

Network Programming - Spring 2003
Midterm Exam

My Name is: Sample Answers

This test contains 6 pages. Please make sure your test packet is complete.

Don't feel you need to fill up all the space on each page (there is plenty of room for answers).

Question 1 (10 pts): TCP/IP

1A (5 pts): Establishing a TCP connection involves the exchange of 3 packets (three-way handshake). Why is a 3-way handshake used and not 2-way or 4-way?

TCP is reliable, so the creation of the initial connection must be reliable.

If only 2 messages (packets) were used, as long as everything went well, both the client and server would know the other's ISN. However, if the second message is lost, the server has no way of knowing this.

Only after the third message do both endpoints know that the other endpoint knows both ISNs – at this point there is a reliable connection.

A 4th message would only let the client know that the server knows that the client knows both ISNs – this is not required for a “reliable” service.

1B (5 pts): Describe how you would go about providing a stream-based service using UDP (a stream-based service ensures that data order is maintained).

The basic idea is to put a sequence number in each UDP datagram, perhaps as the first few bytes in the datagram (similar to TFTP block numbers). The remaining bytes constitute part of the (virtual) stream of bytes. A receiving application could reconstruct the stream by looking at the sequence numbers.

IMPORTANT GRADING NOTE: Using ACKs (alone) is not sufficient! Duplicates can be delayed and arrive out of order – this would mess things up unless there are sequence numbers. So: ACKS alone are not enough, and in fact ACKS don't help provide a stream-based service (only a reliable server, and this question is not about reliability).

Question 2 (20 pts): True/False 2 pts each. Be careful!

If a UDP client calls `connect()` and there is no UDP server running at the destination address, `connect()` will return an error.

TRUE

FALSE

If a TCP client calls `connect()` and there is no TCP server running at the destination address, `connect()` will return an error.

TRUE

FALSE

A CGI program runs on the machine that is also running the HTTP client program.

TRUE

FALSE

An HTTP client using a proxy server doesn't need to be able to resolve hostnames (to browse the web).

TRUE

FALSE

Buffer overflow is a potential problem only for servers, it's not a problem for clients.

TRUE

FALSE

All numeric values sent in an HTTP request (in the headers) must be in network byte order.

TRUE

FALSE

It is possible to develop a connection-oriented transport layer service by using a connectionless network service.

TRUE

FALSE

UDP based TFTP provides reliable file transfer.

TRUE

FALSE

HTTP is a network layer protocol.

TRUE

FALSE

I will do lots of work during Spring Break.

TRUE

FALSE

Question 3 (20 pts): Matching - choose (from among the choices listed below) the word/phrase that best matches each statement.

CHOICES:				
Physical Layer	TCP/IP	Sockets	UDP	ARP
Datalink Layer	RARP	FIN	LAN	WAN
Network Layer	End-to-End	Checksum	ACK	SYN
Transport Layer	ICMP	Sequencing	Flow Control	Hop-by-Hop
Port number	IPv4 Address	OSI	ISO	CGI
Bridge	Repeater	Router	Hub	Gateway
HTTP	TFTP	TELNET	DNS	SSI

Provides communication of chunks of bytes between end-systems that share a communication medium.

Datalink Layer

Provides communication between processes running on end-systems on possibly different networks.

Transport Layer

32 Bit identifier use to route IP datagrams.

Ipv4 Address

16 bit identifier used to deliver TCP segments and UDP datagrams to the correct process

Port Number

A widely used protocol suite.

TCP/IP

A device that forwards datalink-layer messages (frames) from one *wire* to another.

Bridge

Used to lookup Ethernet addresses given an IP address

ARP

Often used for error detection.

Checksum

A standard interface between HTTP servers and external programs.

CGI

A transport layer protocol.

UDP

Question 4 (10 pts): Programming

The following code is given to you for incorporation in to a server you are writing. This code is used to help parse a message received from a client. This code was written by your boss...

```
/* FUNCTION: void removenewlines(char *);
   DESC: removes newlines ('\n' characters) from a message.
   AUTHOR: Bossy Bossman
   NOTES: I don't like pointers - I always use arrays.
*/

void removenewlines(char *s) {
    char tmp[100];
    int i=0,j=0;

    while (s[i]) {
        if (s[i] != '\n')
            tmp[j++] = s[i];
        i++;
    }
    tmp[j]='\0';

    strcpy(s,tmp);
}
```

Describe why you won't be putting this code in your server (you must be specific!), and fix/re write the code so that it is safe to put in your server.

There is a potential overflow when copying the string into the buffer tmp. We have no way of knowing that the original string (minus any newlines) can fit into tmp, and there is not code to prevent the code from writing beyond tmp.

