© INTERNATIONAL JOURNAL FOR RESEARCH PUBLICATION & SEMINAR
ISSN: 2278-6848 | Volume: 06 Issue: 02 | May-June 2015

Implementation of Socket Programming using C#.net

*Jaspreet Kaur Pasricha,M.Tech Scholar, R.N.C.E.T Panipat, Haryana, Department of Computer Science & engineering
**Virender Kadyan, Assistant Professor, Department of Computer Science & Engineering,R.N.C.E.T Panipat, Haryana

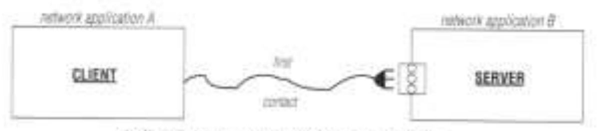
Abstract
TCP socket connections are an essential part of socket programming as they provide a connection oriented service with both flow and congestion control. This means that a TCP connection provides a reliable connection over which data can be transferred with less effort required on the programmer part; TCP takes care of the reliability, flow control, congestion control for you. Firstly we discuss basics concepts, and then we will learn how to implement a simple TCP client and server. This research explains the networking concepts, instance ISO stack, of TCP/IP under the C# framework by employing its essential socket classes and how applications can logically and physically be distributed in a network environment. In this we are going to create a client socket program using C# and a Server Socket Program using C#.net with the use of Encryption and Decryption the message send from client to server transmits securely. Apart from that, we shall learn how to manipulate an IP address and do a DNS lookup to establish a socket connection between the client and server. The anatomy of sockets in depth and examine how to develop a client and server application to set up a socket connection.

1. Socket Programming

The endpoint in an interprocess communication is called a socket, or a network socket. Since most of the communication between two or more computers is based on the Internet Protocol, which is equivalent to *Internet socket*. The data transmission between one or more sockets is organized by communications protocols, which is implemented in the operating system of the computers. Application programs allow us to write and read from these sockets. That's why network programming is essentially known as socket programming.

1. Client server Model

It is possible for the two network processes to begin simultaneously, but it is impractical to require it. Therefore, it needs to design communicating network applications to perform complementary network operations, The server executes first and waits to receive the connection from client; the client executes second and sends the first network packet to the server after connection is established. After initial contact, either the client or the server is capable of sending and receiving data.



A client initiates communications to a server.

2. Overview of IP4 addresses

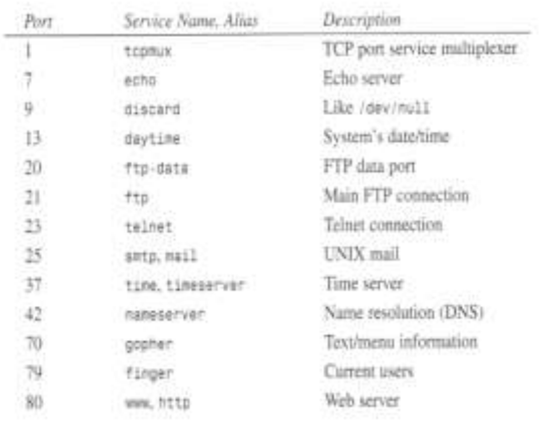
IP4 addresses are 32 bits long. They are expressed commonly which is also known as dotted decimal notation. Each of the four bytes which makes up the 32 address are expressed as an integer value (0 – 255) and separated by a dot. For example, 255.35.26.5 is an example of an IP4 address in dotted decimal notation. There are various conversion functions which convert 32 bit address into a dotted decimal string and vice versa. Often though the IP address is represented by a domain name, for example, army.navy.in. There are several functions described which will allow us to convert from one form into another. The importance of IP addresses follows from the fact that each host on the Internet has a unique IP address. Therefore, the Internet is made up of many networks of networks with many different types of architectures and transport, it is the IP address which provides a structure, (there are routing issues involved as well), any two hosts on the Internet can communicate with each other.

