

INTERNATIONAL SCHOOL OF MANAGEMENT & TECHNOLOGY

**ASSIGNMENT COVER SHEET**

|  |  |  |  |
| --- | --- | --- | --- |
| STUDENT DETAILS | | | |
|  | | | |
| Student ID |  | **Reg No.** |  |
| Family Name | Shrestha | **Given Name** | Bisesh |
| Enrolment Year | 2019 | **Section** | N4 |
| Semester | 4th | **Email** | bisesh@ismt.edu.np |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UNIT DETAILS | | | | |
| Unit Title | Client/Server Computing Systems | **Unit Code** | M/615/1641 | |
| Assessor Name | Sunil Shrestha | **Issued Date** | 07/29/2021 | |
| Assignment Title | Client/Server Computing Systems | | | |
| Assignment No | **1/1** | **Submission Date** | | 07/29/2021 |
| Qualification | BTEC HND IN COMPUTING | **Campus** | | **ISMT** |





**STUDENT ASSESSMENT SUBMISSION AND DECLARATION**

When submitting evidence for assessment, each student must sign a declaration confirming that the work is their own.

|  |  |  |  |
| --- | --- | --- | --- |
| Student Name | Bisesh Shrestha | Assessor Name | Sunil Shrestha |
| Issue Date | 06/03/2021 | Submission Date | 08/02/2021 |
| Programme | Networking | | |
| Unit Name | Client/Server Computing Systems | | |
| Assignment Title | Client/Server Computing Systems | | |

**Plagiarism**

Plagiarism is a particular form of cheating. Plagiarism must be avoided at all costs and students who break the rules, however innocently, may be penalized. It is your responsibility to ensure that you understand correct referencing practices. As a university level student, you are expected to use appropriate references throughout and keep carefully detailed notes of all your sources of materials for material you have used in your work, including any material downloaded from the Internet. Please consult the relevant unit lecturer or your course tutor if you need any further advice.

**Student Declaration**

I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.



Pearson Education 2018  
Higher Education Qualifications

Contents

[Part1 4](#_Toc94356361)

[P1 Illustrate the communication processes between servers and clients. 4](#_Toc94356362)

[Introduction 4](#_Toc94356363)

[Client-Server Model 4](#_Toc94356364)

[How the Client-Server Model works? 4](#_Toc94356365)

[How the browser interacts with the servers? 5](#_Toc94356366)

[Advantages of Client-Server model: 6](#_Toc94356367)

[Disadvantages of Client-Server model: 6](#_Toc94356368)

[Client/Server Communication 6](#_Toc94356369)

[Pipes 6](#_Toc94356370)

[Sockets 7](#_Toc94356371)

[Remote Procedure Calls 7](#_Toc94356372)

[Example of applications using client-server system 8](#_Toc94356373)

[File transfer: 8](#_Toc94356374)

[Mail transfer: 8](#_Toc94356375)

[Hypertext Transfer Protocol 9](#_Toc94356376)

[Client-Server Systems Architecture 9](#_Toc94356377)

[ 2-tier client-server system architecture: 10](#_Toc94356378)

[ 3-tier client-server system architecture: 10](#_Toc94356379)

[Issues and Challenges in Client-Server System 10](#_Toc94356380)

[P2 Compare parent process and child process, PID and PPID. 10](#_Toc94356381)

[Process 11](#_Toc94356382)

[Parent Process 11](#_Toc94356383)

[Child Process 11](#_Toc94356384)

[PID & PPID 12](#_Toc94356385)

[Conclusion 12](#_Toc94356386)

[Part2 13](#_Toc94356387)

[Introduction 13](#_Toc94356388)

[P3 Examine the concepts and functions of several Application Layer protocols. 13](#_Toc94356389)

[Functions of Application Layer: 13](#_Toc94356390)

[Working of Application Layer 14](#_Toc94356391)

[Features provided by Application Layer Protocols: 14](#_Toc94356392)

[Application Layer Protocols 14](#_Toc94356393)

[SMTP (Simple Mail Transfer Protocol): 15](#_Toc94356394)

[TELNET (Terminal Network): 16](#_Toc94356395)

[FTP (File Transfer Protocol): 16](#_Toc94356396)

[Multipurpose Internet Mail Extensions (MIME): 17](#_Toc94356397)

[POP (Post Office Protocol): 17](#_Toc94356398)

[HTTP (Hypertext Transfer Protocol): 18](#_Toc94356399)

[Domain Name System (DNS): 18](#_Toc94356400)

[P4 Operate Wireshark software, and recognize the different windows in the Wireshark interface. 19](#_Toc94356401)

[Wireshark 19](#_Toc94356402)

[How to Install Wireshark on Windows 20](#_Toc94356403)

[Wireshark Interface and Operation 27](#_Toc94356404)

[How to Install Wireshark on Linux 28](#_Toc94356405)

[Conclusion 28](#_Toc94356406)

[Part3 29](#_Toc94356407)

[Introduction 29](#_Toc94356408)

[Linux 29](#_Toc94356409)

[History of Linux 29](#_Toc94356410)

[Advantages of Linux operating system 29](#_Toc94356411)

[Disadvantages of Linux operating system 30](#_Toc94356412)

[P5 Create a UDP system in Linux. 31](#_Toc94356413)

[Server 31](#_Toc94356414)

[Client 34](#_Toc94356415)

[P6 Create a TCP system in Linux 36](#_Toc94356416)

[Server 36](#_Toc94356417)

[Compilation of server code 38](#_Toc94356418)

[Compilation of client code 38](#_Toc94356419)

[Connection between server and client 38](#_Toc94356420)

[Conclusion 39](#_Toc94356421)

[References 40](#_Toc94356422)

# Part1

Explore the concepts of servers, clients, and processes and the differences between PPID and PID

# P1 Illustrate the communication processes between servers and clients.

# Introduction

In this lesson, I'll explain the client-server concept and the communication mechanism between the server and the clients. In addition, I'll talk about the parent and child processes, as well as the PID and PPID. With recent technological advancements, the web is becoming increasingly important in our day-to-day lives, with the web being used in almost everything we do these days. Furthermore, the Web's engineering is based on the Client-Server model, which means that the primary concern should be communication between the server and the client.

# Client-Server Model

The client-server paradigm is a distributed application structure that divides tasks or workload between servers, who supply a resource or service, and clients, who request that service. When a client computer submits a data request to the server via the internet, the server acknowledges the request and returns the data packets requested to the client. Clients do not share any of their resources with one another. Email, the World Wide Web, and other client-server models are examples.

## How the Client-Server Model works?

In this part I am going to take a dive into the Client-Server model and have a look at how the Internet works via, web browsers. This article will help us in having a solid foundation of the WEB and help in working with WEB technologies with ease.

* Client: When we talk the word Client, it mean to talk of a person or an organization using a particular service. Similarly in the digital world a Client is a computer (Host) i.e. capable of receiving information or using a particular service from the service providers (Servers).
* Servers: Similarly, when we talk the word Servers, It mean a person or medium that serves something. Similarly in this digital world a Server is a remote computer which provides information (data) or access to particular services.

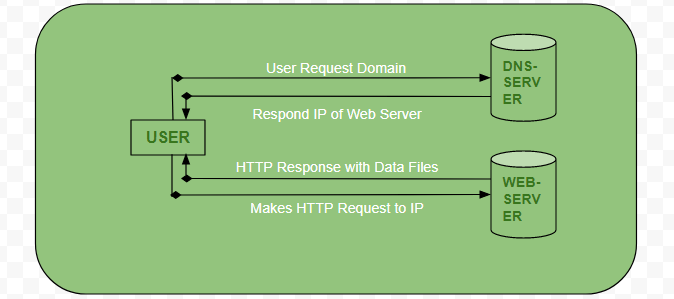
So, its basically the Client requesting something and the Server serving it as long as its present in the database.



## How the browser interacts with the servers?

There are few steps to follow to interact with the servers a client.

