EECS 489 Discussion

Nitish Paradkar

General Info

- Office hours: Tuesday 2:00 - 4:00
- Discussion will contain:
  - Project overview/hints
  - Extra examples to help solidify lecture material
  - Any other things that you consider important

Assignment 1 Overview

REGISTER YOUR GITHUB USERNAME WITH US!!

- Measure throughput and latency across various links
- Use Mininet to create a virtual network
- Write a tool to calculate throughput
- Use ping to measure latency

Socket Programming Review (Server)

```
int sd;
struct sockaddr_in sin;

if ((sd = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) {
  perror("opening TCP socket");
  abort();
}

memset(&sin, 0, sizeof (sin));
sin.sin_family = AF_INET;
sin.sin_addr.s_addr = INADDR_ANY;
sin.sin_port = htons(server_port);

if (bind(sd, (struct sockaddr *) &sin, sizeof (sin)) < 0) {
  perror("bind");
  printf("Cannot bind socket to address\n");
  abort();
}
```

Socket Programming Review (Server)

- bind() used to tie a socket to a certain IP and/or port number
- Returns -1 on error
- You may find strerror(errno) useful to see specific errors

Socket Programming Review (Server)

```
if (listen(sd, qlen) < 0) {
  perror("error listening");
  abort();
}
```

- specifies max number of pending TCP connections allowed to wait to be accepted (by `accept()`)

Socket Programming Review (Server)

```
int addr_len = sizeof(addr);
int td;

td = accept(sd, (struct sockaddr *) &addr,
  &addr_len);

if (td < 0) {
  perror("error accepting connection");
  abort();
}
```

- waits for incoming client connection
- returns a connected socket ← different from the `listen`ed to socket

Socket Programming Review (Client)

- Construct a socket in the same way as a server

```
unsigned short server_port;
char *servername;            // both assume initialized
struct sockaddr_in sin;

struct hostent *host = gethostbyname(servername);

memset(&sin, 0, sizeof(sin));
sin.sin_family = AF_INET;
sin.sin_addr.s_addr = *(unsigned long *) host->h_addr_list[0];
sin.sin_port = htons(server_port);

if (connect(sd, (struct sockaddr *) &sin, sizeof (sin)) < 0) {
  perror("connect");
  printf("Cannot connect to server\n");
  abort();
}
```

`connect()` initiates connection (for TCP)

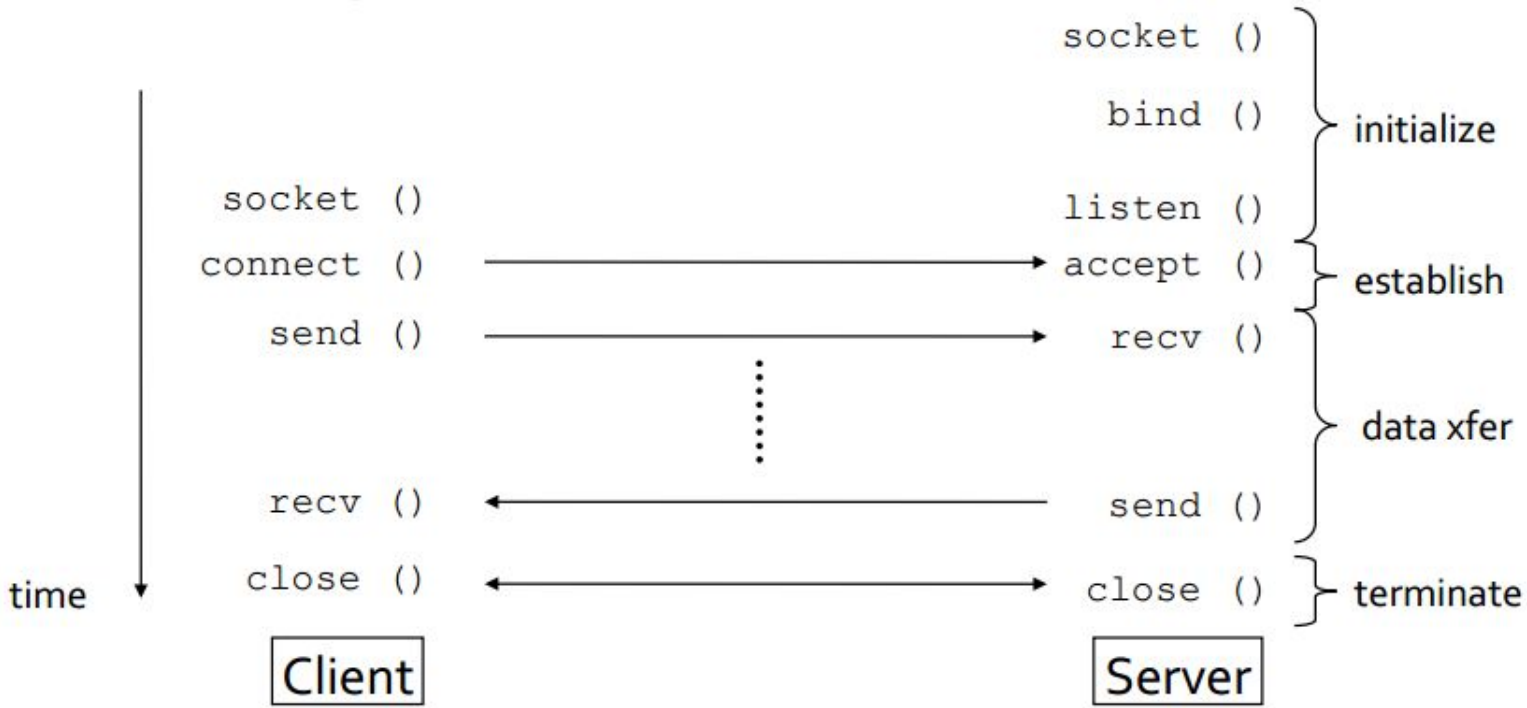
Socket Programming Review (Send/Receive)

- Send and recv return the number of bytes that have actually been sent or received
- Return 0 when the other side closes the connection
  - Also returns 0 if you specified to send or recv 0 bytes
- Return -1 on error

```
ssize_t send(socketfd, buffer, bytes_to_send, flags)

ssize_t recv(socketfd, buffer, bytes_to_recv, flags)
```

Socket Programming Review (Overview)



Assignment 1 Hints

- START EARLY - you could run into possible setup issues
- Look more into assignment1_topology.py
  - It actually shows you what kinds of results you should expect

Acknowledgements

- http://web.eecs.umich.edu/~sugih/courses/eecs489/lectures/02-Protocol+SocketsClient.pdf
- http://web.eecs.umich.edu/~sugih/courses/eecs489/lectures/03-SocketsServer.pdf