

The University of Queensland
School of Information Technology and Electrical Engineering
Semester Two, 2011
COMS3200 / COMS7201 – Assignment Two (Version 3.1)
Part A Due: 4pm Monday October 10, 2011
Part B Due 4pm Monday October 17, 2011
Weighting: 17% of final grade

Introduction

This assignment has two parts: Part A and Part B:

- Part A addresses a variety of issues related to the behaviour and performance of the communication protocols.
- Part B is related to networking programming. In this part of the assignment students are required to develop the networked application with the behaviour described in Assignment 1 but this time using UDP for communication between its components.

This is an **individual assignment**. You may discuss aspects of the design, programming and the assignment specification with fellow students, but should not actively help (or seek help from) other students with the actual design and coding of the assignment solution. It is cheating to look at another student's code and it is cheating to allow your code to be seen or shared in printed or electronic form. You should note that submitted code will be subject to checks for plagiarism and collusion. If we detect plagiarism or collusion, formal misconduct proceedings will be initiated against you. A likely penalty for a first offence would be a mark of 0 for the assignment. If you're having trouble, seek help from the tutor or the lecturer – don't be tempted to copy another student's code. You should read and understand the statements on student misconduct in the course profile and on the school website: http://www.itee.uq.edu.au/about_ITEE/policies/student-misconduct.html

Part A (7% - 70 marks)

Q1 [2% - 20 marks]

Consider a reliable Link Layer Protocol for a point to point link with the following characteristics.

- Data frames are 1025 bytes long, including a preamble, header and footer of a total of 25 bytes.
- data rate is 1 Gbps
- propagation speed is 200,000 km/s
- 6 bits are allocated for the frame sequence number
- Assume no packets are lost, unless explicitly stated.
- Assume that ACK/NAK frames are 25 bytes
- Assume no processing delay in the controllers
- Assume a timeout of 1ms after the end of packet transmission for unacknowledged packets
- Assume that data packets are continually available to be sent.
- Assume that we are sending data in one direction only.
- Assume that Go-back-n and Selective Repeat send both ACK and NAK packets.

If the link is 50km long, find the performance in terms of effective maximum data rate (of payload data) in bps of

- (a) Stop-and-wait protocol
- (b) Go-back-n protocol
- (c) Selective-repeat protocol

If the link is 200m long, find the performance in terms of effective maximum data rate (of payload data) in bps of

- (d) Stop-and-wait protocol
- (e) Go-back-n protocol
- (f) Selective-repeat protocol

If the link is 1000m long, and one data packet is not received, what is the delay between the original packet transmission and the time that the same packet is retransmitted for each of the following:

- (g) Stop-and-wait protocol
- (h) Go-back-n protocol
- (i) Selective-repeat protocol

(j) Which of the above reliable transmission protocols, if any, does Ethernet use?

Q2 [2% - 20 marks]

Consider a TCP segment arriving with the following header field values:

- Source Port: 4000 (decimal)
- Destination Port: 80 (decimal)
- Sequence number = 4245 (decimal)
- URG = 0
- ACK = 0
- PSH = 0
- RST = 0
- SYN = 1
- FIN = 0
- WSOption: Kind=3 Length=3 Shift.cnt=10 (decimal)

Write down the value of these and all other header fields which need to be set for the segment that would be sent in reply to this. Assume no options fields other than WSOpt in the header. Assume that the receiver wishes to use a WSOpt Shift.cnt of ten bits and a receive window size of 1MByte. Assume no congestion information in the header flags.

For all values except individual bit-flags, write the value in **HEXADECIMAL**.

For those fields where the value is not used the value should be listed as 0

For those fields which are used, but where the value is not known from the above information list these as “?”, and explain how the values would be chosen or calculated.