* User enters the URL (Uniform Resource Locator) of the website or file. The Browser then requests the DNS (DOMAIN NAME SYSTEM) Server.
* DNS Server lookup for the address of the WEB Server.
* DNS Server responds with the IP address of the WEB Server.
* Browser sends over an HTTP/HTTPS request to WEB Server’s IP (provided by DNS server).
* Server sends over the necessary files of the website.
* Browser then renders the files and the website is displayed. This rendering is done with the help of DOM (Document Object Model) interpreter, CSS interpreter and JS Engine collectively known as the JIT or (Just in Time) Compilers.



## Advantages of Client-Server model:

* Centralized system with all data in a single place.
* Cost efficient requires less maintenance cost and Data recovery is possible.
* The capacity of the Client and Servers can be changed separately.

## Disadvantages of Client-Server model:

* Clients are prone to viruses, Trojans and worms if present in the Server or uploaded into the Server.
* Server are prone to Denial of Service (DOS) attacks.
* Data packets may be spoofed or modified during transmission.
* Phishing or capturing login credentials or other useful information of the user are common and MITM (Man in the Middle) attacks are common.

# Client/Server Communication

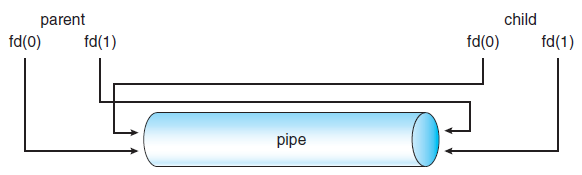
There are three main methods to client/server communication. The following is a list of them:

## Pipes

These are methods of interprocess communication that have two end points. A process enters data from one end of the pipe, while the other process consumes it from the other end.

* Ordinary pipes and named pipes are the two sorts of pipes. Ordinary pipes can only transmit data in one direction. Two pipes are necessary for two-way communication. Because ordinary pipes may only be accessed by processes that originated or inherited them, they have a parent-child connection between them.
* Named pipes are more powerful than regular pipes and may communicate in both directions. These pipelines continue to remain long after the processes that used them have ended. When they are no longer needed, they must be expressly erased.

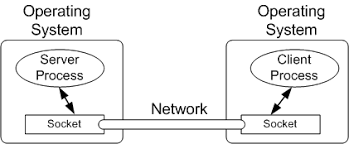
The following is a diagram that shows pipes:



## Sockets

Sockets allow two processes on the same system or on separate computers to communicate with one another. They consist of the IP address and port number and are utilized in a client/server framework. Sockets are used by many application protocols to connect and send data between a client and a server. Because sockets merely communicate an unstructured byte stream between processes, socket communication is rather low-level. The client and server apps impose a structure on the byte stream.

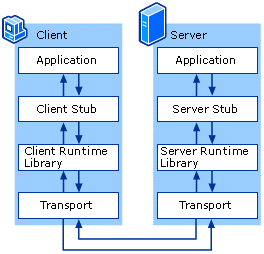
The following is a diagram that shows sockets:



## Remote Procedure Calls

These are techniques for interprocess communication that are used in client-server systems. A subroutine call or a function call is another name for a remote procedure call. The RPC receives a request from a client and interprets it before sending it to the server. This request to a remote server might be a procedure or a function call. When the server gets the request, it responds with the appropriate information.

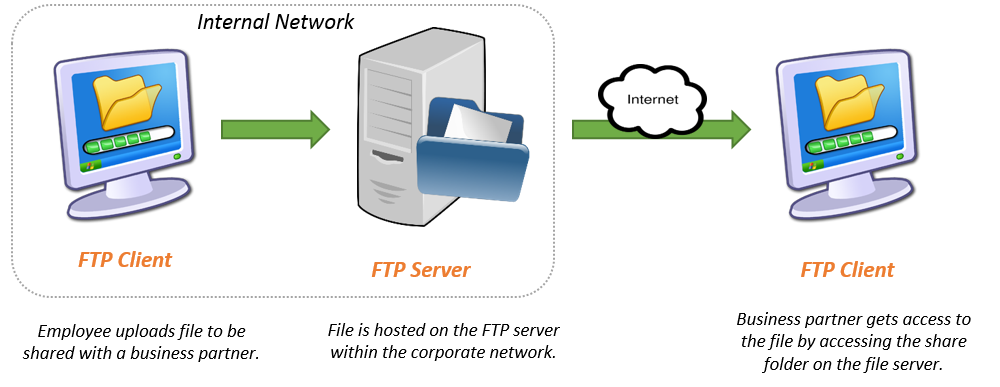
The following is a diagram that shows remote procedure calls:



# Example of applications using client-server system

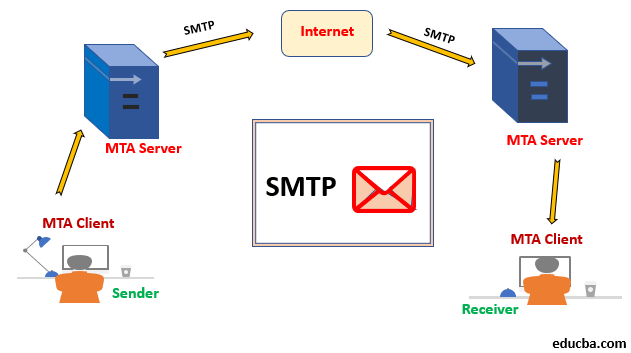
## File transfer:

This is the transmission of files between the client and server. It also allows storing of files on the server. Files such as movies, images, music can be stored.



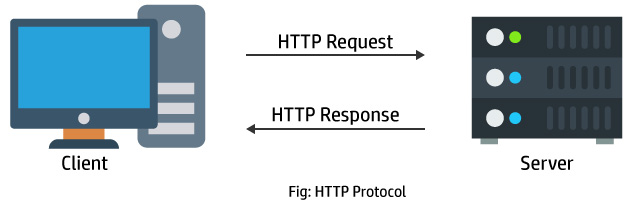
## Mail transfer:

This is the transfer of messages such as the email using the Mail Transfer Protocol (MTP).



## Hypertext Transfer Protocol

The Hypertext Transfer Protocol (HTTP) is a protocol for sending and receiving multimedia files including images and text between a client and a server. HTTP is a request-response protocol that is used to improve communication between the client and the server.



# Client-Server Systems Architecture

Client/server architecture is a computing model in which various components communicate by performing specific tasks in clearly defined roles. The server is responsible for hosting, delivering, and managing the majority of the resources and services that the client will use. One or more client computers are connected to a central server through a network or the internet in this sort of shared resources architecture.

Because all requests and services are supplied across a network, client/server architecture is also known as a networking computing model or client/server network. Because the components function independently of one another, it is classified as a distributed computing system. The application server, database server, and PC are commonly included in a client-server architecture. The 2-tier and 3-tier architectures are the two most common architectures.

## 2-tier client-server system architecture:

Only the database server and a client PC are used in this design. Users will execute apps on their PCs (Clients) that connect to the server through a network in a two-tier architecture. The client application executes the coding as well as the business logic before displaying the results to the user. It's also known as a thick client. It is regarded when a client gets direct access to a database without the need for an intermediary. It's also utilized for application logic, in which the application code is allocated to each client in the workstation.

## 3-tier client-server system architecture:

The client PC, database server, and application server are all part of this architecture. A three-tier design may be extended to an N-tier architecture, which includes additional application servers. Because the client just has display logic in this design, the client requires fewer resources and code. It allows a single server to manage a large number of clients while also increasing the server's resources.

# Issues and Challenges in Client-Server System

There are lots of issues in a client-server system, some of which include: -

* The number of people: In most client-server network, few servers are always involved which make the setup looks like a waste of time. A client server network is quite difficult to set up, so it requires lots of servers so as not to render the application useless.
* It requires skilled expertise: A lot of client-server network are not well build and managed. Setting up a client server network is so complex, so it requires skilled technician and maintenance engineers to handle it.
* The servers are quite expensive: The servers are being designed to meet high standard for it to be reliable and have better performance. Getting the good design and architecture for the client-server system, contributes to it being expensive.
* Security: The most of all is the security issue. The client operating system is easily accessed by servers, and this expose the client system to a number of problems. The message being exchange between the client and server leads to lot of security challenges. A lot of things are encompassing by the security challenges, some of which include: physical damage, threats and virus attack. In other to have a secured network, there are some basic rules to be followed such as;
  + The Audit network
  + The discretionary control
  + The object re-use
  + Authorization and identification

# P2 Compare parent process and child process, PID and PPID.

A process uses the fork () system function to generate another process in the Operating System. The parent process is the one that used the fork () system call, and the child process is the one that was generated as a result.

