# Client Server App Using Protocols 4 & 5

•••

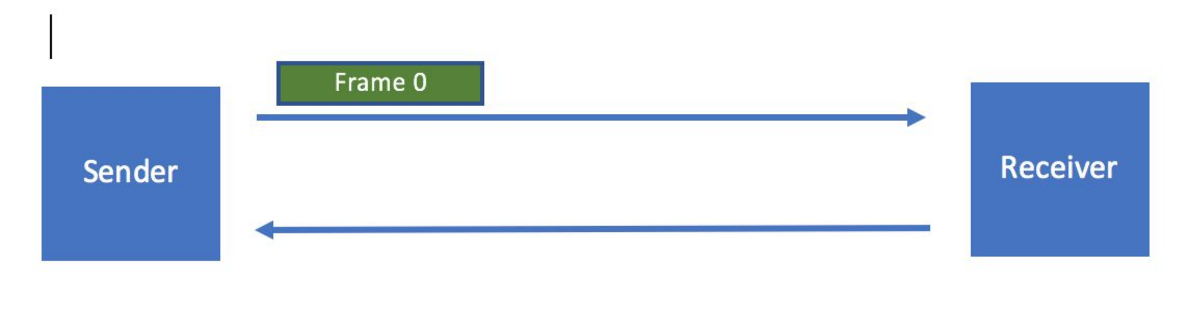
Adam Homann, Alex Preston, Bikram Singh, Zur Nachshon

# Introduction:

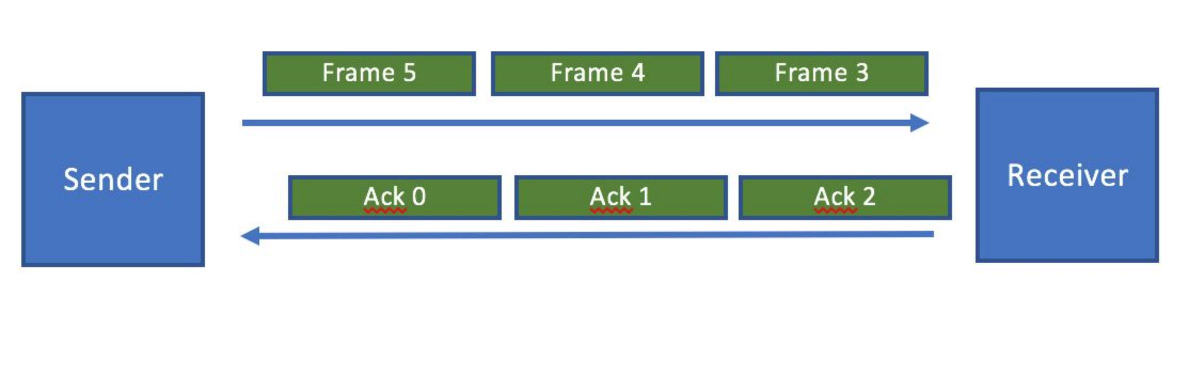
- Communication
- Protocol
- Client Server