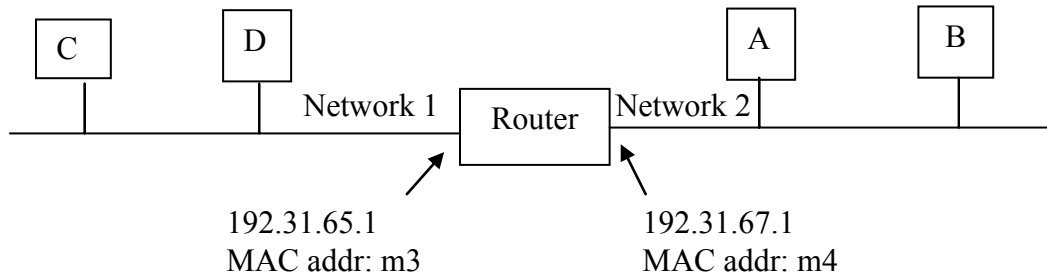
The answers can be placed in a table as follows:

Field	Value in Hexadecimal + Explanation for any values shown as ????
Source Port	
Destination Port:	
Sequence Number	
Acknowledgement number	
TCP Header Length	
CWR	
ECE	
URG	
ACK	
PSH	
RST	
SYN	
FIN	
Window Size	
Checksum	
Urgent pointer	
First 32-bit Option Word	

Q3 [3% - 30 marks]

Consider the following internetwork, consisting of two broadcast-access networks.

c.domain.com	www.domain.com	a.domain.com	b.domain.com
192.31.65.3	192.31.65.2	192.31.67.5	192.31.67.6
MAC addr: m1	MAC addr: m2	MAC addr: m5	MAC addr: m6



Each host on the two networks has the domain names, IP addresses and (simplified) MAC addresses shown. The router is connected to both networks and has the IP addresses and MAC addresses shown for the two network interfaces. Host A serves as the authoritative DNS server for domain.com. Each host is aware of its subnet mask, the IP address of the gateway (i.e. the particular router interface address), and the IP address of the DNS server. Assume that at the beginning of the communication exchange described below no host has ARP or DNS information cached (other than host A knowing DNS information), however once it acquires such information it will be cached. Also assume that no ARP probes or gratuitous ARP requests are sent.

Consider a user on host B clicking on a URL of the form “www.domain.com/.....” and getting a response from the web server. Assume that only one request is sent and host B finishes communication with the web server. Write down the sequence (in time) of ARP and IP packets that will be transferred to complete this request. You should clearly specify which host (and interface) sends each packet, which host(s) receive (i.e. process) each packet and what the contents of the packet are. You may assume that no transport or IP level fragmentation is necessary and that no packets are lost. ACK may be piggybacked with other packets or sent separately, both are OK. As an example of the level of detail required, the first frame in the sequence is provided for you. Your answer should be a table with the headings shown below. You should include this first frame in your table.

Sender (MAC Address) [IP Address]	Receiver(s) (MAC Address) [IP Address]	Contents:
B (m6) [192.31.67.6]	Broadcast (FFFFFFFFFFFF) [192.31.67.5]	ARP request: "Who owns IP address 192.31.67.5? My IP address is 192.31.67.6" (Host B needs to find the MAC address of the DNS server.)
...

Assessment Criteria for Part A

Provided your assignment is submitted using the instructions below it will be marked as indicated in the assignment questions. The marks are allocated for correctness. Partial marks will be allocated for answers that are partially correct.

Submission Instructions for Part A

- Use only A4 paper
- **Staple** only in the top left corner, as indicated
- Do not use binders, folders, clear plastic sheets, clips, etc.
- Attach an EAIT COMS3200 or COMS7201 cover sheet.
- You **MUST** sign the declaration regarding originality. **Failure to do so will result in a mark of 0 for Part A of this assignment.**

Submission Time and Place

Part A Assignments are due at **4pm on Monday October 10**. You must submit Part A of your assignment to the

EAIT assignment chute on level 2 of Hawken Building. You are advised to keep a copy of your assignment.

Part B (10%) - 100 marks

The aim of Part B of the assignment is for students to gain experience in network programming using socket level primitives for the UDP protocol. In this part of the assignment students are required to develop the networked application with the functionality described in Assignment 1 but this time using UDP for communication between the application's components.

Specification

Languages

You must implement the whole application in either Java or C using socket level primitives.