3. Port

Sockets are UNIQUELY identified by Internet address, end-to-end protocol, and port number. That's why when a socket is first created it is vital to match it with a valid IP address and a port number. We are basically working with TCP sockets. Ports are software objects to multiplex data between different applications. When a host receives a packet, it travels up the protocol stack and finally reaches the application layer. Now consider if a user running an ftp client, a telnet client, and a web browser concurrently. To find out which application should the packet be delivered? Well part of the packet contains a value holding a port number, and it is the number which determines to which of the given application the packet should be delivered.
So when a client first tries to contact a server, which port number should the client specify? For many common services, standard port numbers are defined

© INTERNATIONAL JOURNAL FOR RESEARCH PUBLICATION & SEMINAR
ISSN: 2276-6848 | Volume: 06 Issue: 02 | May-June 2015

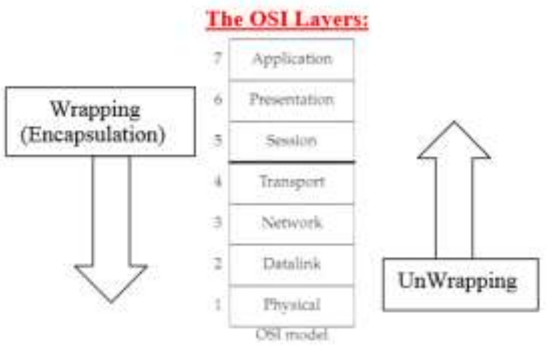
Ports 0 – 1023, are reserved and servers or clients that you create will not be able to Bind to these ports unless you have root privilege



Port	*Service Name, Alias*	*Description*
1	tcpmux	TCP port service multiplexer
7	echo	Echo server
9	discard	Like /dev/null
13	daytime	System's date/time
20	ftp-data	FTP data port
21	ftp	Main FTP connection
23	telnet	Telnet connection
25	smtp, mail	UNIX mail
37	time, timeserver	Time server
42	nameserver	Name resolution (DNS)
70	gopher	Text/menu information
79	finger	Current users
80	www, http	Web server

.
Ports 1024 – 65535 are available for use by your programs, but beware other network applications maybe running and using these port numbers as well so do not make assumptions about the availability of specific port numbers.

4. The Socket Interface



5. The Proposed Protocol

In this research we are going to **create a** socket using C# in which the client sends the encrypted data using the keys from the MS- Access to the server using the port number and IP address. The particular server having the IP Address used by the client is able to decrypt that data using the keys. First the server is waiting for the client to connect after connection is established between the client and the server the client sends the message to the server using encryption so that only that server having decryption keys is able to read the contents of the message. The data transmits securely between the client and the server.

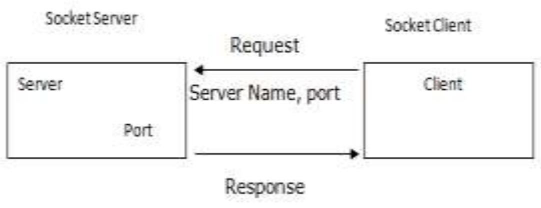
i. Server is waiting for the connection with the client.
ii. After Connection the client sends the message using encryption keys, port number, IP Address of the server.
iii. Server receives the message
iv. Server decrypts the message using Decryption Keys

6. C# Socket programming

C# simplifies the network programming through its namespaces like **System.Net** and **System.Net.Sockets** . A Socket is an End-Point of To and From (Bidirectional) communication link between two programs (Server Program and Client Program) running on the same network . We need two programs for communicating a socket application in C#. A Server Socket Program (Server) and a Client Socket Program (Client) .

C# Server Socket Program: A C# Server Socket Program running on a computer has a socket that bound to a Port Number on the same computer and listening to the client's incoming requests. C# Client Socket Program: A C# Client Socket Program have to know the IP Address (Hostname) of the computer that the C# Server Socket Program resides and the Port Number assign for listening for client's request.

