Neeraj Varma 103210651
OS
A1

__/__/____

CN Lab Assignment

Theory:

(i) Client / Server Communication: The client does not form a connection with the server like in TCP & instead just sends a datagram. Similarly, the server need not accept a connection & just waits for datagrams to arrive.

(ii) Introduction to UDP: It is a communications protocol that is primarily used to establish low-latency & loss-tolerating connections between applications on the internet

(iii) UDP Segment Header: It wraps datagram with a UDP header, which consists of fields totalling eight bytes. The fields are: Source Port- This field can be set to zero if the destination computer doesn't need to reply to the sender

(iv) Introduction to sockets: Sockets are commonly used for client & server interaction. Typical system configuration places the server, exchange information & then disconnect.

(v) UDP socket functions: The server & client both creates a socket. The server uses the bind call to associate a local address to the socket. The client can issue an optional bind call to a local address

__/__/____

(vi) UDP file description on server:

1. Socket()
2. bind()
3. Recieve from()
4. Exit

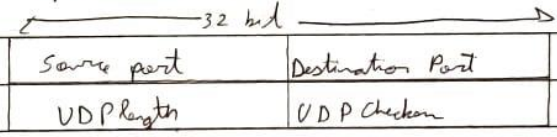
(vii) UDP Socket flow diagram on client:

1. Socket()
2. Send to()
3. Receive from()
4. Close()

FAQ's

1. Draw & explain UDP header

→



1. Source port: Port of sender
2. Dest. port: Port of receiver
3. Length: Length of UDP
4. Checksum: Uses checksum for error detection

__/__/____

2. Differentiate between TCP and UDP

→

TCP	UDP
• Keeps track of lost packets	• Doesn't keep track of lost packets
• Slower, because of added functions	• Faster, because it lacks other features
• Is connection-oriented	• Is connection-less
• Examples: - HTTP - HTTPIS - FTP	• Examples: - DNS - IP telephony - DHCP

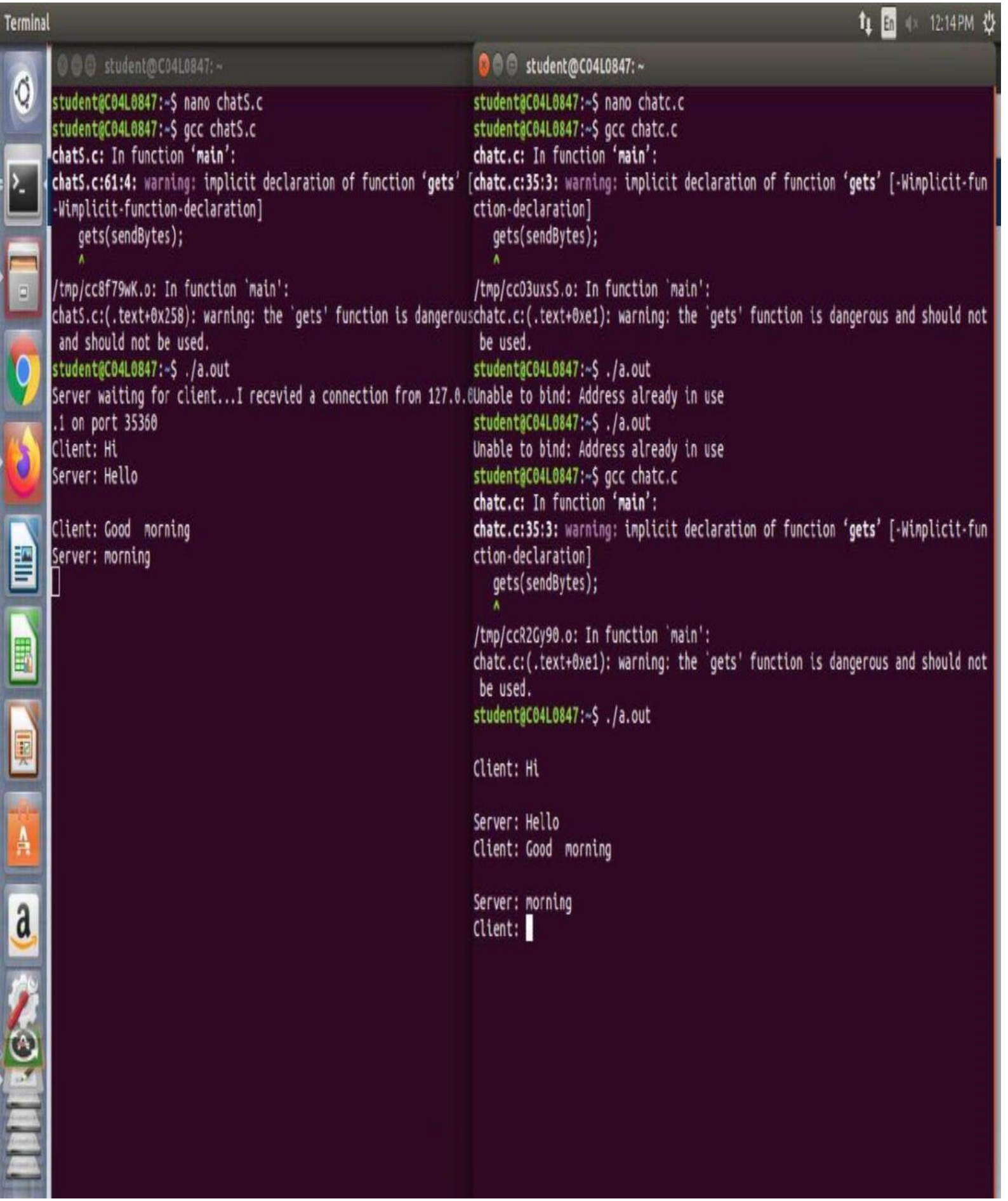
```c
// server program for udp connection
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include<netinet/in.h>
#define PORT 5000
#define MAXLINE 1000

// Driver code
int main()
{
    char buffer[100];
    char *message = "Hello Client";
    int listenfd, len;
    struct sockaddr_in servaddr, cliaddr;
    bzero(&servaddr, sizeof(servaddr));

    // Create a UDP Socket
    listenfd = socket(AF_INET, SOCK_DGRAM, 0);
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);
    servaddr.sin_family = AF_INET;

    // bind server address to socket descriptor
    bind(listenfd, (struct sockaddr*)&servaddr, sizeof(servaddr));

