Operating Systems **Homework 5**
CS4410 Summer 2013 **Due Monday, 24 June 2013**

1 Network Protocols

1. Why do we use protocols for the communication between computers and why do we have multiple layers of protocols?

2. Here, we compare network protocols with a function call. Remember that in order a process to call a function, it should put the arguments in the stack and then jump to the function. When the function ends, it puts the result in a register or in the stack, to be founded by the caller. Is this description a kind of protocol? Why? Give two similarities between the function call and some of the communication protocols.

3. Here, we compare the network communication between applications, with the communication of threads through shared variables, in the common address space. Remember that if two threads want to communicate, a common variable is initialized and then both can update and read from it. What is a tool we learned that imposes a protocol for the communication between the threads? Give two similarities between this protocol and some of the network protocols.

2 Link Layer

(a) What is the main reason we use Link Layer protocols?

(b) Why do we use Media Access Control at broadcast links? Which Link Layer device avoids the use of Media Access Control and how?

(c) In a LAN, where nodes use WiFi connections, does the Link Layer need to use Media Access Control? Can the device you picked at subquestion (b) avoid the use of Media Access Control in this case, too? Please, justify your answers.

(d) Consider the LAN below, and assume that we use the Ethernet as a Link Layer protocol.

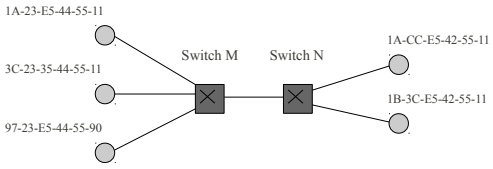

Figure 1: LAN

If the node 1A-23-E5-44-55-11 (source) wants to send a frame to the node 1B-3C-E5-42-55-11 (destination):

(a) At which position in the frame do we have to save the destination MAC address?

(b) Assuming that the forwarding tables of the switches are updated with the proper entries (*MAC address*, *gateway*), describe how the frame will be forwarded from the source to the destination.

(c) Does switch M change the MAC address of the destination, in the header, when it forwards the frame to switch N? Why?

3 Network Layer

1. What is the main reason we use Network Layer protocols?

2. Suppose we have in our disposal this subset of IP addresses: 24.42.xxx.xxx. If we are the administrators of the network below (Figure 2), how would we distribute the addresses to the nodes? Explain, also, how many different subnetworks we should define and which part of the IP address should identify each subnetwork. Remember that each network interface of the routers should have a different IP address.

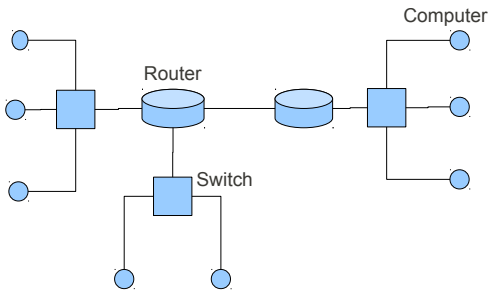

Figure 2: Network

3. Suppose we are examining the network in Figure 3. Also, suppose that Computer B belongs to the subnetwork 124.46.6.0\24 (mask = 255.255.255.0). If Computer A sends a packet to Computer B, what is the path that the packet will follow through the network? What is the IP address and the MAC address inside the headers of the packet when the packet starts its journey from the Computer A? At which points do they change (if they change) and why? In which header (or protocol layer) each network device (switch, router) is interested in and how does it decide where to forward the packet? Assume that the forwarding tables of the switches and the routing table of the router are updated with the positions of nodes and subnetworks, respectively.

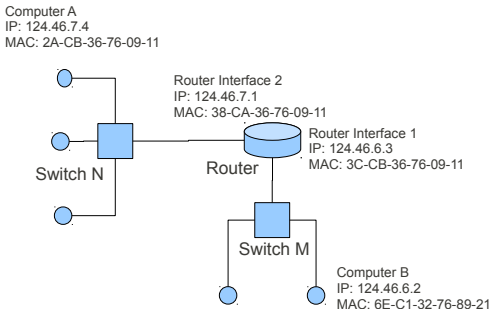

Figure 3: Network

4 Transport Layer

1. Why do we use Transport Layer protocols?
2. For which applications would you use the UDP protocol and for which the TCP protocol?
3. Computer A wants to send the text bellow to Computer B using the TCP protocol for the Transport Layer. Figure 4 shows how the text is split into segments. The numbers correspond to bytes that belong to each segment. For example, 512 corresponds to the first byte of the second segment. Using the simple idea of TCP (one data packet, one acknowledgment packet), describe the sequence of packets that are exchanged, referring the kind of each packet (data,ack) and the sequence number that they convey (for data packets the sequence number of the frame, for the acks the sequence number that they expect to get next).

0... 511	512 ...1023	1024 ...1535	1536... 2047

Figure 4: Text

5 Application Layer

For this exercise we will implement an application layer protocol. We will write two programs. One program will play the role of a server and the other the role of a client. They will communicate with each other using sockets. The client will give the arithmetic representation of an addition to server (ex.

3+4+7+90) and the server will compute the addition and it will send back to the client the result (ex. 104). The protocol that we impose in order the data to be transfered in a proper way between server and client is the following:

- The client sends a message to the server with the string "Hello!".
- The server takes the message and it replies with the message "Hi!\nWhat is your name?".
- The client replies "My name is <name>".
- The server replies "Welcome <name>!\nGive me the addition".
- The client replies "Addition: <arithmetic representation>".
- The server replies "The result is: <result>. More additions?".
- The client repeats the 5th step if it wants to give another addition to the server, or it replies with "Quit".
- In the second case, the server replies with "Buy!".

In reality, the user will type the replies in a terminal (stdin), the client will read the replies and it will simply send the replies to the server. Also, the client will receive the responses from the server and it will print them to the terminal to be read by the user. Here is an example:

Server Terminal:
Hi! I am the addition server!
Connection 1 accepted!
Connection 1 closed!

Client Terminal:
The addition client is up!

Hello!
Hi! What is your name?
My name is Helen
Welcome Helen!
Give me the addition.
Addition: 1+2+3+4
The result is 10. More additions?
Addition: 45+789+10003
The result is 10837. More additions?
Quit
Buy!

Now, try to fill the code gaps of the server.py. When you want o test the programs, first you should start the execution of the server and then the execution of the client. You can change the port number whenever you want, but be careful not to use "well-known" port numbers. Be careful with the spaces and the \n.