Client Server Password Based Authentication Using TCP/IP

A COURSE PROJECT REPORT

By

Pranjal Nikhade (RA2011031010037)
Atandrit Chatterjee (RA2011031010042)
Suvam Nayak (RA2011031010047)
Arsh Bhatia (RA2011031010053)

Under the guidance of

Ms. S. Thenmalar

In partial fulfilment for the Course

of

18CSC302J - COMPUTER NETWORKS

in NWC

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Chenpalpattu District

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this mini project report "**Client Server Password Based Authentication Using TCP/IP**" is the bonafide work of **Pranjal Nikhade (RA2011031010037), Atandrit Chatterjee (RA2011031010042), Suvam Nayak (RA2011031010047)** and **Arsh Bhatia (RA2011031010053)** who carried out the project work under my supervision.

SIGNATURE

Ms. S. Thenmalar
SRM Institute of Science and Technology

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr.C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy,** for his encouragement

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal,** for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman,** for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professor Dr.Annapurani Panaiyappan, Professor and Head, Department of Networking and Communications** and **Course Coordinators** for their constant encouragement and support.

We extend my gratitude to our **HoD Professor Dr.Annapurani Panaiyappan,** and my Departmental colleagues for their Support.

We are highly thankful to our Course project Faculty **Dr.S.Thenmalar, Associate Professor , NWC,** for his/her assistance, timely suggestion and guidance throughout the duration of this course project.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

TABLE OF CONTENTS

1. ABSTRACT

A server assesses if a client has permission to utilize a resource or access a file through the authorization procedure. Authentication and authorization are frequently combined so that the server can identify the client making the access request. Different types of authentication may be required for authorization; some may not require passwords. Good passwords are one of the simplest parts of good network security. Passwords are used to log into systems that use password authentication.

2. INTRODUCTION

1.1 Scenario Description

Authentication is the process of verifying the identity of a user or information. User authentication is the process of verifying the identity of a user when that user logs in to a computer system. The main objective of authentication is to allow authorized users to access the computer and to deny access to unauthorized users.

This Client-Server program would facilitate a client to register itself via request to the server and authenticate the user login based on the user identification and password entered at client side. The server also holds a table that contains all the approved and registered users.

Once connection is established through id and password authentication, user can communicate through the client with the server.

Implementation of a simple client-server program which will facilitate a client to register itself to the server. On the second run it will help the client connect to the server directly by using the registered user id and password. It is implemented using python socket programming. A server keeps a table of user-hash(password) entry for each user.

3. LITERATURE SURVEY

A server assesses if a client has permission to utilize a resource or access a file through the authorization procedure. Authentication and authorization are frequently combined so that the server can identify the client making the access request. Different types of authentication may be required for authorization; some may not require passwords. Good passwords are one of the simplest parts of good network security. Passwords are used to log into systems that use password authentication. Popular mythology says that all network security breaches are caused by sophisticated crackers who discover software security holes. In reality, some of the most famous intruders entered systems simply by guessing or stealing passwords or by exploiting well-known security problems in outdated software.

1. Applying open networks communication authentication

Conference: ITS Telecommunications (ITST), 2011 11th International Conference

Abstract: Information exchanged over open networks in many cases contain sensitive data, accordingly, security became an important issue to many people. Authentication has proven to be a deterrent to intruders and ensures that information is transmitted from trusted senders to a trusted receiver. This work is an extension of previous authentication work. It presents a scrambling and encrypting mechanism to authenticate information in packet headers of the data link layer, HDLC - PPP protocol. HDLC was selected for many reasons one of which is being secured by nature.

Contributors: **Hazem El-Gendy** and **M.W. Youssef**

2. VM NNTP: A TCP/IP server application for news

Article: Computer Networks and ISDN Systems

Abstract: The current evolution in the network world made it necessary to be able to provide access to Netnews from different computer platforms, without the need to store the news more than once. On VM/CMS this is now possible with the VM NNTP server. This server also makes it possible to change the news propagation system used on VM/CMS. With the current tendency to use TCP/IP connections on these systems, it may well prove to be more efficient to handle the news feed process this way. In order to illustrate this view, statistical information is given in this paper.

Contributor: Kris Van Hees and KU Leuven.

