CENG393 Computer Networks
Labwork 7

1 Socket Programming: UDP Sockets

1.1 Definition

UDP is an another transport layer protocol, which has less overhead from TCP. Remember that TCP requires establishment of a connection first and uses acknowledgement messages and sequence numbers to guarentee that messages are delivered correctly to the other end of the socket. In contrary to TCP, UDP utilizes none of these methods: it just sends messages without acknowledging whether the message has been received by the other end of the socket or not. It is a faster way of transmitting data across the network but it is more vulnerable to message loss and corruption (therefore correction must be handled by upper-layer application protocols).

The process of establishing and utilizing a connectionless UDP socket is demonstrated in Figure 1:

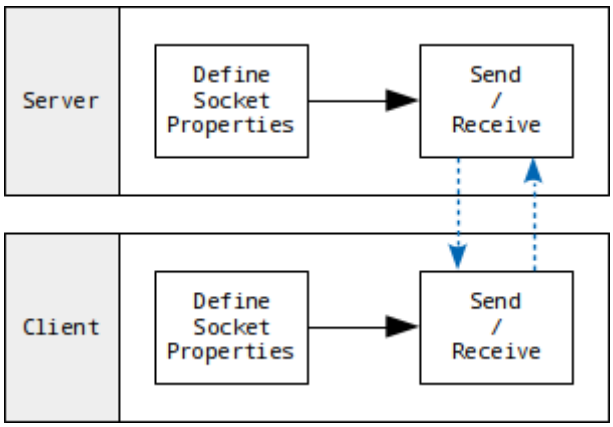

Figure 1: UDP Sockets

2 Exercise

1. Study manpages of **sendto** and **recvfrom** system calls. What is the difference of **send** from **sendto** and **recv** from **recvfrom**?
2. Study the programs given below. Modify them such that server keeps a list of blacklisted IP addresses and if a blacklisted client sends a message to the server, server must send back "MESSAGE REJECTED" as reply to the unwanted client.

2.1 udpserver.c

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <stdio.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <stdlib.h>
int main() {
  int sock;
  int addr_len, bytes_read;
  char recv_data[1024];
  struct sockaddr_in server_addr, client_addr;
  if ((sock = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
    perror("Socket");
    exit(1);
  }
  server_addr.sin_family = AF_INET;
  server_addr.sin_port = htons(5000);
  server_addr.sin_addr.s_addr = INADDR_ANY;
  bzero(&(server_addr.sin_zero), 8);
  if (bind(sock, (struct sockaddr *) &server_addr, sizeof(struct sockaddr))
      == -1) {
    perror("Bind");
    exit(1);
  }
  addr_len = sizeof(struct sockaddr);
  printf("\nUDPServer Waiting for client on port 5000");
  fflush(stdout);
  while (1) {
    bytes_read = recvfrom(sock, recv_data, 1024, 0,
        (struct sockaddr *) &client_addr, &addr_len);
    recv_data[bytes_read] = '\0';
    printf("\n(%s , %d) said : ", inet_ntoa(client_addr.sin_addr),
        ntohs(client_addr.sin_port));
    printf("%s", recv_data);
    fflush(stdout);
  }
  return 0;
}
```

2.2 udpclient.c

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <stdio.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
int main() {
  int sock;
  struct sockaddr_in server_addr;
  struct hostent *host;
  char send_data[1024];
  host = (struct hostent *) gethostbyname((char *) "127.0.0.1");
  if ((sock = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
    perror("socket");
    exit(1);
  }
  server_addr.sin_family = AF_INET;
  server_addr.sin_port = htons(5000);
  server_addr.sin_addr = *((struct in_addr *) host->h_addr);
  bzero(&(server_addr.sin_zero), 8);
  while (1) {
    printf("Type Something (q or Q to quit):");
    gets(send_data);
    if ((strcmp(send_data, "q") == 0) || strcmp(send_data, "Q") == 0)
      break;
    else
      sendto(sock, send_data, strlen(send_data), 0,
          (struct sockaddr *) &server_addr, sizeof(struct sockaddr));
  }
}
```