# Protocols

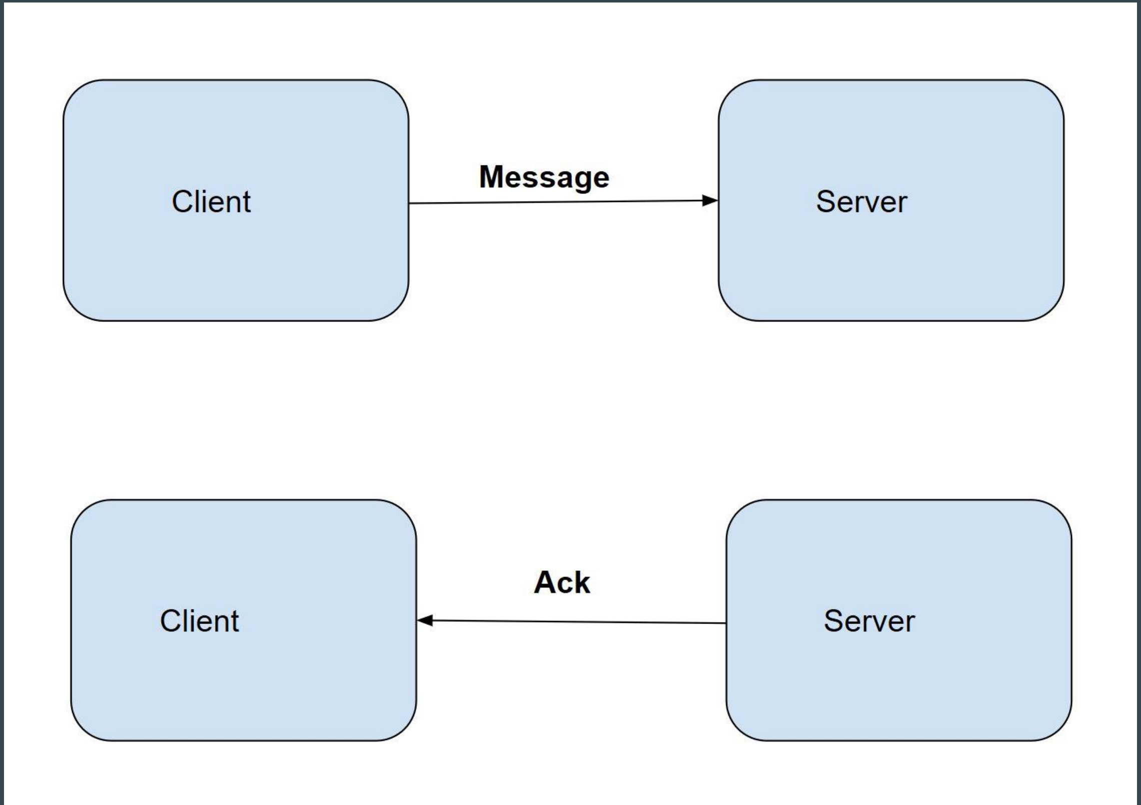
Protocol 4



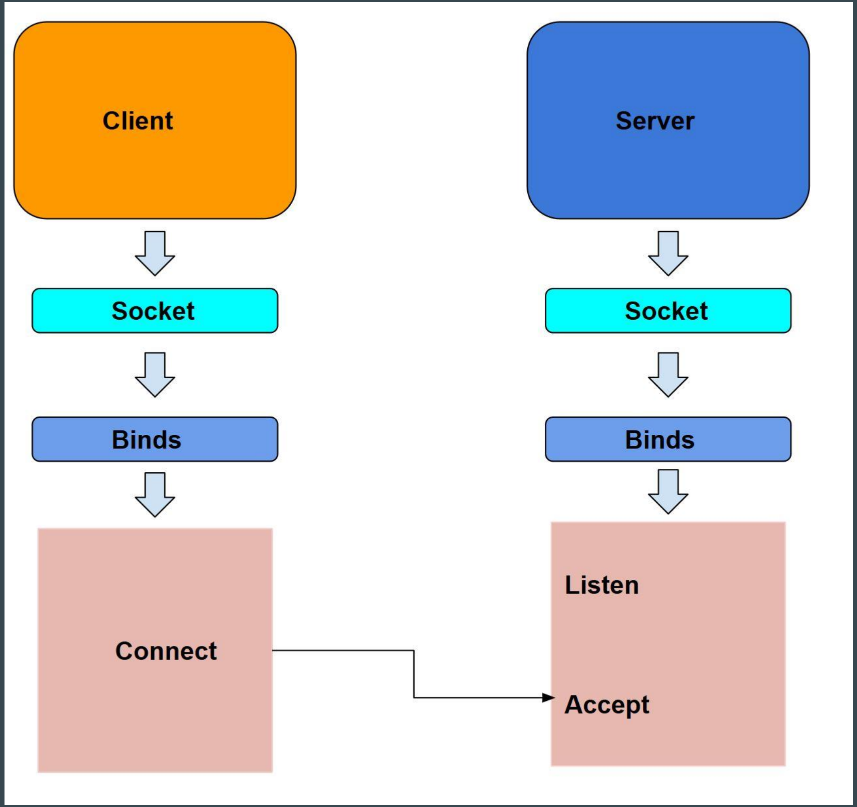
Protocol 5



# Flow diagram



# C Socket

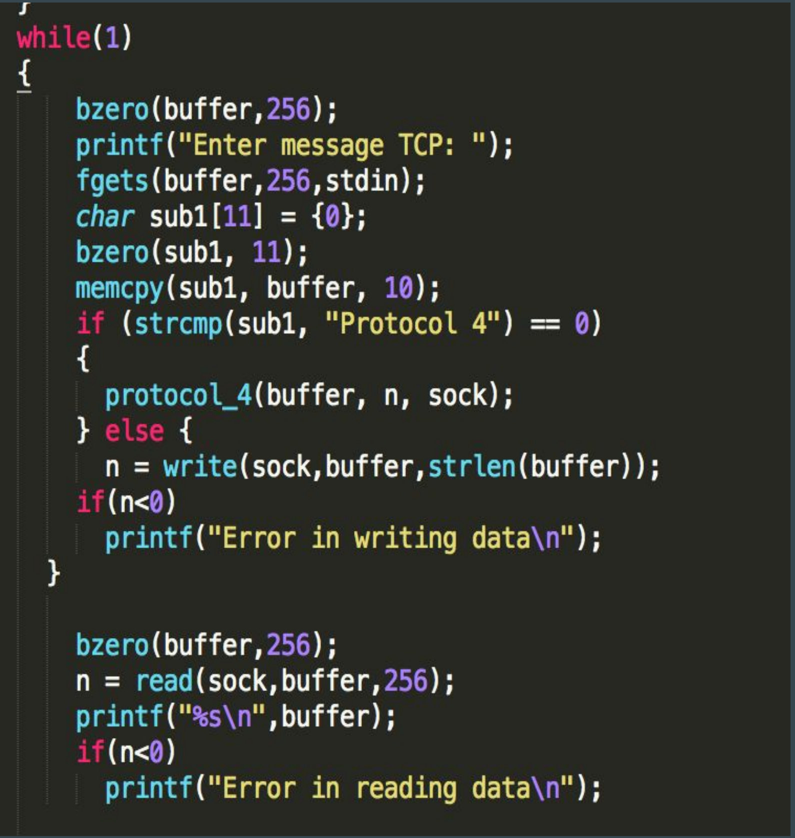


```
struct sockaddr_in server_addr,client_addr;
sds = socket(AF_INET,SOCK_STREAM,0) ;
portno = atoi(argv[1]);
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(portno);
server_addr.sin_addr.s_addr = INADDR_ANY;
if(bind(sds,(struct sockaddr
                  *)&server_addr,sizeof(server_addr)) < 0)
 {
   perror("Failed to bind\n");
   exit(1);
 }
```

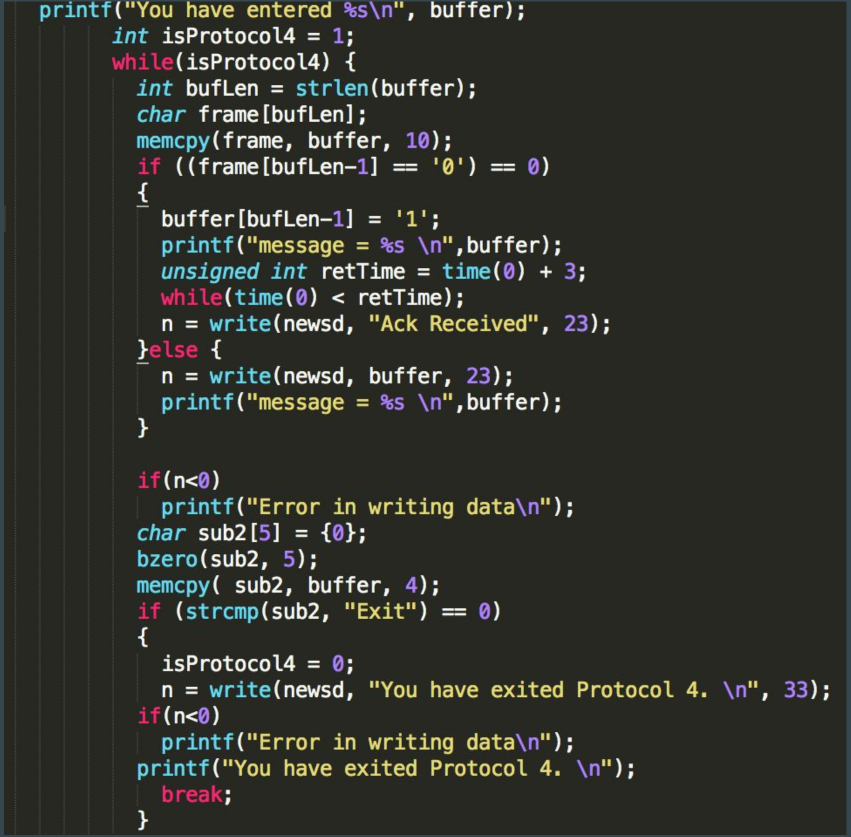
# Main code functions only

- Sockaddr_in
- Socket -Blocking and non-Blocking
- Connect()
- Bind()
- accept()
- listen()

