Introduction to Network Programming

Speaker: Jae Chung
CS4514 Requirements

- We expect that you have taken a programming course similar to CS2005 before coming into this class.
- Programs will be done in C or C++
- We also expect that you have OS (CS3013) programming background.
  - fork()
  - exec()
  - malloc() or new
CS4514 Project 1

- Your programs should compile and work on garden.WPI.EDU.
- This system is running Digital Unix (Ultrix).
- If you program is developed in another platform you should test the software on garden before turning in the assignment.
- Make sure you have the correct #include in your program.
Communication Model

1. (1) register
2. (2) list
3. (3) Connect One Service

Client

Oracle
DNS

Server

TCP
UDP
UDP
UDP
UDP Transmission (Client)

- Stevens (Page 212)
  - Connectionless
    - Specify transport address every time you send/recv data
  - Unreliable Protocol
    - Data lost, bit errors
  - Stevens (Page 216)
    - Simple UDP echo client
    - Lenon page78 (robust)

```
socket()
sendto()
recvfrom()
close()
```
Example: UDP Client

```c
struct hostent *hp;    /* /usr/include/netdb.h */
struct sockaddr_in server;  /* /usr/include/netinet/in.h */
int sd, lserver = sizeof(server);

/* prepare a socket */
if ( (sd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
    perror(strerror(errno));
    exit(-1);
}
```
Example: UDP Client (Continued)

/* prepare server address */

bzero( (char*)&server, sizeof(server) );
server.sin_family = AF_INET;
server.sin_port = htons( SERVER_PORT );
if ( (hp = gethostbyname(SERVER_NAME)) == NULL) {
    perror( strerror(errno) );
    exit(-1);
}
bcopy( hp->addr, (char*)&server.sin_addr, hp->length);
/* prepare your message */
...

/* send/receive data */

sendto( sd, sBuf, data_size, 0, (struct sockaddr*)&server, lserver );

recvfrom( sd, rBuf, MAXLEN, 0, (struct sockaddr*)&server, &lserver );

/* close socket */

close( sd );
TCP Connection (Client)

- Stevens (Page 86)
  - Connection Oriented
    - Specify transport address once at connection
  - Use File Operations
    - read() / write()
  - Reliable Protocol

- Diagram:
  - socket()
  - connect()
  - read() / write() / send() / recv()
  - close()
Example: TCP Client

```c
int sd;
struct hostent *hp;  /* /usr/include/netdb.h */
struct sockaddr_in server;  /* /usr/include/netinet/in.h */

/* prepare a socket */
if ( (sd = socket( AF_INET, SOCK_STREAM, 0 )) < 0 ) {
    perror( strerror(errno) );
    exit(-1);
}
```
/* prepare server address */

bzero((char*)&server, sizeof(server));
server.sin_family = AF_INET;
server.sin_port = htonl(SERVER_PORT);

if ((hp = gethostbyname(SERVER_NAME)) == NULL) {
    perror(strerror(errno));
    exit(-1);
}

bcopy(hp->addr, (char*)&server.sin_addr, hp->length);
Example: TCP Client (Continued)

/* connect to the server */
if (connect( sd, (struct sockaddr*) &server, sizeof(server) ) < 0 ) {
    perror( strerror(errno) );
    exit(-1);
}

/* send/receive data */
while (1) {
    read/write();
}

/* close socket */
close( sd );
TCP Connection (Server)

- Stevens (Page 86)
  - Bind transport address to socket
  - Listen to the socket
  - Accept connection on a new socket

```
socket()
bind()
listen()
accept()
read()/write()
close()
```
Example: TCP Server

```c
int sd, nsd;
struct sockaddr_in server;  /* /usr/include/netinet/in.h */

sd = socket( AF_INET, SOCK_STREAM, 0 );
bzero( (char*)&server, sizeof(server) );
server.sin_family = AF_INET;
server.sin_port = htons( YOUR_SERVER_PORT );
server.sin_addr.s_addr = htonl( INADDR_ANY );
```
Example: TCP Server (Continued)

bind( sd, (struct sockaddr*) &server, sizeof(server) );

listen( sd, backlog );

while (1) {
    nsd = accept( sd, (struct sockaddr *) &client, sizeof(client) );
    read() / write();
    close( nsd );
}

close( sd );
Some Useful System Calls

- **Gethostbyname**: map hostname to IP addr
  
  ```c
  struct hostent *gethostbyname( char *name )
  ```

- **Getservbyname**: look up service name given

  ```c
  struct servent *getservbyname( const char *servname, const char *protocol )
  ```

- **Gethostname**: get own hostname

  ```c
  int gethostname( char *name, size_t len )
  ```

- **Getsockname**: map sd to socket addr

  ```c
  int getsockname( int sd, struct sockaddr *sa, size_t *lsa )
  ```
Oracle Commands (om struct)

- **Find a service:**
  
  ```
  serv.ver = verCur;
  serv.cmd = cmdGet;
  serv.uid = ?;
  serv.sbServ = ?;
  ```

- **Register a service:**
  
  ```
  serv.ver = verCur;
  serv.cmd = cmdPut;
  serv.uid = ?;
  serv.sbServ = ?;
  serv.sbDesc = ?;
  serv.sa = ?
  ```

- **Clear a service:**
  
  ```
  serv.ver = verCur;
  serv.cmd = cmdClr;
  serv.uid = ?;
  serv.sbServ = ?;
  ```

```struct om serv```
Oracle Communication Example

int sd;
struct om sendMsg, recvMsg;
size_t lom = sizeof(struct om);

sendMsg.ver = verCur;
sendMsg.cmd = cmdGet;

sendto( sd, (void *)&sendMsg, lom, 0, &sa, lsa );
recvfrom( sd, (void *)&recvMsg, lom, 0, &sa, &lsa );
Turnin Your Files

- **Turnin Command**
  - `/cs/bin/turnin submit cs4514 proj1 [all files]`

- **Files should include**
  - All source code (including a Makefile)
  - A documentation file (include your compile command if you don’t offer a Makefile)
  - A result script showing the running result
  - Any custom include files that you used, including `oracle.h` if you have not used `#include “/cs/cs4514/pub/lib/oracle.h”`
Some functions that you may need:

- bind
- listen
- accept
- select
- sendto/send
- recvfrom/recv
- gethostbyname
- getservbyname
- gethostname
- getsockname
- fork
- strlen, strtok
UNIX Programming (Continued)

- Use man pages for help on a particular command or function
- Some man pages are not available on garden.WPI.EDU but available on CCC machines.
  
  `> man sendto`
  
  `> man 3 printf`
UNIX Debugging

- Compile program with \texttt{-g} flag
  - \texttt{g++ -g -o program program.cc}
  - \texttt{gcc -g -o program program.c}

- \texttt{gdb program \{core\}}
  - \texttt{set args (command arguments)}
  - \texttt{run, where, list, step, break}
  - \texttt{continue inspect, help, quit}

- Can examine specific data in program
UNIX Debugging (Continued)

- Many more options use help to learn more
- This will be useful to find out where a program crashes or seg faults
- Can set breakpoints to stop at specific line or function
- Can set specific data values in program
HELP

- Bring printouts to office hours.
- Email to TA mailing list with questions.
- You CAN email a specific TA or SA, but do not expect immediate results, better to use the TA mailing list.
- We do have a class mailing list that could be used as a last resort.