The following are some of the specific details:

## Process

An active program, or one that is now being executed, is referred to as a process. It includes the program counter, process stack, registers, and program code, among other things. The program code is only the text component in comparison. As it runs, a process changes its state. This state is influenced by a process's present activity. A process can be new, ready, running, blocked, or terminated at any time throughout its execution. Each of the processes has its own process control block. Process status, process number, program counter, list of files and registers, CPU information, RAM information, and so on are all included.

## Parent Process

Except for the startup process, all processes in the operating system are formed when a process calls the fork() system function. The parent process is the one that used the fork () system function. A parent process, in other terms, is one that generates a child process. While a parent process can have several child processes, a child process can only have one parent process.

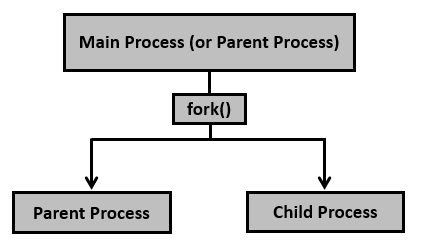
The PID of the child process is returned to the parent process upon the success of a fork () system call, whereas 0 is returned to the child process. When a fork () system call fails, the parent process receives -1 and no child process is generated.

## Child Process

A child process is a process generated by a parent process using the fork () system function in the operating system. A sub process or a subtask is another name for a child process.

A child process is formed as a clone of its parent process, inheriting the majority of its properties. If a child process has no parent process, the kernel generated it directly. A SIGCHLD signal is sent to the parent process when a child process terminates or is halted.

**The following is a diagram depicting the parent-child process.**



Some differences between the child and parent process are:

* They have different pids
* In the parent, fork( ) returns the pid of the child process if a child process is created
* In the child, fork( ) always returns 0
* Separate copies of all data, including variables with their current values and the stack
* Separate program counter (PC) indicating where to execute next; originally both have the
* same value but they are thereafter separate
* After fork, the two processes do not share variables

# PID & PPID

In Linux, a program is an executable saved on disk, while a process is an executable loaded into memory and executing. A process is assigned a unique number called a process ID (PID) when it is initiated, which identifies it to the system. You can refer to a process by its PID, for example, if you need to terminate it. Each process has a unique process ID as well as a parent process ID (PPID) that indicates which process launched it. The PPID is the process's parent's PID. If process1 with a PID of 101 begins a process named process2, process2 will be assigned a unique PID, such as 3240, but the PPID will remain 101. It's a relationship between a parent and a child. A single parent process can have several children, each with their own PID but sharing the same PPID.

When we run a program on your UNIX system, the system generates a unique environment for it. This environment provides everything required for the system to run the application as though there were no other programs running. A new process is created or started whenever you issue a command under UNIX. You began a process when you used the is command to list the contents of a directory. In basic words, a process is an instance of a running program.

The pid, or process ID, is a five-digit ID number used by the operating system to monitor processes. A unique pid is assigned to each process in the system. Because all the potential numbers have been used up, pid ultimately repeat themselves, and the next pid rolls or starts over. Because UNIX employs the pid to track each process, no two processes with the same pid exist in the system at any given moment.

# Conclusion

I explained the client-server concept and the communication procedure between the server and the clients in this part. I went through client server architecture, client server communication, and client server advantages. In addition, I've talked about the parent and child processes, as well as the PID and PPID.

# Part2

**Analyze the communication processes between clients and servers in different application protocols with records captured from the internet**

# Introduction

In this part, I'll talk about how clients and servers communicate in different application protocols, as well as look at the fundamental principles and functions of many application layer protocols. In the following section, I'll go into Wireshark's idea, function, history, and benefits and drawbacks. I will also produce documentation about Wireshark installation and basic operation, as requested by this task.

# P3 Examine the concepts and functions of several Application Layer protocols.

In the Open System Interconnection (OSI) model, the Application Layer is the uppermost layer. This layer includes numerous methods for modifying data (information), allowing any sort of user to easily access the network. This layer also sends a request to the presentation layer, which is its bottom layer, for various sorts of information. The Application Layer interacts with the application directly and offers typical web application capabilities. This layer is the highest level of an open system, and it delivers services to application processes directly.

## Functions of Application Layer:

The Application Layer, as discussed above, being topmost layer in OSI model, performs several kinds of functions which are requirement in any kind of application or communication process.

Following are list of functions which are performed by Application Layer:

* Application Layer provides a facility by which users can forward several emails and it also provides a storage facility.
* This layer allows users to access, retrieve and manage files in a remote computer.
* It allows users to log on as a remote host.
* This layer provides access to global information about various services.
* This layer provides services which include: e-mail, transferring files, distributing results to the user, directory services, and network resources and so on.
* It provides protocols that allow software to send and receive information and present meaningful data to users.
* It handles issues such as network transparency, resource allocation and so on.
* This layer serves as a window for users and application processes to access network services.
* Application Layer is basically not a function, but it performs application layer functions.
* The application layer is actually an abstraction layer that specifies the shared protocols and interface methods used by hosts in a communication network.
* Application Layer helps us to identify communication partners, and synchronizing communication.
* This layer allows users to interact with other software applications.
* In this layer, data is in visual form, which makes users truly understand data rather than remembering or visualize the data in the binary format (0’s or 1’s).
* This application layer basically interacts with Operating System (OS) and thus further preserves the data in a suitable manner.
* This layer also receives and preserves data from it’s previous layer, which is Presentation Layer (which carries in itself the syntax and semantics of the information transmitted).
* The protocols which are used in this application layer depend upon what information users wish to send or receive.
* This application layer, in general, performs host initialization followed by remote login to hosts

## Working of Application Layer

The application layer in the OSI model generally acts only like the interface which is responsible for communicating with host-based and user-facing applications. This is in contrast with TCP/IP protocol, wherein the layers below the application layer, which is Session Layer and Presentation layer, are clubbed together and form a simple single layer which is responsible for performing the functions, which includes controlling the dialogues between computers, establishing as well as maintaining as well as ending a particular session, providing data compression and data encryption and so on.

At first, client sends a command t server and when server receives that command, it allocates port number to client. Thereafter, the client sends an initiation connection request to server and when server receives request, it gives acknowledgement (ACK) to client through client has successfully established a connection with the server and, therefore, now client has access to server through which it may either ask server to send any types of files or other documents or it may upload some files or documents on server itself.

## Features provided by Application Layer Protocols:

To ensure smooth communication, application layer protocols are implemented the same on source host and destination host.

The following are some of the features which are provided by Application layer protocols-

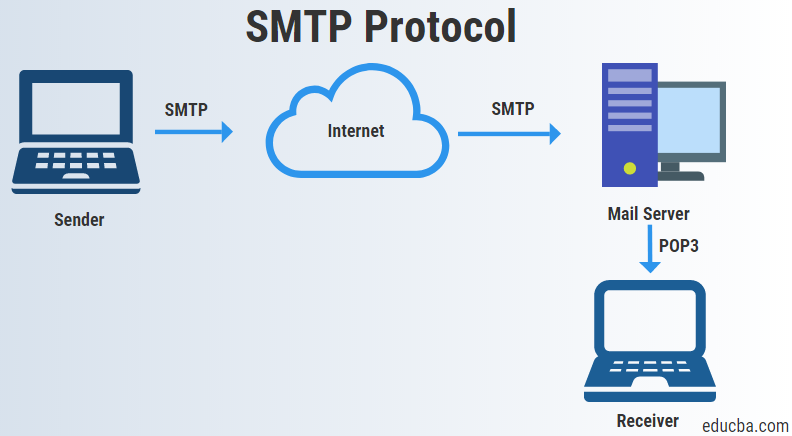
* The Application Layer protocol defines process for both parties which are involved in communication.
* These protocols define the type of message being sent or received from any side (either source host or destination host).
* These protocols also define basic syntax of the message being forwarded or retrieved.
* These protocols define the way to send a message and the expected response.
* These protocols also define interaction with the next level.