```

```
18      bzero(&servaddr, sizeof(servaddr));
19
20      // Create a UDP Socket
21      listenfd = socket(AF_INET, SOCK_DGRAM, 0);
22      servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
23      servaddr.sin_port = htons(PORT);
24      servaddr.sin_family = AF_INET;
25
26      // bind server address to socket descriptor
27      bind(listenfd, (struct sockaddr*)&servaddr, sizeof(servaddr));
28
29      //receive the datagram
30      len = sizeof(cliaddr);
31      int n = recvfrom(listenfd, buffer, sizeof(buffer),
32              0, (struct sockaddr*)&cliaddr,&len); //receive message from server
33      buffer[n] = '\0';
34      puts(buffer);
35      int i;
36      for(i=0;i<n;i++){
37              if(buffer[i]>=65&&buffer[i]<=90)
38                  buffer[i]=buffer[i]+32;
39          }
40      puts(buffer);
41
42      // send the response
43      sendto(listenfd, buffer, MAXLINE, 0,
44          (struct sockaddr*)&cliaddr, sizeof(cliaddr));
45  }
```

```c
// udp client driver program
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include<netinet/in.h>
#include<unistd.h>
#include<stdlib.h>

#define PORT 5000
#define MAXLINE 1000

// Driver code
int main()
{
    char buffer[100];
    char *message = "HELLO SERVER";
    int sockfd, n;
    struct sockaddr_in servaddr;
    //printf("Hello World");
    // clear servaddr
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = htons(PORT);
    servaddr.sin_family = AF_INET;

    // create datagram socket
```

create data ram socket

```
21      //printf("Hello World");
22      // clear servaddr
23      bzero(&servaddr, sizeof(servaddr));
24      servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
25      servaddr.sin_port = htons(PORT);
26      servaddr.sin_family = AF_INET;
27
28      // create datagram socket
29      sockfd = socket(AF_INET, SOCK_DGRAM, 0);
30
31      // connect to server
32      if(connect(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0)
33      {
34          printf("\n Error : Connect Failed \n");
35          exit(0);
36      }
37
38      // request to send datagram
39      // no need to specify server address in sendto
40      // connect stores the peers IP and port
41      sendto(sockfd, message, MAXLINE, 0, (struct sockaddr*)NULL, sizeof(servaddr));
42
43      // waiting for response
44      recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr*)NULL, NULL);
45      puts(buffer);
46
47      // close the descriptor
48      close(sockfd);
```


Terminal
12:14 PM
student@C04L0847: ~
student@C04L0847:~$ nano chatS.c
student@C04L0847:~$ gcc chatS.c
chatS.c: In function 'main':
chatS.c:61:4: warning: implicit declaration of function 'gets' [-Wimplicit-function-declaration]
gets(sendBytes);
^
/tmp/cc8f79wK.o: In function 'main':
chatS.c:(.text+0x258): warning: the 'gets' function is dangerous and should not be used.
student@C04L0847:~$ ./a.out
Server waiting for client...I recevied a connection from 127.0.0.1 on port 35360
Client: Hi
Server: Hello
Client: Good morning
Server: morning
student@C04L0847:~
student@C04L0847:~$ nano chatc.c
student@C04L0847:~$ gcc chatc.c
chatc.c: In function 'main':
chatc.c:35:3: warning: implicit declaration of function 'gets' [-Wimplicit-function-declaration]
gets(sendBytes);
^
/tmp/ccO3uxsS.o: In function 'main':
chatc.c:(.text+0xe1): warning: the 'gets' function is dangerous and should not be used.
student@C04L0847:~$ ./a.out
Unable to bind: Address already in use
student@C04L0847:~$ ./a.out
Unable to bind: Address already in use
student@C04L0847:~$ gcc chatc.c
chatc.c: In function 'main':
chatc.c:35:3: warning: implicit declaration of function 'gets' [-Wimplicit-function-declaration]
gets(sendBytes);
^
/tmp/ccR2Gy90.o: In function 'main':
chatc.c:(.text+0xe1): warning: the 'gets' function is dangerous and should not be used.
student@C04L0847:~$ ./a.out
Client: Hi
Server: Hello
Client: Good morning
Server: morning
Client: