**Dharmsinh Desai University, Nadiad**

**Department of Information Technology**

**ECES, IT718**

**B.Tech. IT, Sem: VII**

**Experiment 8**

**Submitted By**

**Name: - Dishant Modh**

**Roll No: - IT076**

**Aim: -** Write a program to authenticate a user with system using MD5 or SHA-1 Hashing technique.

1. **client.c**

#include <stdio.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <unistd.h>

#include "main.h"

#define SERV\_PORT 7069

int main(int argc, char \*\*argv)

{

    int connectSD, noOfBytesRead = 0, choice;

    struct sockaddr\_in servAddr;

    UserData user;

    UserLoginData userLogin;

    char response[100];

    if (argc != 2)

    {

        printf("Usage: %s IP-Address\n", argv[0]);

        return -1;

    }

    if ((connectSD = socket(AF\_INET, SOCK\_STREAM, 0)) < 0)

    {

        printf("Error: Socket creation not allowed.\n");

        return -1;

    }

    bzero(&servAddr, sizeof(servAddr));

    servAddr.sin\_family = AF\_INET;

    servAddr.sin\_port = htons(SERV\_PORT);

    if (inet\_pton(PF\_INET, argv[1], &servAddr.sin\_addr) < 0)

    {

        printf("Error: Socket not bind for server.\n");

        return -1;

    }

    if (connect(connectSD, (struct sockaddr \*)&servAddr, sizeof(servAddr)) < 0)

    {

        printf("Error: Connecting to server.\n");

        return -1;

    }

    while (1)

    {

        printf("\n1. Register Yourself.\n2. Login.\n3. Exit.\nEnter Your Choice: ");

        scanf("%d", &choice);

        if (choice == 1)

        {

            printf("\nEnter Username : ");

            scanf("%s", user.username);

            getchar();

            printf("Enter Password : ");

            scanf("%s", user.password);

            getchar();

            printf("Enter Your Name : ");

            scanf("%[^\n]s", user.name);

            printf("Enter Your Age : ");

            scanf("%d", &user.age);

            write(connectSD, &choice, sizeof(choice));

            write(connectSD, &user, sizeof(user));

            if ((noOfBytesRead = read(connectSD, &response, sizeof(response))) < 0)

                return -1;

            printf("\tServer response: %s.\n", response);

        }

        else if (choice == 2)

        {

            printf("\nEnter Username : ");

            scanf("%s", userLogin.username);

            getchar();

            printf("Enter Password : ");

            scanf("%s", userLogin.password);

            write(connectSD, &choice, sizeof(choice));

            write(connectSD, &userLogin, sizeof(userLogin));

            if ((noOfBytesRead = read(connectSD, &response, sizeof(response))) < 0)

                return -1;

            printf("\tServer response: %s\n", response);

        }

        else if (choice == 3)

            break;

        else

            printf("\t\tEnter Valid choice.\n");

    }

    return 0;

}

1. **server.c**

#include <stdio.h>

#include <sys/socket.h>

#include <unistd.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <string.h>

#include <arpa/inet.h>

#include <openssl/sha.h>

#include "main.h"

#define SERV\_PORT 7069

int listenSD, clientSD, noOfBytesRead = 0, choice;

struct sockaddr\_in servAddr, clientAddr;

UserLoginData userLogin;

UserData user, tmp;

unsigned char hashPassword[SHA\_DIGEST\_LENGTH];

int compare(unsigned char \*s1, unsigned char \*s2)

{

    for (int i = 0; i < SHA\_DIGEST\_LENGTH; i++)

    {

        if (s1[i] != s2[i])

            return 0;

    }

    return 1;

}

void processClient(int clientSD)

{

    while ((noOfBytesRead = read(clientSD, &choice, sizeof(choice))) > 0)

    {

        printf("\nUser selected choice: %d.\n", choice);

        if (choice == 1)

        {

            if ((noOfBytesRead = read(clientSD, &user, sizeof(user))) > 0)

            {

                printf("\nServer recieved following data:\n");

                printf("\tUsername : %s\n\tPassword : %s\n\tName : %s.\n\tAge : %d\n",user.username, user.password, user.name, user.age);

                SHA1(user.password, strlen(user.password), hashPassword);

                printf("\t\tHashed Password: ");

                for (int i = 0; i < SHA\_DIGEST\_LENGTH; i++)

                    printf("%x", hashPassword[i]);

                printf("\n");

                strcpy(user.password, hashPassword);

                FILE \*file = fopen("UserDB.txt", "a+");

                fwrite(&user, sizeof(user), 1, file);

                fclose(file);

                char response[100];

                memset(response, 0, sizeof(response));

                strcpy(response, "Record sucessfully stored");

                write(clientSD, &response, strlen(response));

            }

        }

        else

        {

            if ((noOfBytesRead = read(clientSD, &userLogin, sizeof(userLogin))) > 0)

            {

                printf("\nServer recieved following data:\n");

                printf("\tUsername : %s\n\tPassword : %s\n", userLogin.username, userLogin.password);

                SHA1(userLogin.password, strlen(userLogin.password), hashPassword);

                printf("\t\tHashed Password: ");

                for (int i = 0; i < SHA\_DIGEST\_LENGTH; i++)

                    printf("%x", hashPassword[i]);

                printf("\n");

                FILE \*file = fopen("UserDB.txt", "r");

                int flag = 0;

                char response[100];

                memset(response, 0, sizeof(response));

                while (fread(&tmp, sizeof(tmp), 1, file))

                {

                    if (compare(tmp.password, hashPassword) && !strcmp(tmp.username, userLogin.username))

                    {

                        printf("\t\tRecord Found in DB.\n");