# Application Layer Protocols

The application layer provides several protocols which allow any software to easily send and receive information and present meaningful data to its users.

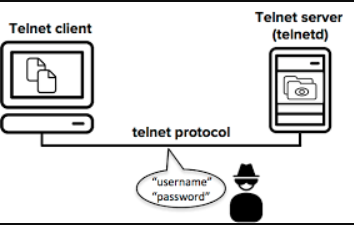
The following are some of the protocols which are provided by the application layer:

## SMTP (Simple Mail Transfer Protocol):



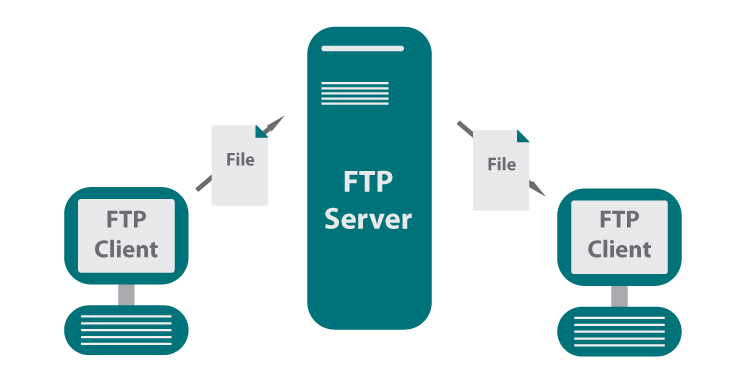
* One of the most popular network service is electronic mail (e-mail).
* The TCP/IP protocol that supports electronic mail on the Internet is called Simple Mail Transfer Protocol (SMTP).
* SMTP transfers messages from senders' mail servers to the recipients' mail servers using TCP connections.
* Users based on e-mail addresses.
* SMTP provides services for mail exchange between users on the same or different computers.
* Following the client/server model:
  + SMTP has two sides: a client side which executes on a sender's mail server, and server side which executes on recipient's mail server.
  + Both the client and server sides of SMTP run on every mail server.
  + When a mail server sends mail (to other mail servers), it acts as an SMTP client.
  + When a mail server receives mail (from other mail servers) it acts as an SMTP server.

## TELNET (Terminal Network):



* TELNET is client-server application that allows a user to log onto remote machine and lets the user to access any application program on a remote computer.
* TELNET uses the NVT (Network Virtual Terminal) system to encode characters on the local system.
* On the server (remote) machine, NVT decodes the characters to a form acceptable to the remote machine.
* TELNET is a protocol that provides a general, bi-directional, eight-bit byte oriented communications facility.
* Many application protocols are built upon the TELNET protocol
* Telnet services are used on PORT 23.

## FTP (File Transfer Protocol):

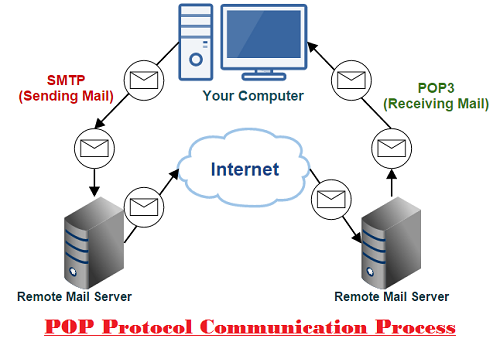


* FTP is the standard mechanism provided by TCP/IP for copying a file from one host to another.
* FTP differs from other client-server applications because it establishes 2 connections between hosts.
* Two connections are: Data Connection and Control Connection.
* Data Connection uses PORT 20 for the purpose and control connection uses PORT 21 for the purpose.
* FTP is built on a client-server architecture and uses separate control and data connections between the client and the server.
* One connection is used for data transfer, the other for control information (commands and responses).
* It transfer data reliably and efficiently.

## Multipurpose Internet Mail Extensions (MIME):

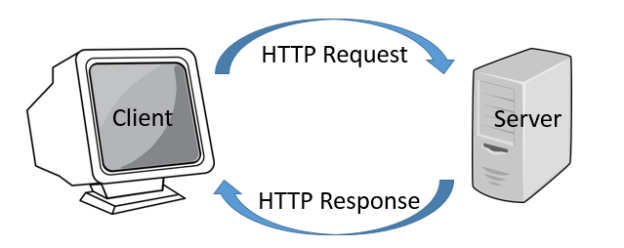
* It is an extension of SMTP that allows the transfer of multimedia messages.
* If binary data is included in a message MIME headers are used to inform the receiving mail agent:
* Content-Transfer-Encoding: Header alerts the receiving user agent that the message body has been ASCII encoded and the type of encoding used.
* Content-Type: Header informs the receiving mail agent about the type of data included in the message.

## POP (Post Office Protocol):



* POP is also called as POP3 protocol.
* This is a protocol used by a mail server in conjunction with SMTP to receive and holds mail for hosts.
* POP3 mail server receives e-mails and filters them into the appropriate user folders. When a user connects to the mail server to retrieve his mail, the messages are downloaded from mail server to the user's hard disk.

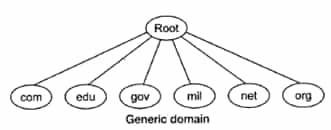
## HTTP (Hypertext Transfer Protocol):



* This is a protocol used mainly to access data on the World Wide Web (www).
* The Hypertext Transfer Protocol (HTTP) the Web's main application-layer protocol although current browsers can access other types of servers
* A repository of information spread all over the world and linked together.
* The HTIP protocol transfer data in the form of plain text, hyper text, audio, and video and so on.
* HTTP utilizes TCP connections to send client requests and server replies.
* It is a synchronous protocol which works by making both persistent and non-persistent connections.

## Domain Name System (DNS):

* To identify an entity, TCP/IP protocol uses the IP address which uniquely identifies the connection of a host to the Internet.
* DNS is a hierarchical system, based on a distributed database, that uses a hierarchy of Name Servers to resolve Internet host names into the corresponding IP addresses required for packet routing by issuing a DNS query to a name server.
* However, people refer to use names instead of address. Therefore, we need a system that can map a name to an address and conversely an address to name.
* In TCP/IP, this is the domain name system.
* DNS in the Internet: DNS is protocol that can be used in different platforms.
* Domain name space is divided into three categories.
  + Generic Domain: The generic domain defines registered hosts according, to their generic behavior. Each node in the tree defines a domain which is an index to the domain name space database



* + Country Domain: The country domain section follows the same format as the generic domain but uses 2 characters country abbreviations (e.g., np for nepal) in place of 3 characters.
  + Inverse Domain: The inverse domain is used to map an address to a name.

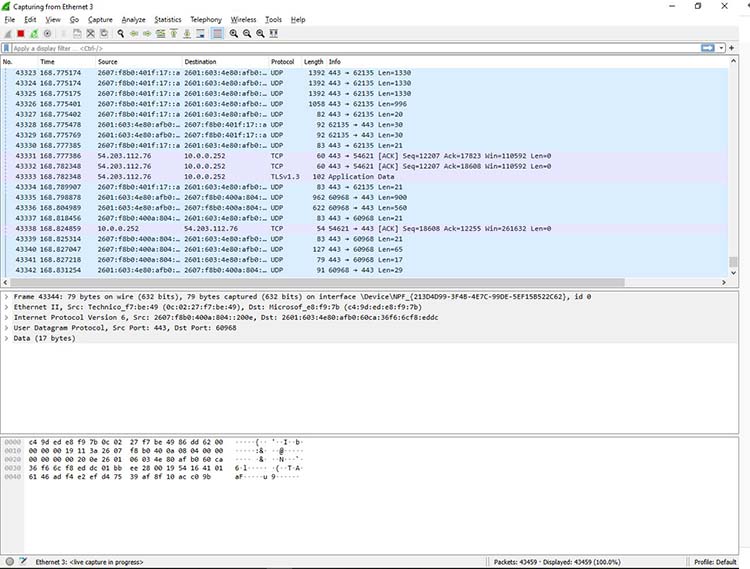
# P4 Operate Wireshark software, and recognize the different windows in the Wireshark interface.