3 .Authentication in peer-to-peer systems

Published: Linköping University, Department of Electrical Engineering.

Abstract: In the environment of the 3:rd generation Internet based on peer-to-peer architecture, well-trusted methods must exist to establish a secure environment. One main issue is the possibility to verify that a node actually is who it claims to be (authentication). Establishment of authentication between nodes in a peer-to-peer environment where nodes are exchanging information directly with each other requires more planning than in a typical client-server environment where the authentication methods are server-based. The peer-to-peer applications described in this report use authentication methods based on central authorities as well as solutions without central authorities. Lack of standards in the way peer-to-peer systems should communicate and apply security lead to a variety of "local" communication and security solutions. These local solutions make different applications incompatible with each other, meaning that a peer using one application will not be able to communicate and exchange information with other peers using some other application.

Contributor: Åslund, Jonas.

4. REQUIREMENT ANALYSIS

Requirement are as follows

1. Pycharm IDE for Running Client and Server Side Code

2. The client should be starting first.

3. TCP/IP Network design.

4. Run the server first then the client.

6. Free ports

7. Sockets for connecting server and client.

5 .ARCHITECTURE DESIGN (ALGORITHM)

Server :

- Include the necessary header files.
- Create a socket using socket function with family AF_INET, type as SOCK_STREAM.
- Bind the local host address to socket using the bind function
- Listen on the socket for connection request from the client.
- Accept connection request from the client using accept function.
- Within an infinite loop, using the recv function receive message from the client and print it on the console.
- Using hash Table store the user id along with decrypted password

Client :

- Include the necessary header files
- Create a socket using socket function with family AF_INET, type as SOCK_STREAM.
- Get the server IP address and port number from the console
- Request a connection from the server using the connect function
- Using recv() and send() function , the username and encrypted password is sent from client to server.

6. IMPLEMENTATION

SERVER SIDE CODE :

```
import socket
import os
import threading
import hashlib
ServerSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_STREAM)
host = '127.0.0.1'
port = 1233
ThreadCount = 0
try:
    ServerSocket.bind((host, port))
except socket.error as e:
    print(str(e))

print('Waitiing for a Connection..')
ServerSocket.listen(50)
HashTable = {}



# Function : For each client
def threaded_client(connection):
    connection.send(str.encode('ENTER USERNAME : '))  # Request Username
    name = connection.recv(port)
    connection.send(str.encode('ENTER PASSWORD : '))  # Request Password
    password = connection.recv(port)
    password = password.decode()
    name = name.decode()
    password = hashlib.sha256(str.encode(password)).hexdigest()  # Password hash using SHA256
    # REGISTERATION PHASE
    # If new user,  regiter in Hashtable Dictionary
    if name not in HashTable:
        HashTable[name] = password
        connection.send(str.encode('Registeration Successful'))
        print('Registered : ', name)
        print("{:<8} {:<20}".format('USER', 'PASSWORD'))
        for k, v in HashTable.items():
            label, num = k, v
            print("{:<8} {:<20}".format(label, num))
        print("-------------------------------------------")

    else:
        # If already existing user, check if the entered password is correct
        if (HashTable[name] == password):
            connection.send(str.encode('Connection Successful'))  # Response Code for Connected Client
            print('Connected : ', name)
        else:
            connection.send(str.encode('Login Failed'))
            print('Connection denied : ', name)
    while True:
        break
    connection.close()
while True:
    Client, address = ServerSocket.accept()
    client_handler = threading.Thread(
        target=threaded_client,
        args=(Client,)
    )
    client_handler.start()
    ThreadCount += 1
    print('Connection Request: ' + str(ThreadCount))
ServerSocket.close()
```

