Assignment 3
Questions?
Today

• Network Communication
  – Introduction
  – Network Protocols
  – Client-Server Model (TCP)
  – Sockets API
Network Communication

- Computers are exchanging data via network links and intermediate devices (routers, switches…etc)

- Layering approaches are used, each layer does specific functionalities

- Each layer contains a set of protocols
Network Communication

• TCP/IP layering approach is used in the Internet

• Transport layer provides end-to-end packet delivery

• Network layer (IP layer) provides host-to-host packet delivery
Network Communication

• Programs must agree how to exchange their data ahead of time.
• Protocol is a format agreement between network programs (servers and clients)
• Application-Layer Protocols: HTTP, FTP, SMTP etc.
• Transport-Layer: TCP, UDP
Network Communication

- TCP: Connection-oriented, reliable data transfer: packets arrive in-order

- UDP: Connectionless, unreliable data transfer: may lose packets, packets may arrive out of order.
TCP Operation

Server Side:

1. Create socket
2. Bind Socket to port. This is the port used for incoming connection requests.
3. Listen to incoming connection requests
4. Accept connection and assign a port number to it.
5. Use/Create worker thread to handle communication with client (if server is multi-threaded)
TCP Operation

Client Side:
1. Create socket
2. Connect to server (need to specify address and port)
3. Read/write to server
Socket Creation

```c
int socket_fd=socket(AF_INET,SOCK_STREAM,0)
```

- Creates a handle for a communication endpoint
- `SOCK_STREAM` is used for TCP, `SOCK_DGRAM` for UDP
- The first parameter specifies the protocol family
Bind Socket

```c
bind(socket_fd, (struct sockaddr*) &server, sizeof(server));

struct sockaddr_in server
server.sin_family= AF_INET;
server.sin_addr.s_addr= htons(INADDR_ANY);
server.sin_port=htons((short) 5500);
```
Listen to Socket

`listen(socket_fd,100);`

- Listen is used to tell the system to allocate a queue for pending requests
- The number specifies the how many requests can be pending (in the queue) at any given time
Accept Clients

```c
int new_socket=accept(socket_fd, (struct sockaddr*) &client_addr, &size)

struct sockaddr_in client_addr;
int size=sizeof(struct sockaddr);
• Accept creates new socket that can be used to communicate with the client. So now we can send a message to the client:
  write(new_socket," Hello Client",13);
  close(new_socket);
```
On the Client Side

//create socket
Int sock=socket(AP_INET,SOCK_STREAM,0);

//set up server address struct
sockaddr_in server;
server.sin_family=AP_INET;
server.sin_port=htons((short)5500)
name2addr(“homer.itlabs.umn.edu” , &(server.sin_addr.s_addr)) ;

//create connection and communicate
connect(socket,(struct sockaddr*)&server, sizeof(server))
write(socket,” Hello My Server” ,16);
Communication Process

Client

socket()

bind()

connect()

write()

read()

close()

socket()

bind()

listen()

accept()

block until connection from client

read()

process request

write()

close()
Exercise

• Try running the sample code by executing the following:
  
  ./server <port number>

  ./client localhost <same port number as above>

• Modify the code so that:

  Client will accept messages repeatedly until input ‘quit’. Messages will be sent to the server.

• The server converts the message to upper case and sends it back until ‘quit’
Questions?