Java Requirements

Your implementation must consist of the 5 classes: Investor, Broker, Exchange, Registry, and NameServer which will be in files Investor.java, Broker.java, Exchange.java, Registry.java and NameServer.java. Each of these classes will contain a `public static void main(String args[])` method. **You may also submit extra .java files which contain code that is used by some or all of the five main files, eg. to implement reliable communications.**

C Requirements

If you choose to use C, your implementation must consist of any needed .h and .c files as well as a Makefile. When `make all` is performed, your Makefile must compile and link your source files into five executable programs: `Investor`, `Broker`, `Exchange`, `Registry`, `NameServer`. **Your code must be compiled with the `-Wall` flag.**

Other requirements

All communication is to be UDP based. All the communication has to be reliable and the reliability should be ensured by the communicating processes (at the application layer), i.e. the sender process needs to set timeout for an ACK arrival and retransmit the message if the timeout expires. The receiver has to acknowledge received messages. You are only allowed to use **one** port for sending and receiving UDP messages (one port per process), i.e. you cannot let the client contact the server on the server's UDP port, get the server to create a new UDP port and send a message back to the client using the new port. Except for the NameServer, the port number should be allocated by the system, i.e. the process should not specify the port number. Multi-threading is allowed for implementation of the communication reliability, i.e. the sending process may create a separate thread that reacts to timeout expiry and retransmits the required packet. The thread uses the same port as the process that instantiated the thread.

If a data packet is transmitted and after that it does not receive the corresponding ACK within 0.01 s, a retransmission is started. The maximum number of tries including the first try for a data packet is 10. If the number of retransmissions reaches the maximum, the following message will be output to standard error depending on the destination of the packet, **and the sending process will exit with status 1.**

1. "Investor is not reachable\n <contents of message> \n"
2. "Broker is not reachable\n <contents of message> \n"
3. "NameServer is not reachable\n <contents of message> \n"
4. "Exchange is not reachable\n <contents of message> \n"
5. "Registry is not reachable\n <contents of message> \n"

ACK packets sent in response to data transmissions do not need to be ACKed but you should also simulate the loss of ACK packets. In this case, no timer is needed to schedule ACK retransmissions.

In the cases where multiple packets are sent as part of a transaction, we assume that if the first data packet is not ACKed, the second data packet should not be transmitted.

Also, be careful to distinguish ACK packets from server reply messages. For example a request to the NameServer should consist of the following transmissions:

1. Client Request message to server (possibly with retransmissions)
2. Client Request ACK from server
3. Server Reply from server (possibly with retransmissions)
4. Server Reply ACK from client

As the local area network on which you test your application is reliable and does not lose packets it is necessary to simulate packet loss to test introduced reliability. Please simulate the packet loss by using a random number generator and deciding based on the generated value whether the packet is sent or not. The following pseudo-code shows how packet loss can be simulated:

```
set constant FAIL = 0.3; generate uniform random number x from 0 to 1;
if(x >= FAIL)
    transmit packet;
```

Assessment Criteria

Provided your assignment is submitted following the instructions below, it will be marked according to the following criteria. You must pay careful attention to the details of any required behaviour. Part marks may be awarded for a given criteria if the specification is partially met. All marking will be performed in a Linux environment, specifically `moss.labs.eait.itee.uq.edu.au` and it is expected that your code will work in this environment. Note that some criteria can only be tested for if other criteria are met.

Investor [15 marks]

- Code compiles successfully (1 mark)
- Investor correctly deals with invalid command line arguments (1 mark)
- Investor correctly prints out error message and exits with the correct status when unable to contact the Name Server (1 mark)
- Investor correctly displays the response for a quote request (3 marks)
- Investor correctly displays the response for a sale request (3 marks)
- Investor correctly displays the response for a buy request (3 marks)
- Communication is reliable (3 marks)

Name Server [15 marks]

- Code compiles successfully (1 mark)
- Name Server correctly deals with invalid command line arguments (1 mark)
- Name Server correctly deals with being unable to listen on the port number (1 mark)
- Name Server correctly responds to a valid query (3 marks)
- Name Server correctly registers an item when receiving a valid register request (3 marks)
- Name Server correctly responds to an invalid register or query request (3 marks)
- Communication is reliable (3 marks)

Broker Server [25 marks]