                        sprintf(response, "Welcome %s. Your name: %s. Your Age: %d.", tmp.username, tmp.name, tmp.age);

                        write(clientSD, &response, strlen(response));

                        flag = 1;

                        break;

                    }

                }

                fclose(file);

                if (!flag)

                {

                    strcpy(response, "Either Username or Password not matched.");

                    write(clientSD, &response, strlen(response));

                }

            }

        }

        printf("\nServer have data of User until now: \n");

        FILE \*rfile = fopen("UserDB.txt", "r");

        while (fread(&tmp, sizeof(tmp), 1, rfile))

        {

            printf("\t\tUsername : %s.\tName : %s.\tAge : %d.\n", tmp.username, tmp.name, tmp.age);

        }

        fclose(rfile);

    }

}

int main()

{

    if ((listenSD = socket(AF\_INET, SOCK\_STREAM, 0)) < 0)

    {

        printf("Error: Socket creation not allowed.\n");

        return -1;

    }

    bzero(&servAddr, sizeof(servAddr));

    servAddr.sin\_family = AF\_INET;

    servAddr.sin\_port = htons(SERV\_PORT);

    servAddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

    if (bind(listenSD, (struct sockaddr \*)&servAddr, sizeof(servAddr)) < 0)

    {

        printf("Error: Socket not bind for server.\n");

        return -1;

    }

    if (listen(listenSD, 5) < 0)

    {

        printf("Error: Socket not available for listening.\n");

        return -1;

    }

    while (1)

    {

        clientSD = accept(listenSD, (struct sockaddr \*)NULL, NULL);

        if (fork() == 0)

        {

            close(listenSD);

            processClient(clientSD);

            close(clientSD);

            return 0;

        }

        close(clientSD);

    }

    return 0;

}

1. **Main.h**

typedef struct

{

    char username[30];

    unsigned char password[30];

    char name[30];

    int age;

} UserData;

typedef struct

{

    char username[30];

    unsigned char password[30];

} UserLoginData;

**Output**

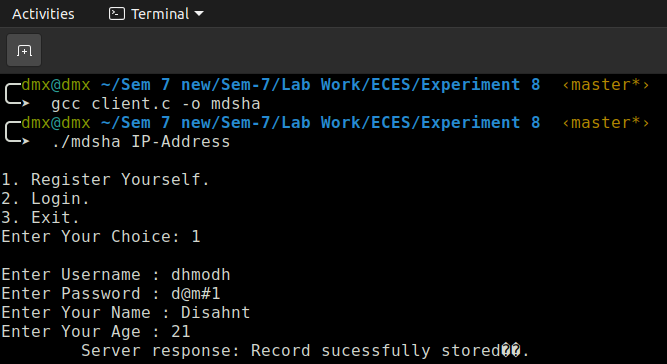
****

Fig. Register from client

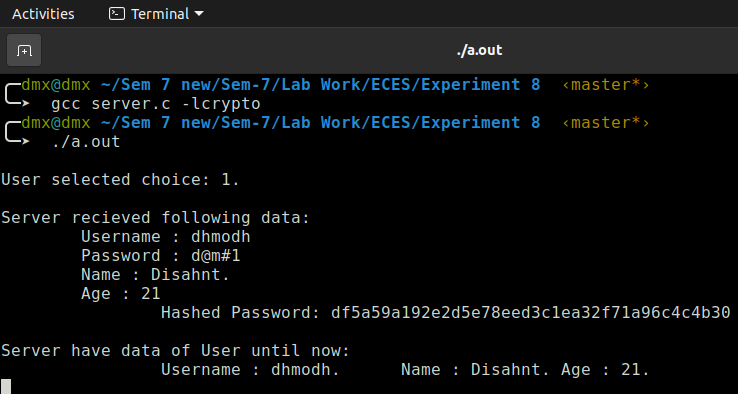


Fig. Server Side

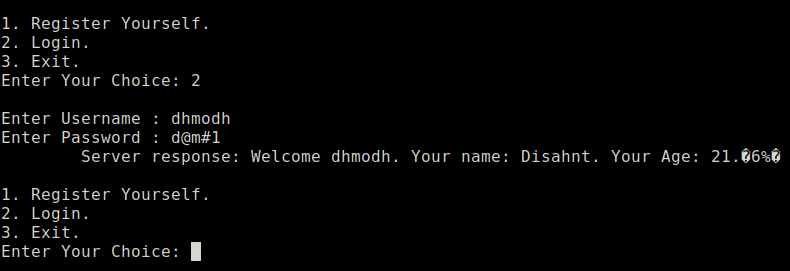


Fig. Client Login

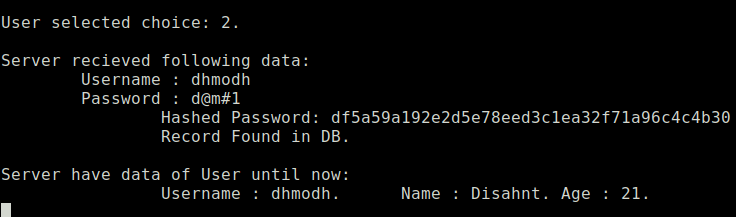


Fig. Server Side after client login

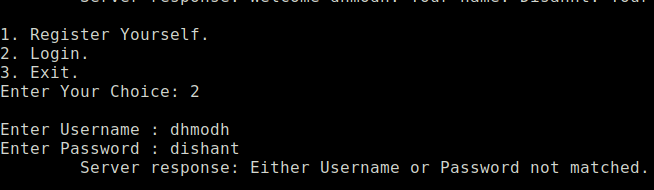


Fig. Client entering Wrong data