# Code samples



```
}
while(1)
{
    bzero(buffer,256);
    printf("Enter message TCP: ");
    fgets(buffer,256,stdin);
    char sub1[11] = {0};
    bzero(sub1, 11);
    memcpy(sub1, buffer, 10);
    if (strcmp(sub1, "Protocol 4") == 0)
    {
      protocol_4(buffer, n, sock);
    } else {
      n = write(sock,buffer,strlen(buffer));
    if(n<0)
      printf("Error in writing data\n");
  }

    bzero(buffer,256);
    n = read(sock,buffer,256);
    printf("%s\n",buffer);
    if(n<0)
      printf("Error in reading data\n");
```



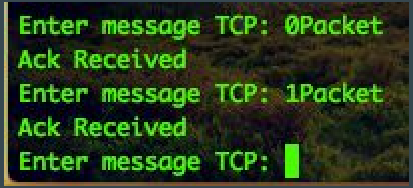
```
printf("You have entered %s\n", buffer);
   int isProtocol4 = 1;
   while(isProtocol4) {
    int bufLen = strlen(buffer);
    char frame[bufLen];
    memcpy(frame, buffer, 10);
    if ((frame[bufLen-1] == '0') == 0)
    {
     buffer[bufLen-1] = '1';
     printf("message = %s \n",buffer);
     unsigned int retTime = time(0) + 3;
     while(time(0) < retTime);
     n = write(newsd, "Ack Received", 23);
    }else {
     n = write(newsd, buffer, 23);
     printf("message = %s \n",buffer);
    }

    if(n<0)
     printf("Error in writing data\n");
    char sub2[5] = {0};
    bzero(sub2, 5);
    memcpy( sub2, buffer, 4);
    if (strcmp(sub2, "Exit") == 0)
    {
     isProtocol4 = 0;
     n = write(newsd, "You have exited Protocol 4. \n", 33);
    if(n<0)
     printf("Error in writing data\n");
    printf("You have exited Protocol 4. \n");
     break;
    }
```

# More Code Samples

```
n = read(newsd,buffer,255);
if(n<0)
        printf("Error in reading data\n");

n = write(newsd, "Ack\n", 23);
          printf("message = %s\n",buffer);

memcpy( sub1, buffer, 10);
      if(strcmp(sub1, "Protocol 4") == 0){
           …….
```

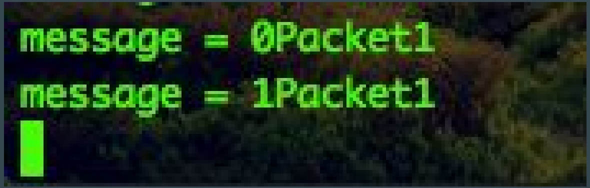
# Terminals

Client: Protocol 4



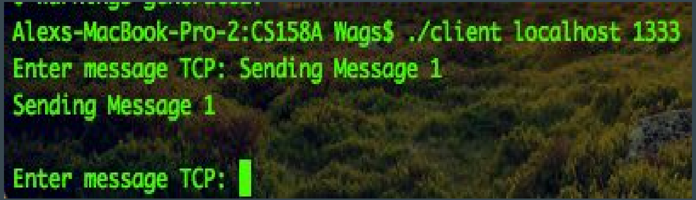
```
Enter message TCP: 0Packet
Ack Received
Enter message TCP: 1Packet
Ack Received
Enter message TCP: 
```

Server: Protocol 4



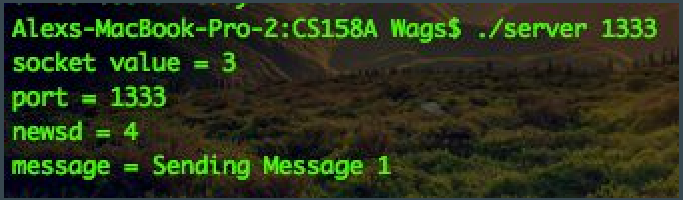
```
message = 0Packet1
message = 1Packet1
```

Client:



