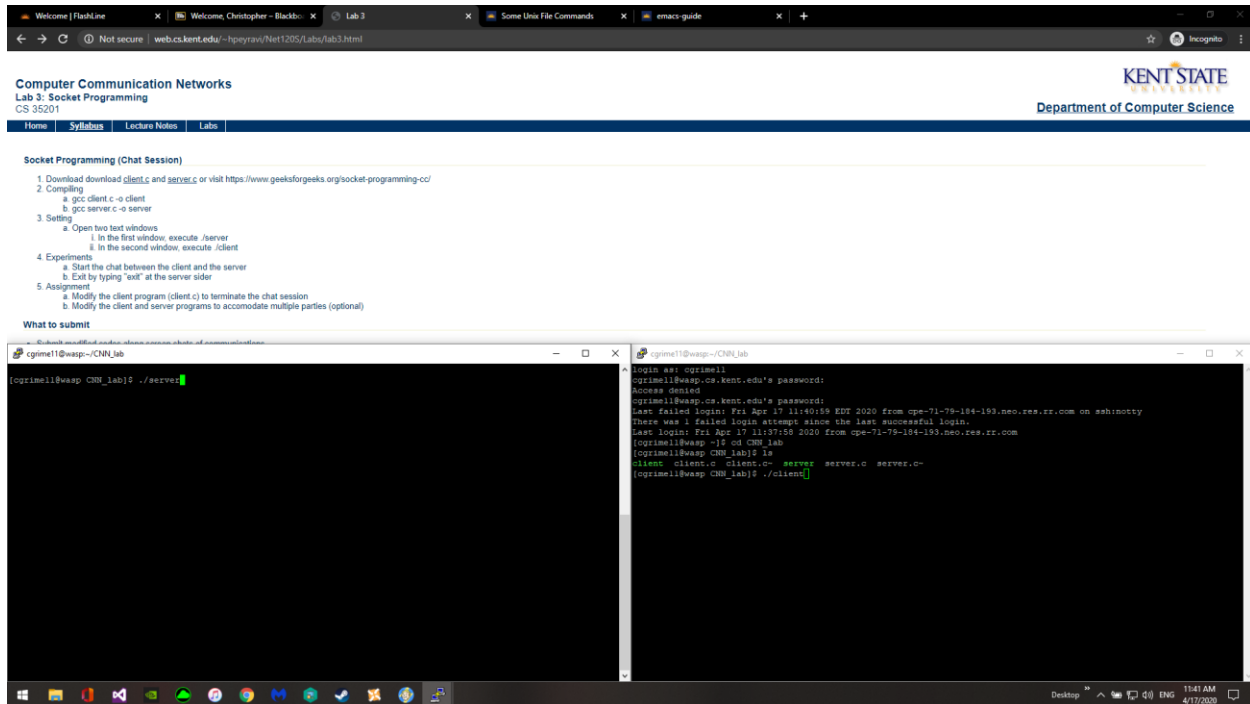


# Chris Grimes

## CS 35201

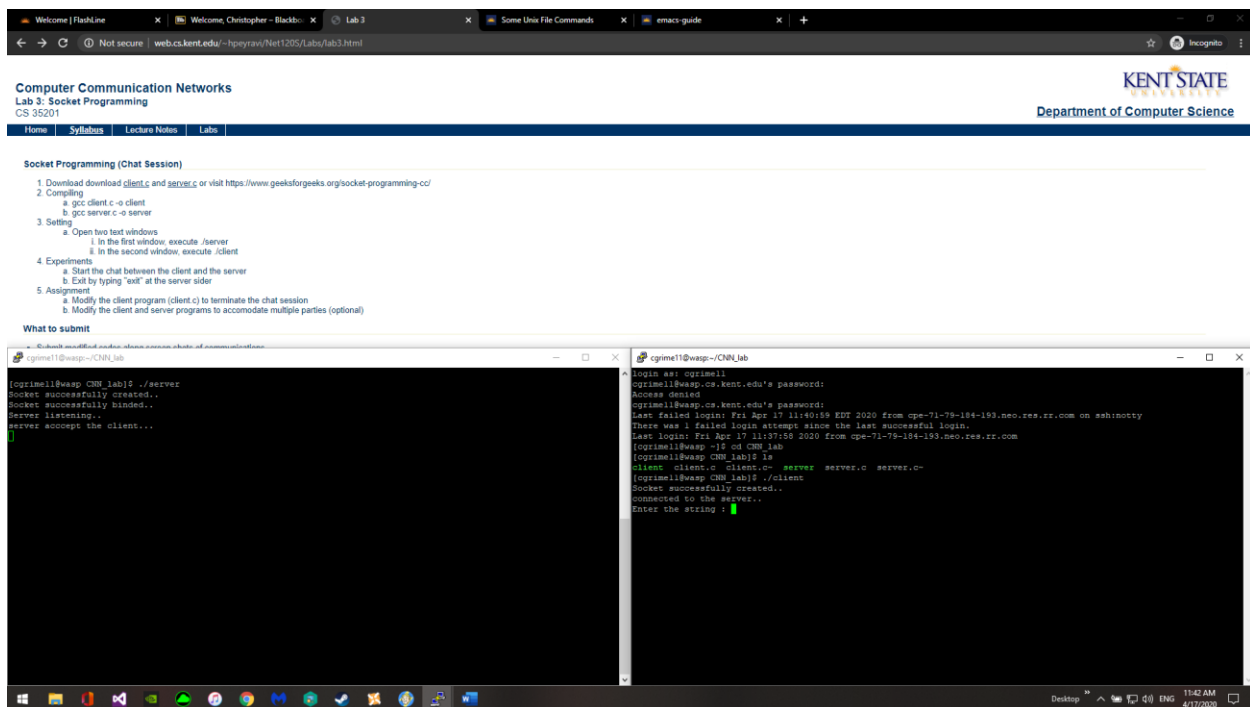
### Lab 3

### Chatting:



The screenshot shows a web browser window with the URL `web.cs.kent.edu/~hpeyavi/Net1205/Labs/lab3.html`. The page title is "Computer Communication Networks" and the subtitle is "Lab 3: Socket Programming CS 35201". The Kent State University logo and "Department of Computer Science" are visible in the top right. The page content includes a "Socket Programming (Chat Session)" section with instructions for downloading, compiling, and running a chat program. Below the instructions, there are two terminal windows. The left terminal window shows the user `cgimel1@wasp:~/CNP_Lab` running `./server`. The right terminal window shows the user `cgimel1@wasp:~/CNP_Lab` logging in as `cgimel1`, running `./client`, and then `./server`. The terminal output shows a successful connection and the exchange of messages.

```
login as: cgimel1
cgimel1@wasp.cs.kent.edu's password:
Access denied
cgimel1@wasp.cs.kent.edu's password:
Last failed login: Fri Apr 17 11:40:19 EDT 2020 from ope-71-79-184-193.neo.res.rr.com on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Fri Apr 17 11:37:58 2020 from ope-71-79-184-193.neo.res.rr.com
[cgimel1@wasp ~]$ cd CNP_Lab
[cgimel1@wasp CNP_Lab]$ ls
client client.c client.o server server.c server.o
[cgimel1@wasp CNP_Lab]$ ./client
```



The screenshot shows a web browser window with the URL `web.cs.kent.edu/~hpeyavi/Net1205/Labs/lab3.html`. The page title is "Computer Communication Networks" and the subtitle is "Lab 3: Socket Programming CS 35201". The Kent State University logo and "Department of Computer Science" are visible in the top right. The