Network Lab - Assignment

Socket Programming - UDP Sockets

Usirikayala Likhith 20223295

1. Echo Client and Server using UDP Sockets

The goal is to implement an Echo Client and an Echo Server in C using UDP Datagram Sockets. The client sends a string to the server, and the server echoes the same string back to the client.

Echo Client Code (C)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <arpa/inet.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
   if (argc != 3) {
      printf("Usage: %s <Server IP> <Message>
", argv[0]);
      return 1;
   }

   int sockfd;
   struct sockaddr_in servaddr;
   char buffer[1024];

   if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
      perror("Socket creation failed");
      return 1;
   }

   memset(&servaddr, 0, sizeof(servaddr));
   servaddr.sin_family = AF_INET;
   servaddr.sin_port = htons(8080);
   servaddr.sin_addr.s_addr = inet_addr(argv[1]);

   sendto(sockfd, argv[2], strlen(argv[2]), 0, (struct sockaddr*)&servaddr,
sizeof(servaddr));
```

```
    int n = recvfrom(sockfd, buffer, sizeof(buffer), 0, NULL, NULL);
    buffer[n] = '\0';
    printf("Received from server: %s\n", buffer);

    close(sockfd);
    return 0;
}
```

Echo Server Code (C)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <arpa/inet.h>
#include <unistd.h>

int main() {
    int sockfd;
    struct sockaddr_in servaddr, cliaddr;
    char buffer[1024];

    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        return 1;
    }

    memset(&servaddr, 0, sizeof(servaddr));
    memset(&cliaddr, 0, sizeof(cliaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = INADDR_ANY;
    servaddr.sin_port = htons(8080);

    if (bind(sockfd, (const struct sockaddr *)&servaddr, sizeof(servaddr)) < 0) {
        perror("Bind failed");
        return 1;
    }

    int len = sizeof(cliaddr);
    int n = recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr*)&cliaddr, &len);
    buffer[n] = '\0';
    printf("Client : %s\n", buffer);
```

```
    sendto(sockfd, buffer, strlen(buffer), 0, (const struct sockaddr*)&cliaddr, len);
    printf("Echo message sent.\n");

    close(sockfd);
    return 0;
}
```

2. Time of Day Client and Server using UDP Sockets

This task requires creating a client-server application where the client requests the current time, and the server responds with the time of day.

Time of Day Client Code (C)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <arpa/inet.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
    if (argc != 2) {
        printf("Usage: %s <Server IP>\n", argv[0]);
        return 1;
    }

    int sockfd;
    struct sockaddr_in servaddr;
    char buffer[1024];

    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        return 1;
    }

    memset(&servaddr, 0, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(8080);
    servaddr.sin_addr.s_addr = inet_addr(argv[1]);

    sendto(sockfd, "TIME", 4, 0, (struct sockaddr*)&servaddr, sizeof(servaddr));
    int n = recvfrom(sockfd, buffer, sizeof(buffer), 0, NULL, NULL);
    buffer[n] = '\0';
```

```
    printf("Server time: %s\n", buffer);

    close(sockfd);
    return 0;
}
```

Time of Day Server Code (C)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <time.h>

int main() {
    int sockfd;
    struct sockaddr_in servaddr, cliaddr;
    char buffer[1024];

    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror("Socket creation failed");
        return 1;
    }

    memset(&servaddr, 0, sizeof(servaddr));
    memset(&cliaddr, 0, sizeof(cliaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = INADDR_ANY;
    servaddr.sin_port = htons(8080);

    if (bind(sockfd, (const struct sockaddr *)&servaddr, sizeof(servaddr)) < 0) {
        perror("Bind failed");
        return 1;
    }

    int len = sizeof(cliaddr);
    int n = recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr*)&cliaddr, &len);
    buffer[n] = '\0';
    printf("Client requested: %s\n", buffer);

    if (strcmp(buffer, "TIME") == 0) {
```

```
        time_t now = time(NULL);
        snprintf(buffer, sizeof(buffer), "%s", ctime(&now));
        sendto(sockfd, buffer, strlen(buffer), 0, (const struct sockaddr*)&cliaddr, len);
    }

    close(sockfd);
    return 0;
}
```