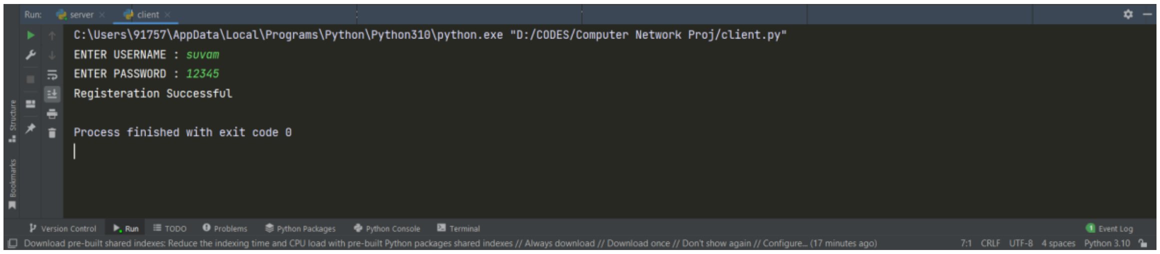
CLIENT SIDE CODE :

```
import socket
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
port=1233
client.connect(('127.0.0.1', port))
response = client.recv(port)
name = input(response.decode())
client.send(str.encode(name))
response = client.recv(port)
password = input(response.decode())
client.send(str.encode(password))
''' Response : Status of Connection :
            1 : Registeration successful
            2 : Connection Successful
            3 : Login Failed
'''
response = client.recv(2048)
response = response.decode()

print(response)
client.close()
```

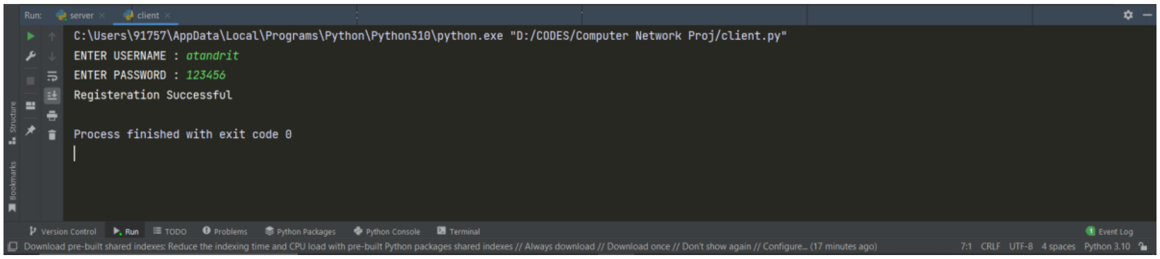
7 .EXPERIMENT RESULTS AND DISCUSSION

O/P

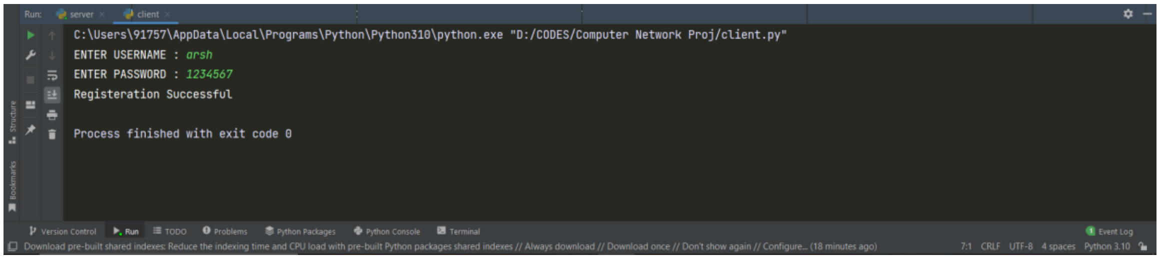
Registration Client :


Run: server client
Version Control Run TODO Problems Python Packages Python Console Terminal Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (17 minutes ago)
7:1 CRLF UTF-8 4 spaces Python 3.10

```
C:\Users\91757\AppData\Local\Programs\Python\Python310\python.exe "D:/CODES/Computer Network Proj/client.py"
ENTER USERNAME : suvam
ENTER PASSWORD : 12345
Registeration Successful

Process finished with exit code 0
```