page content includes a "Socket Programming (Chat Session)" section with instructions for downloading, compiling, and running a chat program. Below the instructions, there are two terminal windows. The left terminal window shows the user `cgimel1@wasp:~/CNP_Lab` running `./server`. The right terminal window shows the user `cgimel1@wasp:~/CNP_Lab` logging in as `cgimel1`, running `./client`, and then `./server`. The terminal output shows a successful connection and the exchange of messages.

```
login as: cgimel1
cgimel1@wasp.cs.kent.edu's password:
Access denied
cgimel1@wasp.cs.kent.edu's password:
Last failed login: Fri Apr 17 11:40:19 EDT 2020 from ope-71-79-184-193.neo.res.rr.com on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Fri Apr 17 11:37:58 2020 from ope-71-79-184-193.neo.res.rr.com
[cgimel1@wasp ~]$ cd CNP_Lab
[cgimel1@wasp CNP_Lab]$ ls
client client.c client.o server server.c server.o
[cgimel1@wasp CNP_Lab]$ ./client
```

# Chris Grimes

## CS 35201

### Lab 3

Computer Communication Networks  
Lab 3: Socket Programming  
CS 35201

Home Syllabus Lecture Notes Labs

Socket Programming (Chat Session)  
1. Download download client.c and server.c or visit <https://www.geeksforgeeks.org/socket-programming-cc/>  
2. Compiling  
a. gcc client.c -o client  
b. gcc server.c -o server  
3. Setting  
a. Open two text windows  
i. In the first window, execute ./server  
ii. In the second window, execute ./client  
4. Experiments  
a. Start the chat between the client and the server  
b. Exit by typing "exit" at the server side  
5. Assignment  
a. Modify the client program (client.c) to terminate the chat session  
b. Modify the client and server programs to accommodate multiple parties (optional)  
  