# Wireshark

Wireshark is a network protocol analyzer, or an application that captures packets from a network connection, such as from your computer to your home office or the internet. Packet is the name given to a discrete unit of data in a typical Ethernet network.

Wireshark is the most often-used packet sniffer in the world. Like any other packet sniffer, Wireshark does three things:

* Packet Capture: Wireshark listens to a network connection in real time and then grabs entire streams of traffic – quite possibly tens of thousands of packets at a time.
* Filtering: Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain just the information you need to see.
* Visualization: Wireshark, like any good packet sniffer, allows you to dive right into the very middle of a network packet. It also allows you to visualize entire conversations and network streams.

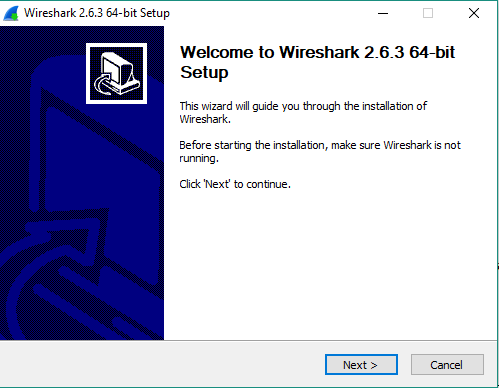


Packet sniffing can be compared to spelunking – going inside a cave and hiking around. Folks who use Wireshark on a network are kind of like those who use flashlights to see what cool things they can find. After all, when using Wireshark on a network connection (or a flashlight in a cave), you’re effectively using a tool to hunt around tunnels and tubes to see what you can see.

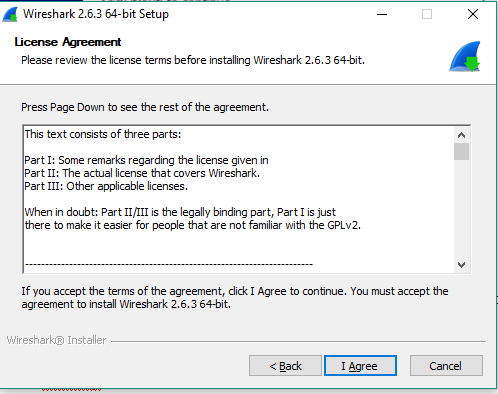
## How to Install Wireshark on Windows

If you’re a Windows operating system user, download the version appropriate for your particular version. If you use Windows 10, for example, you’d grab the 64-bit Windows installer and follow the wizard to install. To install, you’ll need administrator permissions.

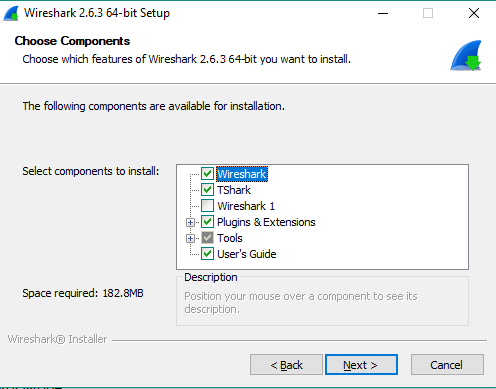
Steps1: Select Next to start the Setup Wizard.

****

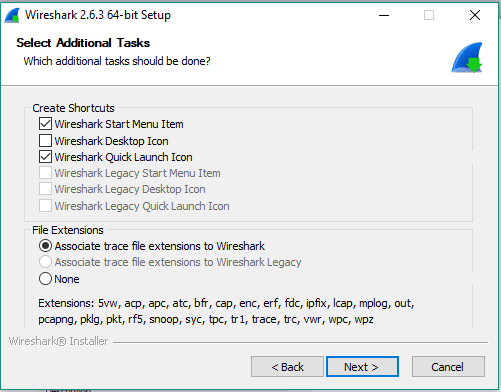
Step2: Review the license agreement. If you agree, select I Agree to continue

****

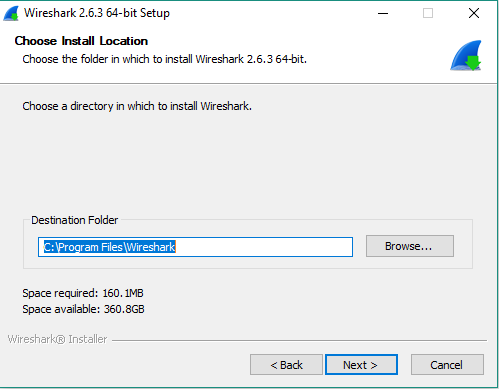
Step3: Select “Next” to accept the default components.

****

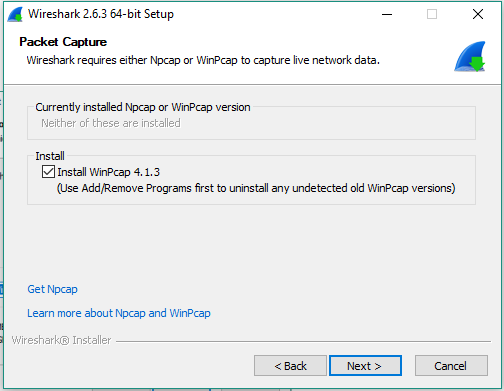
Step4: Select the shortcuts you would like to have created. Leave the file extensions selected. Select Next  to continue

****

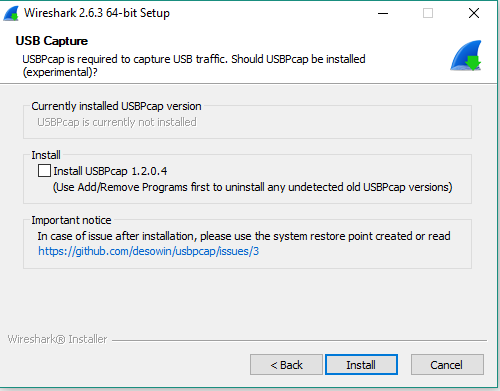
Step5: Select Next  to accept the default install location.

****

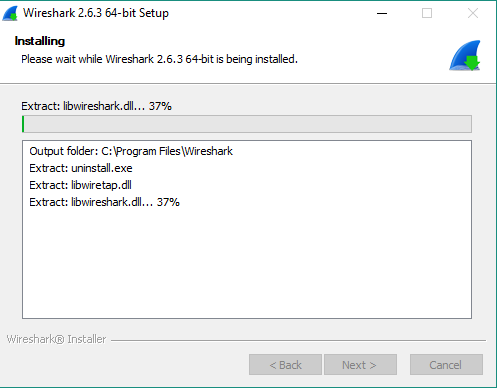
Step6: Select Next  to install WinPcap.

****

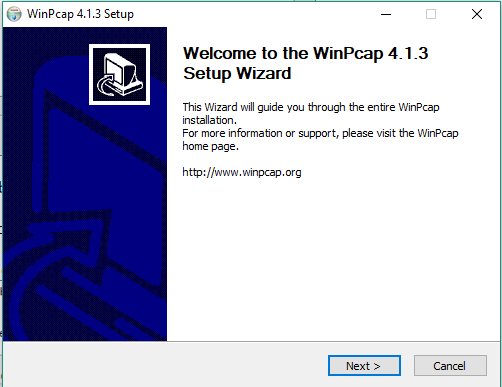
Step7: If you would like to capture USB traffic, install USBPcap as well.

