SASTRA DEEMED TO BE UNIVERSITY
THANJAVUR

Course Code: **CSE303**
Course Name: **Computer Networks Laboratory (CNL)**

CNL Manual

Experiment Number: **2**
Experiment Name: **Development of a secure file transfer application**

Tools: **Java Programming**
Interface**:** **Graphical User Interface (GUI)** using **(Java Swing/AWT)**

Important Classes connection-oriented socket programming:

1. **Socket** class -- to communicate client and server
2. **ServerSocket** class -- to listen clients

Socket class
Object of the Socket class provides features to communicate between client and server.

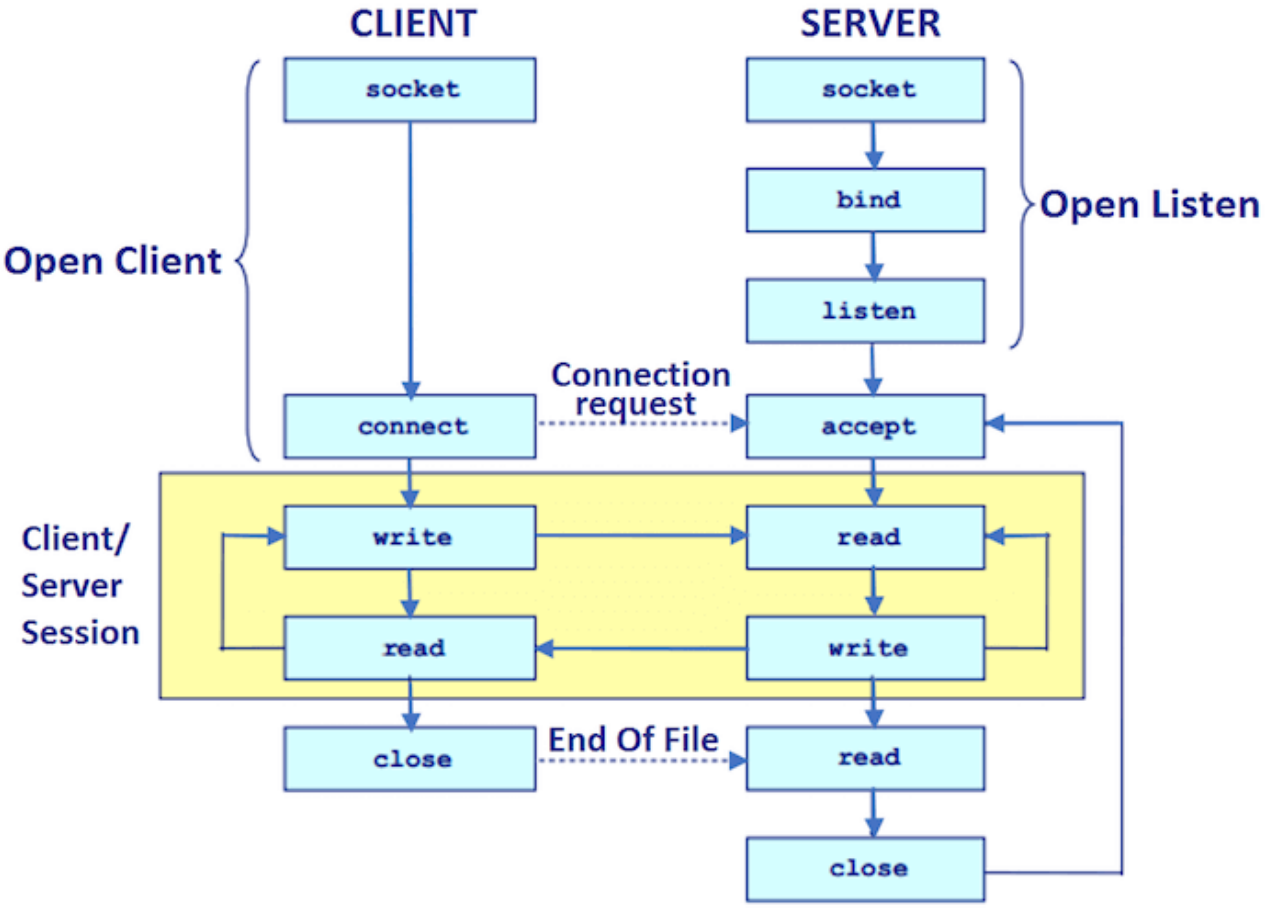
Important methods

Method	Description
1) public InputStream getInputStream()	returns the InputStream attached with this socket.
2) public OutputStream getOutputStream()	returns the OutputStream attached with this socket.
3) public synchronized void close()	closes this socket

ServerSocket class
Object of the ServerSocket to establish communication with the clients.

Important methods

Method	Description
1) public Socket accept()	returns the socket and establish a connection between server and client.
2) public synchronized void close()	closes the server socket.



(https://www.javatpoint.com/socket-programming)

Creating Server:

ServerSocket ss=new ServerSocket(4567);	// 4567 - user defined port number
Socket s=ss.accept();	//establishes connection and waits for the client

Creating Client:

We need to pass the **IP address or hostname** of the **Server** and a **port number**. For the same system, we can either use "localhost" or 127.0.0.1.

Socket s=new Socket("localhost",4567);

Use the Java Swing to create GUI with desired components.

Some important componants are like

1. Use JFrame class to create a GUI.
2. JTextfield ---- for text entry
3. JTextarea ---- text area for display texts
4. JButton ---- button to do some action
5. JPanel ---- to attach other GUI components

To run the client-server programs, following points you need to follow:

1. Keep the S**erver** program (**TCPServer.java**) in separate directory
2. Compile it, and run it.
3. To run it, atleat one argument is required i.e. the path of the S**erver** program such “c:\server program\”.
4. Keep the **Client** program (TCPClient.java) in separate directory
5. Compile it, and run it.
6. To run it, atleat one argument is required i.e. the path of the **Client** program such “c:\client program\”.
7. There two options in Client user interface i.e. Upload and Download.