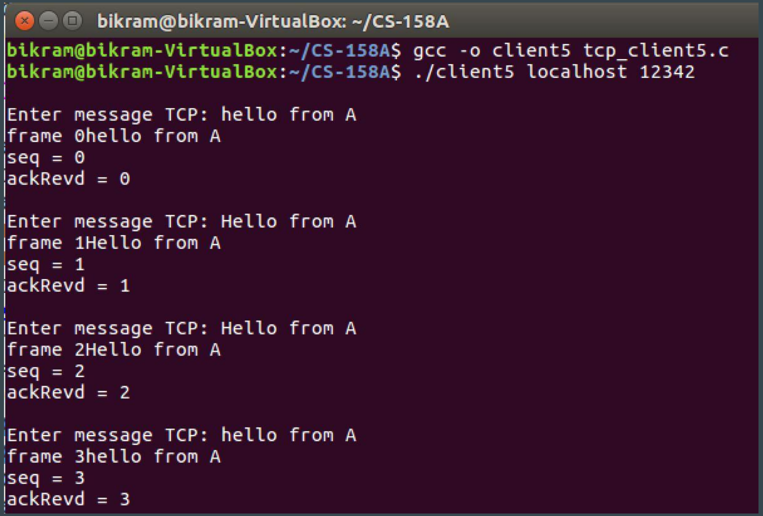
```
Alexs-MacBook-Pro-2:CS158A Wags$ ./client localhost 1333
Enter message TCP: Sending Message 1
Sending Message 1

Enter message TCP: 
```

Server:

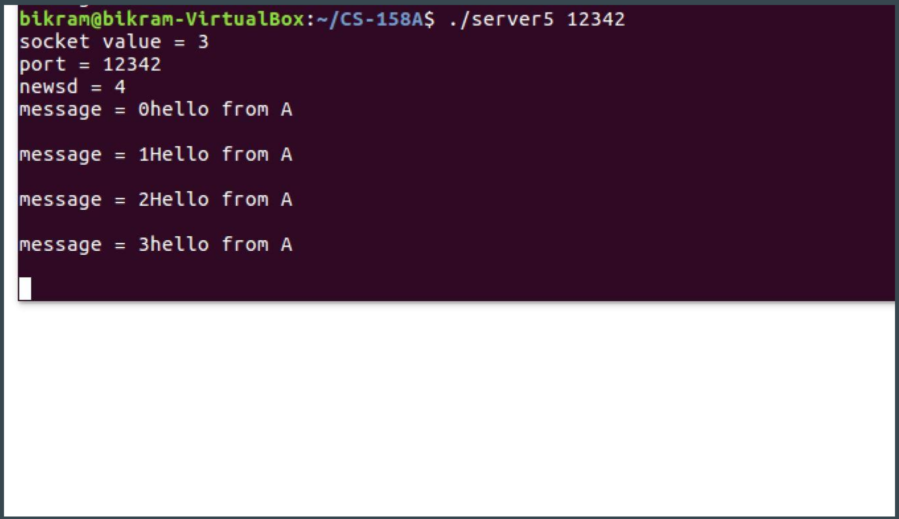


```
Alexs-MacBook-Pro-2:CS158A Wags$ ./server 1333
socket value = 3
port = 1333
newsd = 4
message = Sending Message 1
```

# 2 Terminals



```
bikram@bikram-VirtualBox:~/CS-158A$ gcc -o client5 tcp_client5.c
bikram@bikram-VirtualBox:~/CS-158A$ ./client5 localhost 12342

Enter message TCP: hello from A
frame 0hello from A
seq = 0
ackRevd = 0

Enter message TCP: Hello from A
frame 1Hello from A
seq = 1
ackRevd = 1

Enter message TCP: Hello from A
frame 2Hello from A
seq = 2
ackRevd = 2

Enter message TCP: hello from A
frame 3hello from A
seq = 3
ackRevd = 3
```



```
bikram@bikram-VirtualBox:~/CS-158A$ ./server5 12342
socket value = 3
port = 12342
newsd = 4
message = 0hello from A

message = 1Hello from A

message = 2Hello from A

message = 3hello from A

```

# Conclusion:

- This project helped put what was learned in the course into practice
- Learning about Protocol 4 and Protocol 5 in the course and then simulating them in C was challenging
- Learning how to create sockets in C was interesting

# Future Work

- GUI
- Educational App
- Further Implementation of Protocols

# Questions