****

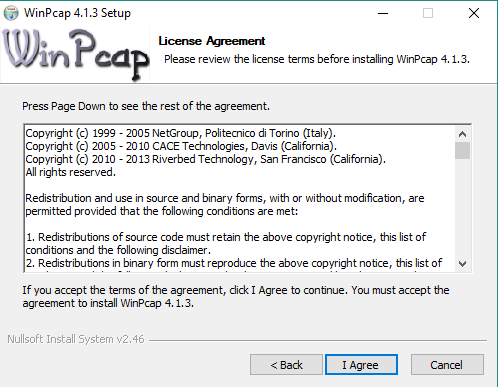
Step8: Select Next  to start the Setup Wizard.

****

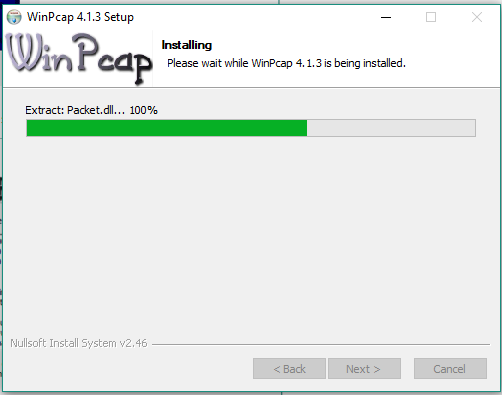
Step9: Select Install to proceed with the installation the requisite software WinPCap. Please note that WinPcap is a mandatory software to ensure Wireshark Packet Analyzer works properly.

****

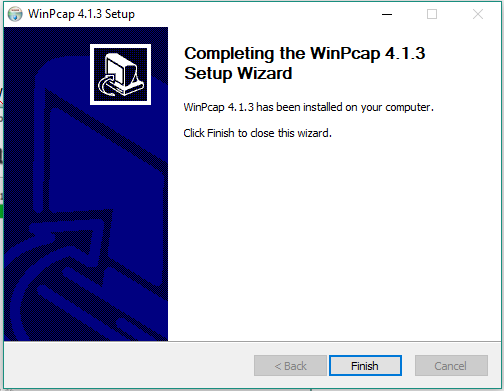
Step10: Review the license agreement. If you agree, select I Agree to continue

****

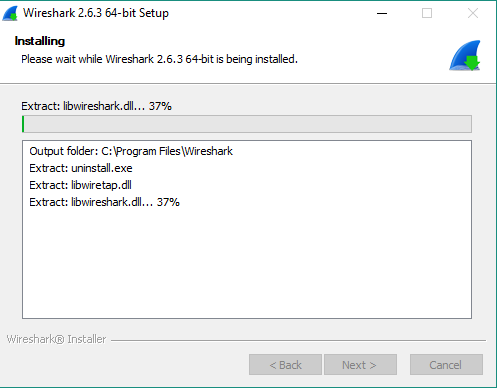
Step11: Installation of WinPcap should start automatically one you agreed and selected next

****

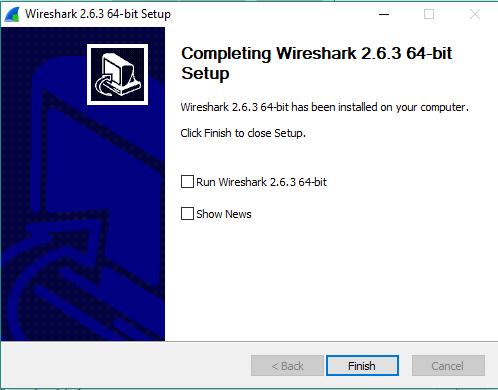
Step12: Select Finish to complete the installation of WinPcap.

****

Step13: Select Next  to continue with the installation of Wireshark

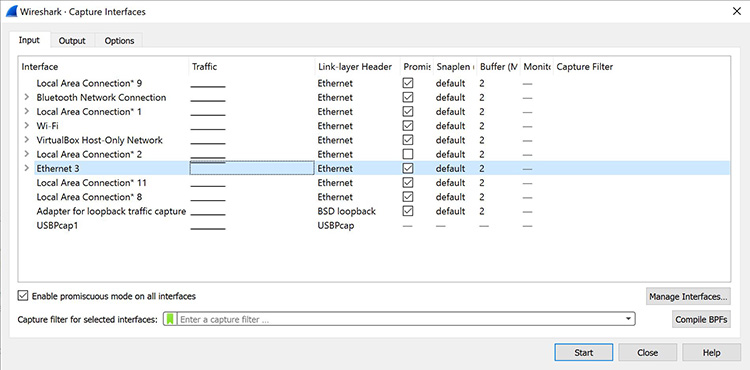
****

Step14: Select Finish to complete the installation of Wireshark. Once installed, you can open the Wireshark and start monitoring network traffic.

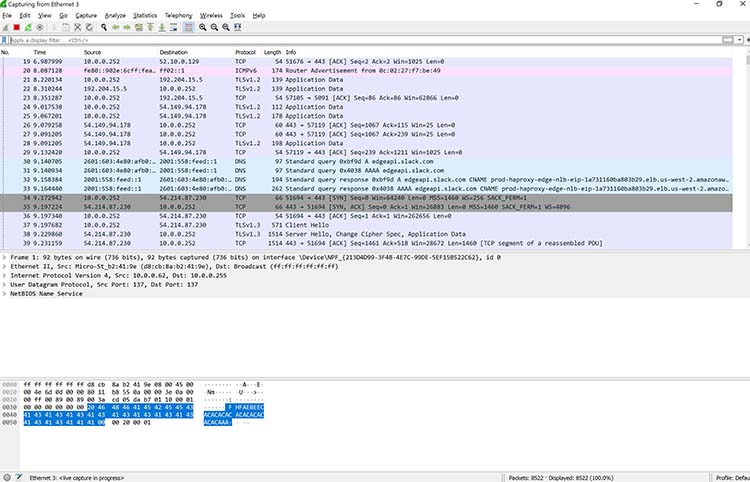
****

## Wireshark Interface and Operation

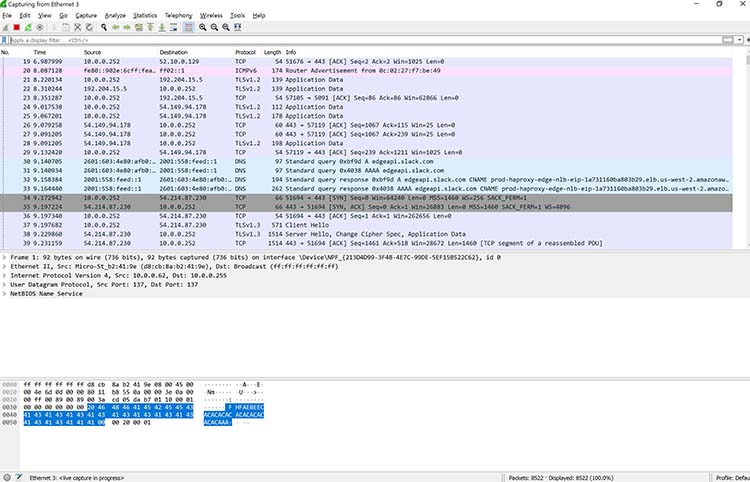
Below image is the interface of Wireshark which appears when you first open Wireshark. This is also the start Screen.



As soon as you click the interface’s name, you’ll see the packets start to appear in real time. Wireshark captures each packet sent to or from your system.



Click the red “Stop” button near the top left corner of the window when you want to stop capturing traffic.



## How to Install Wireshark on Linux

If you have a Linux system, you’d install Wireshark using the following sequence (notice that you’ll need to have root permissions):

$ sudo apt-get install wireshark

$ sudo dpkg-reconfigure wireshark-common

$ sudo usermod -a -G wireshark $USER

$ newgrp wireshark

Once you have completed the above steps, you then log out and log back in, and then start Wireshark:

$ wireshark &

# Conclusion

In this job, I examined the basic principles and functions of many application layer protocols, as well as the communication process between clients and servers in different application protocols. In the next section, I'll have been through Wireshark's idea, function, and benefits and drawbacks. Last but not least, as requested by this assignment, I created documentation for Wireshark installation and basic usage.

