.

   (b) Channel has a signal to noise ratio S/N = 7.

4. (*4 marks*) Consider a scenario in which 100 sessions share a link of data rate 10 Mbps (i.e., 10 x 10^6 bps). Suppose that each session alternates between idle periods (when no data is being sent) of average duration 1 second, and busy periods of average duration B.

   (a) Supposing that circuit switching is used (i.e., channel capacity is evenly divided among the sessions, with a session's allocated capacity unused when the session is idle), give the achievable data transmission rate a session can achieve during its busy period.

   (b) Give an estimate of the average achievable data rate of each session (during its busy period) when packet switching is used, for B = 10 seconds, B = 1 second, B = 100 milliseconds, and B = 10 milliseconds, assuming:

      - at each point in time the link capacity is shared approximately equally among all of the sessions currently in their busy period; and
      - when a given session is in its busy period, the number of other busy sessions can be approximated by the total number of other sessions (99) times the fraction of time that each is in its busy period.

5. (*6 marks*) Consider a source and destination pair connected by a network path of capacity 10 Mbps and an average round-trip time of 50 ms (measured from when the source transmits the first bit of a packet until the destination's ack is received). Assume a packet size of 1000 bytes, and that the loss probability is negligibly small.

   (a) Give the maximum achievable data transfer rate in Mbps, assuming use of the *stop-and-wait* reliable data transfer protocol.

(b) Give the maximum achievable data transfer rate in Mbps, assuming use of a *sliding window* reliable data transfer protocol with maximum sending window sizes of 1, 10, and 50.

(c) Assuming again use of a sliding window protocol, how big would the maximum sending window have to be to ensure that, when all packets and acks are received correctly, the source would never have to pause its transmissions to wait for an ack to be returned?

6. (*2 marks*) Consider a communication link using a *go-back-n* sliding window protocol, with a maximum sending window size of 5, and a 4-bit sequence number (i.e., sequence numbers range from 0 through 15). Suppose that the receiver has received (and transmitted acks) for frames with sequence numbers 0,1,…,13. List the possible sequence numbers of the frame that the receiver **could receive next** (but not necessarily accept!), in order from the "oldest" frame, to the "newest" frame.