What to submit

cgimel1@waspi:~/CNR\_Lab

```
[cgimel1@waspi CNR_Lab]$ ./server
Socket successfully created...
Server listening...
server accept the client...
From client: Hello
To client : Hello
From client: Bye
To client : exit
```

cgimel1@waspi:~/CNR\_Lab

```
login as: cgimel1
cgimel1@waspi.cs.kent.edu's password:
Access denied
cgimel1@waspi.cs.kent.edu's password:
Last failed login: Fri Apr 17 11:40:19 EDT 2020 from cpe-71-79-184-193.neo.res.rr.com on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Fri Apr 17 11:37:56 2020 from cpe-71-79-184-193.neo.res.rr.com
[cgimel1@waspi ~]$ cd CNR_Lab
[cgimel1@waspi CNR_Lab]$ ls
client client.o client.c server server.c
[cgimel1@waspi CNR_Lab]$ ./client
Socket successfully created...
connected to the server..
Enter the string : Hello
From Server : Hello
Enter the string : Bye
exit
```

Computer Communication Networks  
Lab 3: Socket Programming  
CS 35201

Home Syllabus Lecture Notes Labs

Socket Programming (Chat Session)  
1. Download download client.c and server.c or visit <https://www.geeksforgeeks.org/socket-programming-cc/>  
2. Compiling  
a. gcc client.c -o client  
b. gcc server.c -o server  
3. Setting  
a. Open two text windows  
i. In the first window, execute ./server  
ii. In the second window, execute ./client  
4. Experiments  
a. Start the chat between the client and the server  
b. Exit by typing "exit" at the server side  
5. Assignment  
a. Modify the client program (client.c) to terminate the chat session  
b. Modify the client and server programs to accommodate multiple parties (optional)  
  
What to submit

cgimel1@waspi:~/CNR\_Lab

```
[cgimel1@waspi CNR_Lab]$ ./server
Socket successfully created...
Server listening...
server accept the client...
From client: Hello
To client : Hello
From client: Bye
To client : exit
Server Exit...
```

cgimel1@waspi:~/CNR\_Lab

```
login as: cgimel1
cgimel1@waspi.cs.kent.edu's password:
Access denied
cgimel1@waspi.cs.kent.edu's password:
Last failed login: Fri Apr 17 11:40:19 EDT 2020 from cpe-71-79-184-193.neo.res.rr.com on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Fri Apr 17 11:37:56 2020 from cpe-71-79-184-193.neo.res.rr.com
[cgimel1@waspi ~]$ cd CNR_Lab
[cgimel1@waspi CNR_Lab]$ ls
client client.o client.c server server.c
[cgimel1@waspi CNR_Lab]$ ./client
Socket successfully created...
connected to the server..
Enter the string : Hello
From Server : Hello
Enter the string : Bye
exitFrom Server : exit
Client Exit...
[cgimel1@waspi CNR_Lab]$ exit
```

Modified code as to allow client to end session:

# Chris Grimes

## CS 35201

### Lab 3

```
cgimell@wasp:~/CNU_Lab$
[cgimell@wasp CNU_Lab]$ gcc server.c -o server
[cgimell@wasp CNU_Lab]$ ./server
Socket successfully created..
Socket bind failed...
[cgimell@wasp CNU_Lab]$ ./server
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client...
From client : exit
To client : exit
Server Exit...
[cgimell@wasp CNU_Lab]$ emacs client.c
[cgimell@wasp CNU_Lab]$ gcc server.c -o server
[cgimell@wasp CNU_Lab]$ ./server
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client...
From client: hi
To client : hi
From client : exit
To client : exit
Server Exit...
[cgimell@wasp CNU_Lab]$ emacs server.c
[cgimell@wasp CNU_Lab]$ gcc server.c -o server
[cgimell@wasp CNU_Lab]$ ./server
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client...
From client: hi
To client : hi
From client : exit
To client : exit
Server Exit...
[cgimell@wasp CNU_Lab]$ emacs client.c
[cgimell@wasp CNU_Lab]$ gcc client.c -o client
[cgimell@wasp CNU_Lab]$ ./client
Socket successfully created..
connection with the server failed...
[cgimell@wasp CNU_Lab]$ emacs client.c
[cgimell@wasp CNU_Lab]$ gcc client.c -o client
[cgimell@wasp CNU_Lab]$ ./client
Socket successfully created..
connected to the server..
Enter the string : hi
From Server : hi
From Server : exit
Client Exit...
[cgimell@wasp CNU_Lab]$ emacs client.c
[cgimell@wasp CNU_Lab]$ gcc client.c -o client
[cgimell@wasp CNU_Lab]$ ./client
Socket successfully created..
connected to the server..
Enter the string : hi
From Server : hi
From Server : exit
Client Exit...
[cgimell@wasp CNU_Lab]$ emacs client.c
[cgimell@wasp CNU_Lab]$ gcc client.c -o client
[cgimell@wasp CNU_Lab]$ ./client
Socket successfully created..
connected to the server..
Enter the string : hi
From Server : hi
From Server : exit
Client Exit...
[cgimell@wasp CNU_Lab]$
```

client modified code:

```
cgimell@wasp:~/CNU_Lab$
File Edit Options Buffers Tools C Help
// Write C++ code here
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
void func(int sockfd)
{
    char buff[MAX];
    int n;
    // infinite loop for chat
    for (;;) {
        bzero(buff, sizeof(buff));
        // write message
        printf("Enter the string : ");
        n = 0;
        while ((buff[n++] = getchar()) != '\n')
            ;
        // send to server
        write(sockfd, buff, sizeof(buff));
        if (strncmp("exit", buff, 4) == 0) {
            printf("Client Exit...\n");
            break;
        }
        bzero(buff, sizeof(buff));
        // read message
        read(sockfd, buff, sizeof(buff));
        // print message
        printf("From Server : %s", buff);
        // if (strncmp(buff, "exit", 4) == 0) {
        //     printf("Client Exit...\n");
        //     break;
        // }
        if (strncmp("exit", buff, 4) == 0) {
            printf("Client Exit...\n");
            break;
        }
    }
}

int main()
{
    int sockfd, connfd;
    struct sockaddr_in servaddr, cli;

    // socket create and verification
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        perror("socket creation failed");
        return 1;
    }
    // set port and ip
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(PORT);
    inet_aton("127.0.0.1", &servaddr.sin_addr);
    bzero(servaddr.sin_zero, 8);

    if (connect(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0)
        perror("connect failed");
    func(sockfd);
    close(sockfd);
    return 0;
}

Building cc-lab3.c.o
Building cc-lab3.o
Building cc-lab3
```

# Chris Grimes

## CS 35201

### Lab 3

```
cgimel1@wasp:~/CNR_Lab
File Edit Options Buffers Tools C Help
sockfd = socket(AF_INET, SOCK_STREAM, 0);
if (sockfd == -1) {
    printf("socket creation failed...\n");
    exit(0);
}
else
    printf("Socket successfully created...\n");
memset(&servaddr, sizeof(servaddr));

// assign IP, PORT
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
servaddr.sin_port = htons(PORT);

// connect the client socket to server socket
if (connect(sockfd, (&servaddr), sizeof(servaddr)) != 0) {
    printf("connection with the server failed...\n");
    exit(0);
}
else
    printf("connected to the server...\n");

// function for chat
func(sockfd);

// close the socket
close(sockfd);
}

UDP-1-----> client:~  Bot 118 (C/I Abbrev)
Desktop 12:42 PM 4/17/2020
```

server modified code:

```
cgimel1@wasp:~/CNR_Lab
File Edit Options Buffers Tools C Help
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr

// Function designed for chat between client and server.
void func(int sockfd)
{
    char buff[MAX];
    int n;
    // infinite loop for chat
    for (;;) {
        memset(buff, MAX);

        // receive message from client and copy it in buffer
        read(sockfd, buff, sizeof(buff));

        // print buffer which contains the client contents
        printf("From client: %s\n", buff);

        if(strcmp("exit", buff, 4) == 0) {
            printf("Server Exit...\n");
            break;
        }

        memset(buff, MAX);
        n = 0;
        // copy server message in the buffer
        while ((buff[n++] = getchar()) != '\n')
            ;

        // and send that buffer to client
        write(sockfd, buff, sizeof(buff));

        // if may contains "Exit" then server exit and chat ended.
        if (strcmp("exit", buff, 4) == 0) {
            printf("Server Exit...\n");
            break;
        }
    }
}

// Driver function
int main()
{
    int sockfd, connfd, len;
    struct sockaddr_in servaddr, cli;

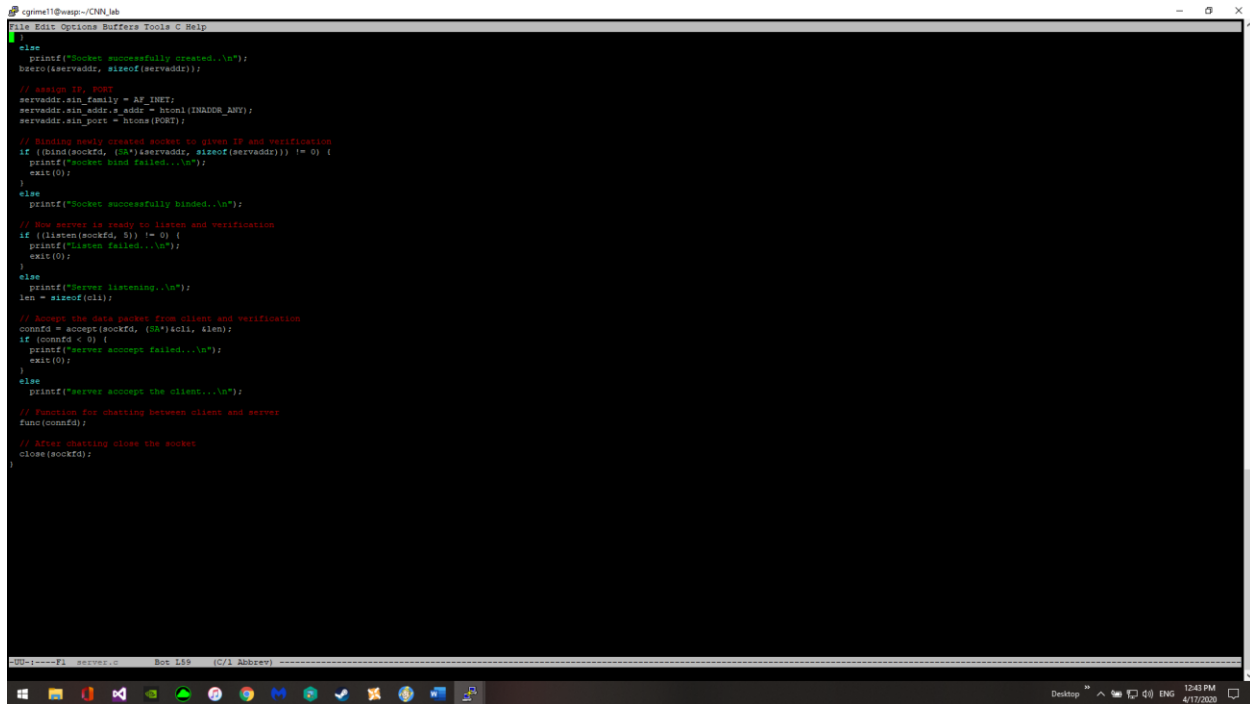
    // socket creation and verification
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created...\n");
}

UDP-1-----> server:~  Top 11 (C/I Abbrev)
Loading cc-lan2v, done
Desktop 12:43 PM 4/17/2020
```

Chris Grimes

CS 35201

Lab 3



```
1
2
3 // Socket successfully created.\n");
4 // Server address, sizeof(servaddr);
5
6 // Assign IP port
7 servaddr.sin_family = AF_INET;
8 servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
9 servaddr.sin_port = htons(PORT);
10
11 // Binding newly created socket to given IP and verification
12 if (bind(sockfd, (struct sockaddr*)&servaddr, sizeof(servaddr)) != 0) {
13     printf("socket bind failed...\n");
14     exit(0);
15 }
16
17 // Socket successfully binded.\n");
18
19 // Now server is ready to listen and verification
20 if (listen(sockfd, 5)) != 0 {
21     printf("listen failed...\n");
22     exit(0);
23 }
24
25 // Server listening.\n");
26 len = sizeof(cli);
27
28 // Accept the data packet from client and verification
29 sockfd = accept(sockfd, (struct sockaddr*)&cli, &len);
30 if (sockfd < 0) {
31     printf("server accept failed...\n");
32     exit(0);
33 }
34
35 // Server accept the client.\n");
36
37 // Function for chatting between client and server
38 func(chatfd);
39
40 // After chatting close the socket
41 close(sockfd);
42 }
```

Windows taskbar at the bottom shows the Start button, taskbar icons, and system tray with the date and time: 12:43 PM 4/17/2020.