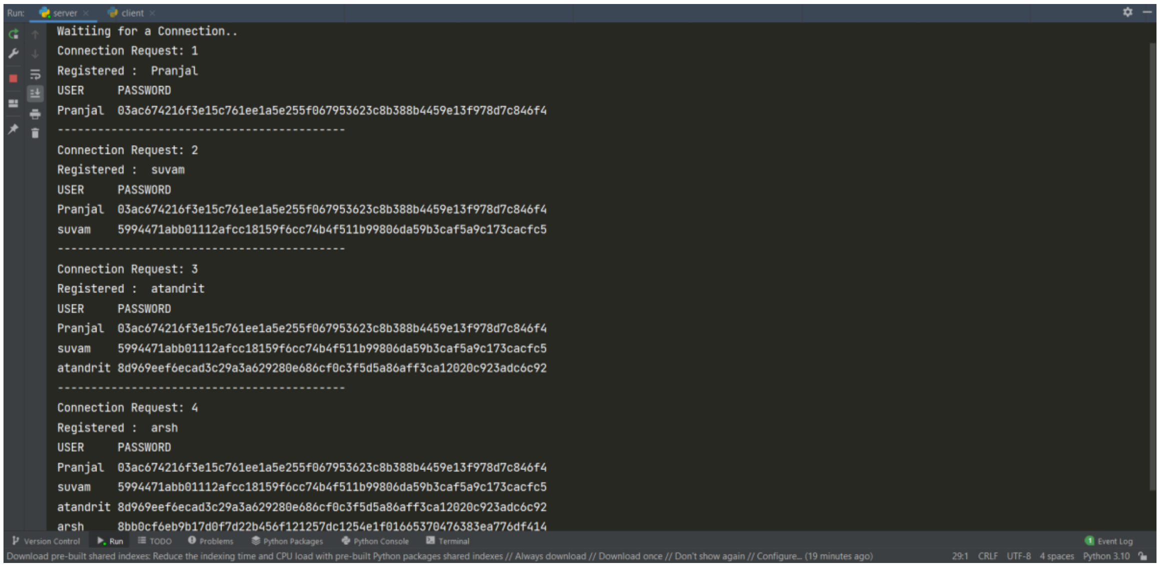

Run: server client
Version Control Run TODO Problems Python Packages Python Console Terminal Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (17 minutes ago)
7:1 CRLF UTF-8 4 spaces Python 3.10

```
C:\Users\91757\AppData\Local\Programs\Python\Python310\python.exe "D:/CODES/Computer Network Proj/client.py"
ENTER USERNAME : atandrit
ENTER PASSWORD : 123456
Registeration Successful

Process finished with exit code 0
```


Run: server client
Version Control Run TODO Problems Python Packages Python Console Terminal Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (18 minutes ago)
7:1 CRLF UTF-8 4 spaces Python 3.10

```
C:\Users\91757\AppData\Local\Programs\Python\Python310\python.exe "D:/CODES/Computer Network Proj/client.py"
ENTER USERNAME : arsh
ENTER PASSWORD : 1234567
Registeration Successful

Process finished with exit code 0
```

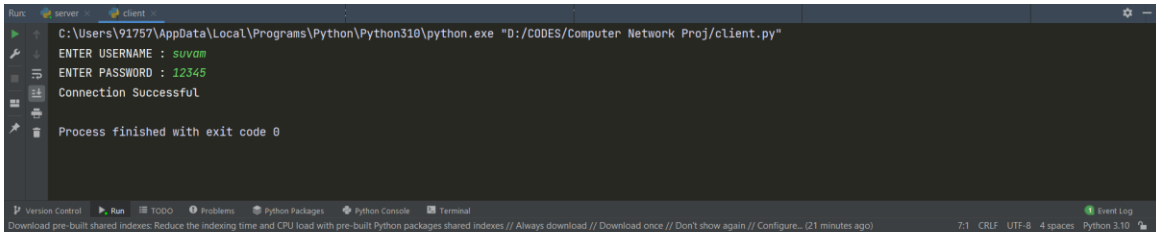
User registration on server :


Run: server client
Version Control Run TODO Problems Python Packages Python Console Terminal Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (19 minutes ago)
29:1 CRLF UTF-8 4 spaces Python 3.10

```
Waitiing for a Connection..
Connection Request: 1
Registered :  Pranjal
USER     PASSWORD
Pranjal  03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
-------------------------------------------
Connection Request: 2
Registered :  suvam
USER     PASSWORD
Pranjal  03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
suvam    5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5
-------------------------------------------
Connection Request: 3
Registered :  atandrit
USER     PASSWORD
Pranjal  03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
suvam    5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5
atandrit 8d969eef6ecad3c29a3a629280e686cf0c3f5d5a86aff3ca12020c923adc6c92
-------------------------------------------
Connection Request: 4
Registered :  arsh
USER     PASSWORD
Pranjal  03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4
suvam    5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5
atandrit 8d969eef6ecad3c29a3a629280e686cf0c3f5d5a86aff3ca12020c923adc6c92
arsh     8bb0cf6eb9b17d0f7d22b456f121257dc1254e1f01665370476383ea776df414
```

