**Operating System**

**Lab Report 7**

**Hafiz Ahmad**

**19l-1316**

**Section-6A2**

Inter Process Communication using Sockets

**INTRODUCTION:**

A socket is defined as an endpoint for communication. A pair of processes communicating over a network employs a pair of sockets—one for each process. A socket is identified by an IP address concatenated with a port number. In general, sockets use a client–server architecture. The server waits for incoming client requests by listening to a specified port. Once a request is received, the server accepts a connection from the client socket to complete the connection. Servers implementing specific services (such as telnet, FTP, and HTTP) listen to well-known ports (a telnet server listens to port 23; an FTP server listens to port 21; and a web, or HTTP, server listens to port 80). All ports below 1024 are considered well known; we can use them to implement standard services.

• Communication points on the same or different computers to exchange data

• Allows communication between two different processes on the same of different machines

• Always has an address (IP and Port)

• A UNIX socket is used in a client-server application framework

• Server is a process that performs some functions on request from a client

• Just like a file (open, close, read, write).

**OBJECTIVES:**

• Learn and Understand InterProcess Communication using socket programming

**Application:**

Sockets allow you to exchange information between processes on the same machine or across a network, distribute work to the most efficient machine, and they easily allow access to centralized data. Socket application program interfaces APIs are the network standard for TCP/IP. Sockets provide point-to-point, two-way communication between two processes. Sockets are a basic component of inter process and intersystem communication. A socket is an endpoint of communication to which a name can be bound. It has a type and one or more associated processes. Sockets exist in communication domains.

**Examples of IPC systems**

* Posix : uses shared memory method.
* Mach : uses message passing.
* Windows XP : uses message passing using local procedural calls.

**Issues:**

No issue found regarding this lab.

**Conclusion:**

In this lab we that how to communicate between two process using socket programming .