# Part3

**Create a client/server model in a Linux system with User Datagram Protocol (UDP), Transmission Control Protocol (TCP)** and **Application Layer protocols**

# Introduction

In the assigned work, I will describe the Linux operating system. In this section, I will discuss the Linux system's brief history as well as its benefits and drawbacks. Under this work, I will also develop a UDP system and a TCP system in Linux. I'll be utilizing C programming for this socket programming.

# Linux

Linux, like Windows and MacOS, is an open-source operating system. It is no longer only an operating system; it is now also a platform for running PCs, servers, and embedded systems. Because it is open source and modular, it has a wide range of distributions and modifications. The kernel is an essential component of the Linux operating system.

Process scheduling, application scheduling, basic peripheral devices, file system, and other services are all managed by the Linux system. Compared to other operating systems like Windows and macOS, Linux has a number of benefits. As a result, it's found in practically every industry, from automobiles to household appliances, cellphones, and servers (supercomputers).

## History of Linux

In 1991, Linus Torvalds a student at the University Of Helsinki, Finland, thought to have a freely available academic version of Unix started writing its own code. Later this project became the Linux kernel. He wrote this program especially for his own PC as he wanted to use Unix 386 Intel computer but couldn't afford it. He did it on MINIX using GNU C compiler. GNU C compiler is still the main choice to compile Linux code but other compilers are also used like Intel C compiler. He started it just for fun but ended up with such a large project. Firstly he wanted to name it as 'Freax' but later it became 'Linux'.

He published the Linux kernel under his own license and was restricted to use as commercially. Linux uses most of its tools from GNU software and are under GNU copyright. In 1992, he released the kernel under GNU General Public License.

## Advantages of Linux operating system

**Open source**: Linux is an open-source OS that means anyone can see the source code and change it according to his needs. You can freely install Linux on many computers without getting paid license. If we compare this with windows or mac then they are paid operating systems. You have to get license of windows and mac to use on your machine.

**No anti-virus software needed**: In Linux, you do not need anti-virus software to be installed on your PC. Linux has fewer chances to be affected with virus. The reason for strong virus protection is that Linux has large number of open source developers which keeps an eye on virus-related stuff. If any source code needs to be updated then it is done in no time.

**Text editors**: Linux has a vast range of text editors available. If you are a programmer then you can pick any of free software packages like visual studio code, Vim, Atom etc. Most of text editors are freely available and you can use it without any issue.

**Powerful command prompt**: Command prompt in Linux is very advanced and if you are developer then you can perform most of your work using the command-line interface. You can install different repositories and packages through the command-line interface.

**No reboot needed**: If you are windows user then you have seen system reboot while you install/uninstall any software or rebooting when the system becomes slow. But in case of Linux, you do not need to reboot your system in such cases.

**Low system specifications**: If you have an old computer that has low specification then you can still run Linux. Linux has different distributions that are available for all types of computers e.g. large scale computers, servers, Pc etc.

**Good at multitasking**: If you want to do some batch works like printing a large file or downloading large file then you can concurrently perform other tasks like typing or coding any program. Linux is good in doing such multitasking and your system will not slow down.

**Less disk space needed**: If you have limited disk space then you can still run Linux. You do not need extra disk space for running Linux for a longer time.

**File** **formats**: Linux supports a large number of file formats. So you have to not worry if any file format does not run on Linux. You can install different software packages for specific file format and it will work fine.

## Disadvantages of Linux operating system

**Hardware** **drivers**: One of the issues that most Linux users face is that some hardware drivers are not available for Linux. Hardware companies prefer to make drivers for windows or mac because they have more users as compared to Linux.

**Learning** **curve**: Getting started with windows is easy for beginners but learning Linux is difficult. You have to learn about the command-line interface and searching for new software is also little bit difficult. If you face any problem in the operating system then finding solution is problematic. There are fewer experts for Linux as compared to windows and mac.

**Software** **alternative**: Take an example of Photoshop which is a popular graphic editing tool. Photoshop is available for windows but is not present in Linux. There are other photo editing tools but Photoshop is a more powerful tool than others. MS office is another example which is not available for Linux users.

**Games**: Most of the games are made for windows but not Linux. As windows platform is widely used so game developers have more interest in windows.

Examples of Linux operating system

Some of Linux distributions are below:

* Ubuntu
* Debian
* Fedora
* Linux mint
* Zorin OS
* Elementary OS
* Manjaro Linux

# P5 Create a UDP system in Linux.

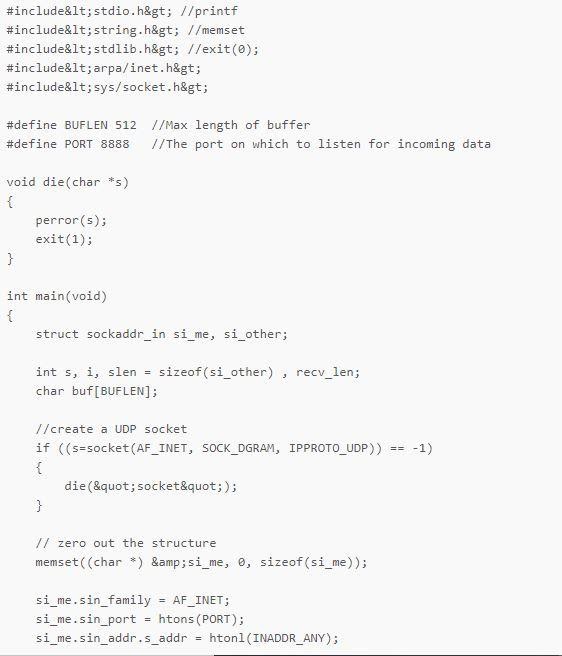
UDP system in linux

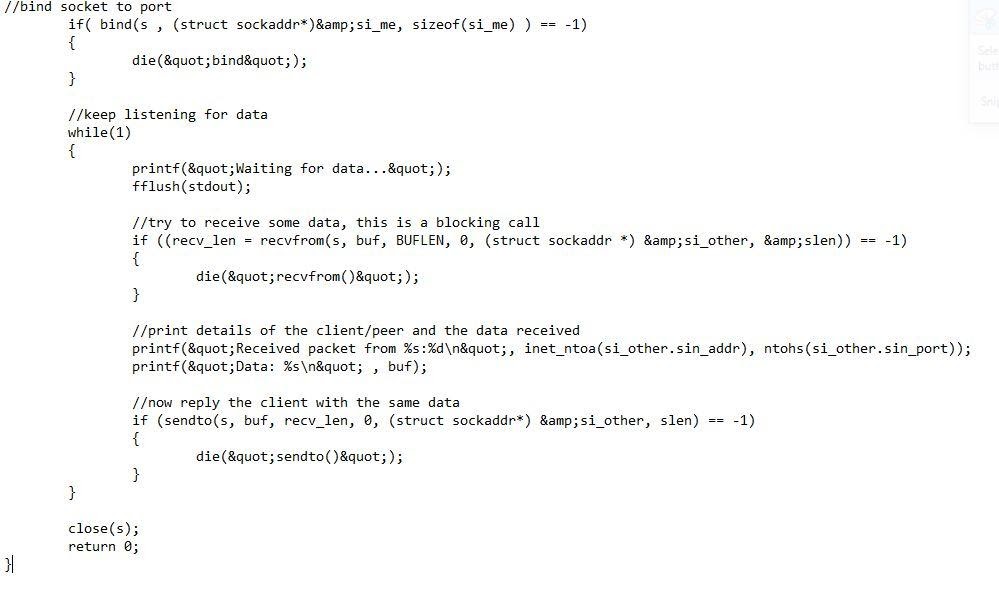
UDP sockets, also known as Datagram sockets, vary from TCP sockets in several respects. UDP sockets are not connection orientated, which is the most significant distinction. A UDP server does not accept connections, and a UDP client does not connect to the server, to put it another way. The server will bind and then receive data directly, whereas the client will transmit data directly.