Once the connection is established between Server and Client, they can communicate (read or write) through their own sockets.



There are two types of communication protocol uses for Socket Programming in C# ,

1. **TCP/IP** (Transmission Control Protocol/Internet protocol) Communication
2. **UDP/IP** (User Datagram Protocol/Internet protocol) Communication .

In the following section we are going to communicate C# Server Socket Program and C# Client Socket Program using TCP/IP Communication Protocol.
The above picture shows a Server and Client communication interfaces in C#.
C# Server Socket Program:
The Server Socket Program is done through a C# Console based application. Here the Server is listening for the Client's request, and when the C#

Server gets a request from Client socket , the Server sends a response to the Client .

7. Server Socket Programming

The Server Socket Program here is a **C# Console based Application**. This program act as a Server and listening to clients request. Here we assign a Port No. 8888 for the Server Socket, it is an instance of the C# Class TcpListener , and call its **start()** method.

TcpListener serverSocket = new TcpListener(8888);

serverSocket.Start();

The next step is to create an infinite loop for monitoring the request from Client's side . When the Server Socket accept a request from the Client side, it reads the data from Network Stream and also it write the response to **Network Stream** .

8. C# Client Socket Programming

The C# Client Socket Program is a windows based application. When the C# Client program execute , it will establish a connection to the C# Server program and send request to the Server , at the same time it also receive the response from C# Server . Click the following link to see in detail of C# Client Socket Program.

The C# Client Socket Program is the second part of the C# Server Socket Program. The C# Client Socket Program is a Windows based application . The Client is connected to the Port 8888 of the C# Server Socket Program , and the IP Address (Computer Name) here we give as 127.0.0.1 , because the Server and Client running on the same machine .

clientSocket.Connect("127.0.0.1", 8888);

When the C# Client program starts, it will connect to the C# Server Socket Program and start to reads data from NetworkStream , and also write to the NetworkStream . When you start the client program you will get a message from Server "client started". When press the button at the bottom of Client window, it will send a message to the Server and also receive response from the Server.

9. Scope of Implementation

A *socket* is one of the most fundamental technologies of computer networking. Sockets allow applications to communicate using standard mechanisms built into network hardware and operating systems. Although network software may seem to be a relatively new "Web" phenomenon, socket technology actually has been employed for roughly two decades.

Software applications that rely on the Internet and other computer networks continue to grow in popularity. Many of today's most popular software packages -- including Web browsers, instant messaging applications and peer to peer file sharing systems -- rely on sockets.

10. References

1. C#.net –Server socket Implementation http://csharp.net-informations.com/communications/csharp-server-socket.htm
2. Introduction to Socket Programming http://alumni.cs.ucr.edu/~ecegelal/TAw/socketTCP.pdf
3. Introduction to Socket and socket programming http://compnetworking.about.com/od/itinformationtechnology/l/aa083100a.htm
4. Networking Complete Reference (TMH)
5. Encyclopedia of Networking (PHI)
6. Shankland S (2011) Amazon cloud outage derails Reddit, Quora
7. Anand N (2010) The legal issues around cloud computing. http://www.labnol.org/internet/cloud-computing-legal-issues/14120/
8. Kaufman C, Venkatapathy R (2010) Windows Azure Security Overview.go.microsoft.com/?linkid=9740388, [August].
9. Kandukuri BR, Paturi R, Rakshit A (2009) Cloud Security Issues. In:Proceedings of the 2009 IEEE International Conference on Services Computing, SCC '09
10. Yan L, Rong C, Zhao G (2009) Strengthen Cloud Computing Security with Federal Identity Management Using Hierarchical Identity-Based Cryptography. In: Proceedings of the 1st International Conference on Cloud Computing, CloudCom '09
11. Musthaler L (2009) Cost-effective data encryption in the cloud. Network World