- Code compiles successfully (1 mark)
- Broker correctly deals with invalid command line arguments (1 mark)
- Broker correctly sends a valid register request to the Name Server (2 marks)
- Broker correctly sends requests to the Exchange Server (5 marks)
- Broker correctly responds to a quote request sent from an Investor (3 marks)
- Broker correctly responds to a sale request sent from an Investor (3 marks)
- Broker correctly responds to a buy request sent from an Investor (3 marks)
- Broker updates the record of last sale when receiving it from the Exchange Server (4 marks)
- Communication is reliable (3 marks)

Stock Exchange Server [35 marks]

- Code compiles successfully (1 mark)
- Exchange Server correctly deals with invalid command line arguments (1 mark)
- Exchange Server correctly sends a register message to the Name Server (2 marks)
- Exchange Server correctly sends a Registry lookup request to the Name Server (2 marks)
- Exchange Server correctly deals with a buy request forwarded from a broker (11 marks)
- Exchange Server correctly deals with a sale request forwarded from a broker (11 marks)
- Exchange Server sends a valid share registry information to the Registry Server (4 marks)
- Communication is reliable (3 marks)

Share Registry Server [10 marks]

- Code compiles successfully (1 mark)
- Registry Server correctly deals with invalid command line arguments (1 mark)
- Registry Server correctly sends a valid register message to the Name Server (1 mark)
- Registry Server correctly deals with information received from the Exchange Server (4 marks)
- Communication is reliable (3 marks)

Penalties that may be applied

- Code must be modified by marker to permit compilation and/or marking (-1 to -50 depending on the severity of the change required. Deduction is at the discretion of the course coordinator whose decision is final.)

Submission Instructions for Part B

Part B Assignments are due at 4pm on Monday October 17, 2011. You must submit Part B electronically via <http://submit.itee.uq.edu.au>. Only the last of your submissions will be marked. Your submission time will be considered the time of the last of your submissions. Assignments submitted in any other manner (e.g. email) will be discarded. You are advised to keep a copy of your assignment.

Late Submission of Part A or Part B

Late submission will be penalized by the loss of 10% of your Part A or Part B assignment mark per working day late (or part thereof). In the event of exceptional personal or medical circumstances that prevent on-time hand-in, you should contact the lecturer and be prepared to supply appropriate documentary evidence (e.g. medical certificate). Late Part A submissions must be made directly to the lecturer or tutors and late electronic submissions must be made via <http://submit.itee.uq.edu.au>. Emailed assignments will be discarded.

Modifications to Part A and Part B Requirements

Note: it is possible that there are inconsistencies in the above requirements and/or that not all details have been specified. Please ask if you are unsure of the requirements. Please monitor your email, the course newsgroup, or the course website for clarifications and/or corrections to the above information. It will be assumed that students see such email or postings by the end of the next business day. Requirements changes/clarifications emailed and/or posted by one of the teaching staff before 4pm Monday October 3 are considered to be part of the assignment requirements.

Academic Merit, Plagiarism, Collusion and Other Misconduct

You should read and understand the statement on academic merit, plagiarism, collusion and other misconduct contained within the course profile and the document referenced in that course profile. You should note that this is an **individual assignment**. **All submitted source code will be subject to plagiarism and/or collusion detection.** Work without academic merit will be awarded a mark of 0.

Updates – Version 2.0 , 19/9:

Added some additional explanation to Part A – Q1:

- Assume a timeout of 1ms after the end of packet transmission for unack'd packets
- Assume that we are sending data in one direction only
- Assume that Go-back-n and Selective Repeat send both ACK and NAK packets

Q3. Added clarification: ACK may be piggybacked with other packets or sent separately, both are OK.

Version 2.1, 20/9. Added explanation to behaviour when a host is unreachable (bottom of page 6):- If the number of retransmissions reaches the maximum, the following message will be output to standard error depending on the destination of the packet, **and the sending process will exit with status 1.**

1. “Investor is not reachable\n <contents of message> \n” ...etc....

Version 2.2, 22/9

Probability of packet failure (page 7) changed from 50% to 30%.

Version 3, 28/9 – Extension for Part B until 17 October, Part A still due 10 October.

Version 3.1 5/10: Change to allowable java submission: - You may also submit extra .java files which contain code that is used by some or all of the five main files, eg. to implement reliable communications.