To fix the problem, we could change the while loop to this:

```
while ( (s[i]) && (j<99))
```

The strcpy is dangerous only if the string in tmp is never null terminated. By itself, the strcpy is **NOT THE PROBLEM**.

Notes for graders:

5 points for correctly identifying the problem.

5 points for solving the problem (fixing the code).

Question 5 (16 pts): Case Studies:

5A (4 pts): How does a TFTP *file receiver* know when it has received a complete file?

When it receives a DATA message with less than 512 bytes of actual data. The entire message length (including the opcode and block number) will be less than 516 (all other DATA TFTP Datagrams are 516 bytes total).

5B (4 pts): How does an HTTP 1.1 server know when it has reached the end of a *complete* HTTP/1.1 request?

For GET or HEAD: when it reads the blank line that terminates the request headers.
In other words: `\r\n\r\n`.

For POST: when it's read Content-length bytes beyond the blank line that ends the headers.

5C (4 pts): Explain how the TELNET protocol sends both control information (TELNET commands) and data (payload) over a single *connection*.

There is a special byte value (IAC - 255) that is interpreted as being special – it indicates that the next byte is actually control information. If a 255 is followed immediately by another 255, this means one actual byte of data with the value 255.

5D (4 pts): Describe the network traffic that results from a hostname lookup (like a call to `gethostbyname ()`). For this question, assume that the hostname-to-address mapping is not cached locally or at any DNS server. You must explicitly state whether your description assumes that recursion is used (or not).

Assuming recursion:

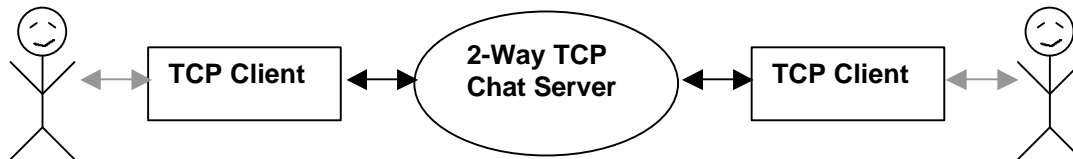
request is sent to local name server. If local name server doesn't know the answer it sends a request to its name server (or a root server). root server may contact the actual name server for the domain in the query, get the answer and the answer trickles back to the client through its local name server.

Without recursion:

request is sent to local name server. If local name server doesn't know the answer, it sends the client a response that tells it who to ask. The client now repeats the process with the new server (possibly multiple times).

Note that the local name server may ask a root server for the address of the name server that can answer the question, and then send this to the client (or it might tell the client to ask the root server itself).

Question 6 (24 pts): Programming



A simple TCP based chat server could allow users to use any TCP client (telnet, for example) to communicate with each other. For this question you should consider a single process, single thread server that can support exactly 2 clients at once, the server simply forwards whatever is sent from one client to the other (in both directions). Your server must not insist on any specific ordering of messages – as soon as something is sent from one client it is immediately forwarded to the other client. As soon as either client terminates the connection the server can exit. Provide a high-level description of this program (a few lines describing how the program works), and then provide as much of the code as possible. Pseudo code is fine (we won't be compiling your code!), but make sure you mention the name of any system calls you are using.

```
// create passive TCP socket:
passive=socket()
bind(passive,...)
listen(passive,...)

// wait for first client
first = accept(passive,...);

// wait for second client
second = accept();

// loop until someone closes
while (1)
    // set up fdset to give select
    FD_ZERO(set); FD_SET(first,&set), FD_SET(second,&set);
    // call select
    select(max(first,second)+1, &set, ...)
    // check for data coming from first
    if (FD_ISSET(first,&set) {
        // read from first
        n=read(first,buff,MAX);
        // if connection was closed - break out of loop
        if (n==0) break;
        // write to second
        n=write(second,buff,n);
    }
    // check for data coming from second
    if (FD_ISSET(second,&set) {
        // read from second
        n=read(second,buff,MAX);
        // if connection was closed - break out of loop
        if (n==0) break;
        // write to first
        n=write(first,buff,n);
    }
}
```

**Must happen
every time select
is called!**

NOTE TO GRADERS:

General idea is to use select() to do multiplexing. At least 10 points off for any solution that does not actually have a chance of working (doesn't use select or poll or something effectively the same).

No points depend on knowing the details of how to call any specific system call, so grade should not depend on how close the code is to actual C code.

Grade should reflect how close the program described is to providing the service. Better solutions should receive better grades, for example a solution that uses non-blocking I/O (and becomes a CPU hog) is not as good as my solution using select. A program that uses alarm is also not as good as using select, as the response time will suffer.

Extra Credit (0 pts): Joe (a network programmer) has a chocolate chip cookie, a peanut butter cookie and an Oreo. In what order should Joe sample the cookies?