Login Successful :

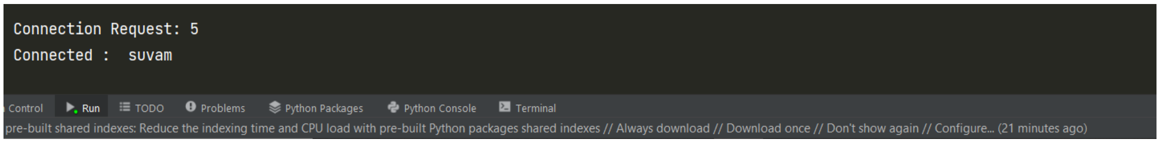
Client :


Run: server client
Version Control Run TODO Problems Python Packages Python Console Terminal Event Log
7:1 CRLF UTF-8 4 spaces Python 3.10

```
C:\Users\91757\AppData\Local\Programs\Python\Python310\python.exe "D:/CODES/Computer Network Proj/client.py"
ENTER USERNAME : suvam
ENTER PASSWORD : 12345
Connection Successful

Process finished with exit code 0
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (21 minutes ago)

Server :

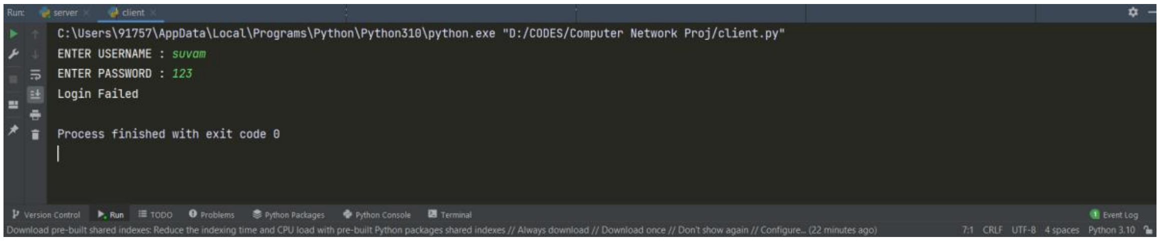

Control Run TODO Problems Python Packages Python Console Terminal

```
Connection Request: 5
Connected :  suvam
```

pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (21 minutes ago)

Login Failed :

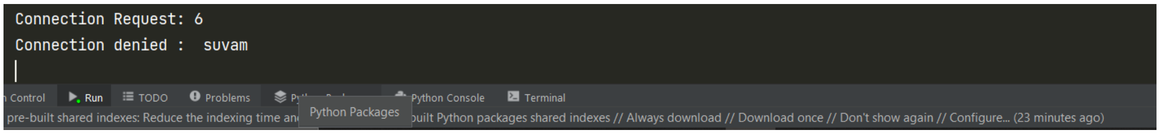
Client :


Run: server client
Version Control Run TODO Problems Python Packages Python Console Terminal Event Log
7:1 CRLF UTF-8 4 spaces Python 3.10

```
C:\Users\91757\AppData\Local\Programs\Python\Python310\python.exe "D:/CODES/Computer Network Proj/client.py"
ENTER USERNAME : suvam
ENTER PASSWORD : 123
Login Failed

Process finished with exit code 0
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (22 minutes ago)

Server :



```
Connection Request: 6
Connection denied :  suvam
```

pre-built shared indexes: Reduce the indexing time and an uilt Python packages shared indexes // Always download // Download once // Don't show again // Configure... (23 minutes ago)

8. CONCLUSION AND FUTURE ENHANCEMENT

Thus authentication using client server communication was established successfully ,using TCP model in which user can register on client end using username and password , wherein the credentials of the user gets stored on server side.

This is the foundation for a highly secured and controlled environment for chat based communication between more than one registered user.

9.REFERENCES

1) https://www.researchgate.net
2) https://en.wikipedia.org/wiki/Transmission_Control_Protocol
3) https://wiki.python.org/moin/TcpCommunication