- **Upload**
- User writes filename such \CData.txt, and then press **Upload** button, which encrypts the client's file using **CAESAR cipher**, and sends to the server.

- **Download**
- User chooses one of the available files from server to download i.e. Sdata.txt, and then puts **“Backslash”** before chosen file like \Sdata.txt, upon pressing **Download** button, the data of the chosen file gets encrypted and sent to the client, and the decryption of the data gets done using **CAESAR cipher**.

CAESAR cipher: it is one of earliest and easist symmetric cipher. It uses 3 as the default key.

Encryption:

CipherText=(PlainText + 3) mod 256 // 256 for entire ASCII data

Decryption:

PlainText=(CipherText – 3) mod 256

if PlainText is <0, then add PlainText with 256;

Expected Sample Output of Lab-2

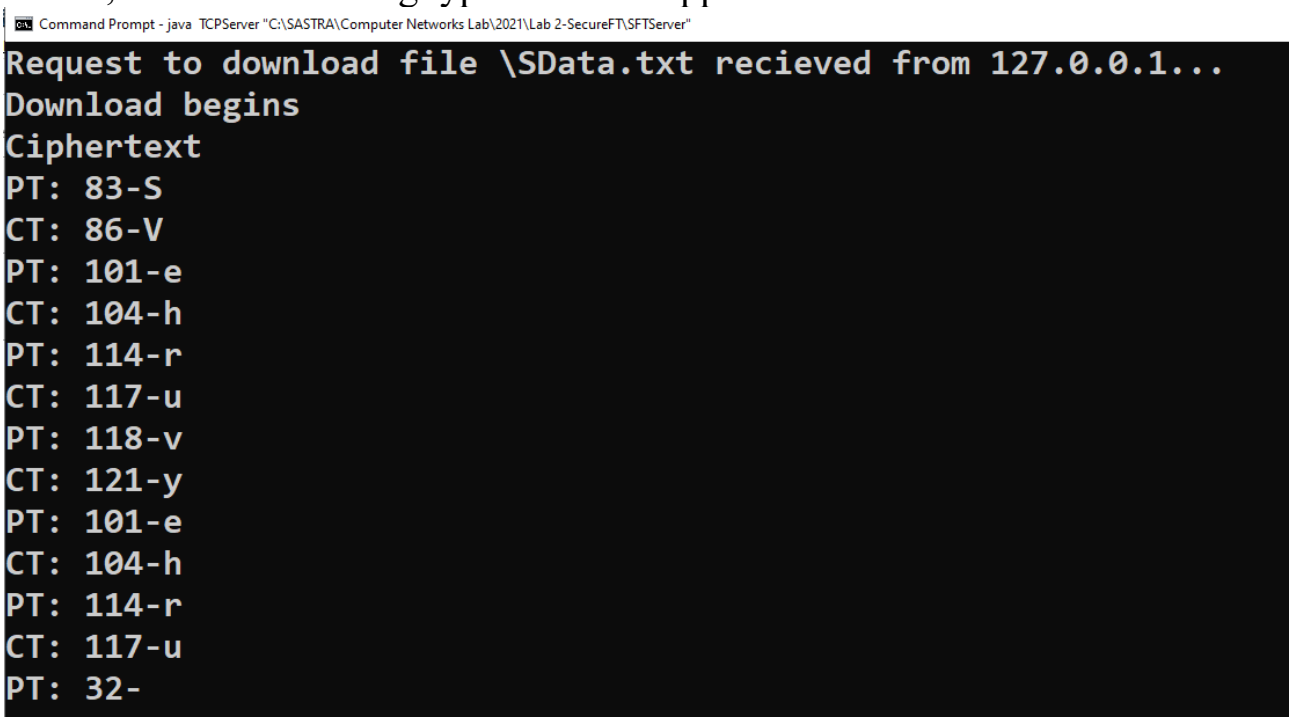
1. Server runs and waits for client

```
C:\SASTRA\Computer Networks Lab\2021\Lab 2-SecureFT\SFTServer>java TCPServer "
C:\SASTRA\Computer Networks Lab\2021\Lab 2-SecureFT\SFTServer"
Server started...
Waiting for connections...
```

2. Client runs, a window gets appeared and waits for user interaction

```
C:\SASTRA\Computer Networks Lab\2021\Lab 2-SecureFT\SFTClient>java TCPClient "
C:\SASTRA\Computer Networks Lab\2021\Lab 2-SecureFT\SFTClient"
Server says Hi!
```

Suppose, you want **download** a file such as SData.txt, and press Download button, then the following type of screen appears.



Note: Verify the **SData.txt** at client's directory, it contains plaintext, because, while downloading at client system, decryption process gets executed.

Suppose, you want **Upload** a file such as CData.txt, and press Upload button, then the following type of screen appears.


```
C:\SASTRA\Computer Networks Lab\2021\Lab 2-SecureFT\SFTClient>java TCPClient "
C:\SASTRA\Computer Networks Lab\2021\Lab 2-SecureFT\SFTClient"
Server says Hi!
9
.gitignore
CData.txt
README.md
reference.txt
SData.txt
TCPServer - Copy.java
TCPServer.class
TCPServer.java
ThreadedServer.class
Upload begins
67
108
105
101
110
```

Here, the ciphertext are shown as numbers.

Note: Verify the **Cdata.txt** at server directory, it contains ciphertext.