The client-server paradigm in a Linux system employing User Datagram Protocol (UDP) and C code is shown below:

## Server

First make a very simple ECHO server with UDP socket. The flow of the code would be socket() -> bind() -> recvfrom() -> sendto().

****

****

Run the above code by doing a gcc server.c && ./a.out at the terminal. Then it will show waiting for data like this:

**https://lh6.googleusercontent.com/O2vQfsJtXLs6x6RWoywIU7VwnkegLB89VSn9hctldW62sHXMUXkp2Zpf2fp1hbMip09joXsj03qyzMxSt5EjzLh4W7oM9EI2P93pPh9BdtxE2Vb7guhfGlDdv1Egn1h0NHdGd6wu**

Open a new terminal and use netcat to connect to this udp server, then transmit some data. The server will transmit the same data back. We're using the ncat command from the nmap package over here.

****

We had to use netcat because the ordinary telnet command does not support udp protocol. The -u option of netcat specifies udp protocol.

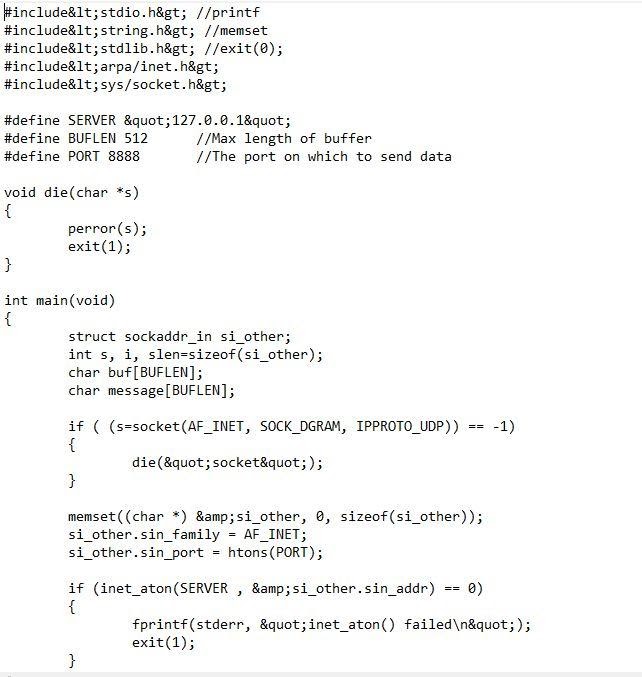
The netstat command can be used to check if the udp port is open or not.

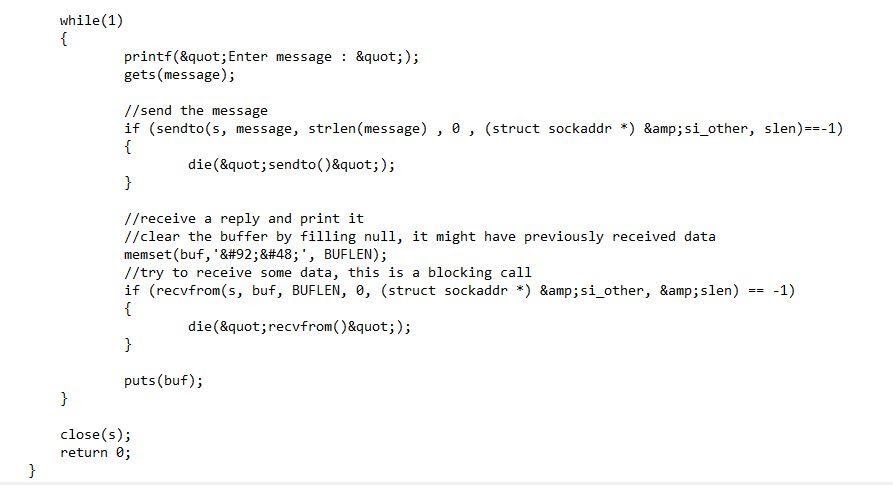
****

The \*:8888 entry of output. Thats our server program. The entry that has localhost:8888 in "Foreign Address" column, indicates some client connected to it, which is netcat over here.

## Client

client code in c

****

****

Note: Compile the code same as did for the server code

Run the above program and it will ask for some message, whatever message the client sends to server, the same comes back as it is and is echoed.



DNS and other protocols utilize UDP sockets. The main goal of utilizing UDP is to send little quantities of data in situations where dependability isn't a major concern. UDP is also used for multicasting and broadcasting.

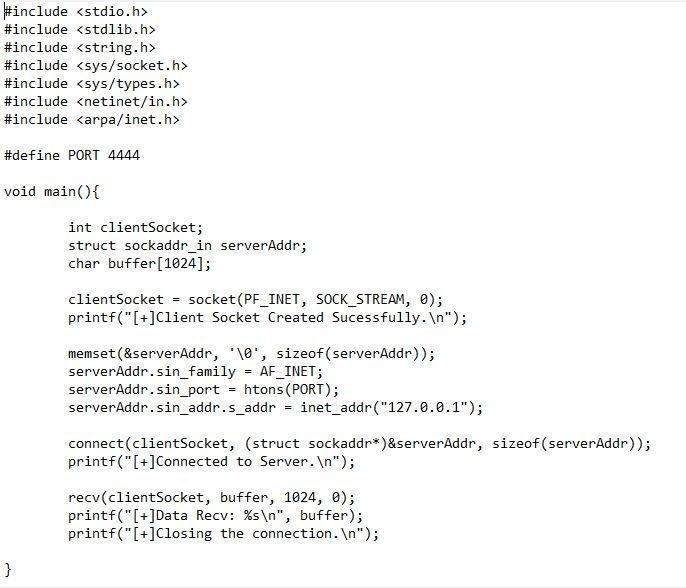
# P6 Create a TCP system in Linux

**Tcp in linux**

## Server



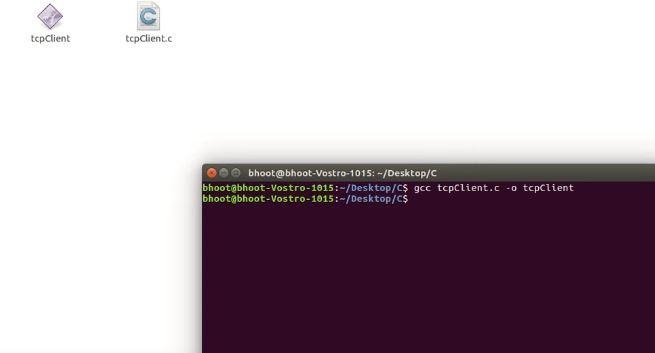
Client

****

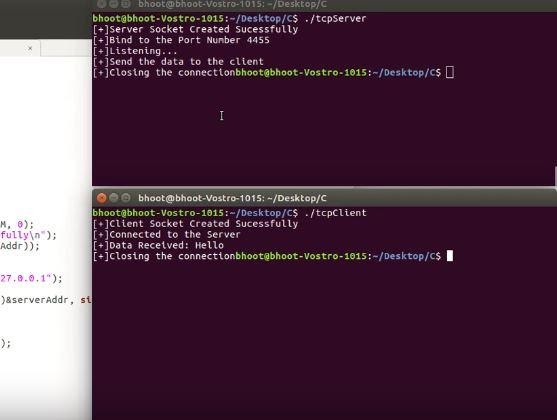
## Compilation of server code



## Compilation of client code



## Connection between server and client



**Hence the Connection between Client and server created successfully.**

# Conclusion

In this job, I discussed the Linux operating system, including its brief history as well as its benefits and drawbacks. In Linux, I developed the UDP system. In addition, I developed the Linux UDP system. I used C programming to develop the socket programming.

# References

<https://www.geeksforgeeks.org/client-server-model/?ref=lbp>

<https://www.tutorialspoint.com/operating-systems-client-server-communication>

<https://www.techopedia.com/definition/438/clientserver-architecture>

<https://www.tutorialspoint.com/process-vs-parent-process-vs-child-process>

<https://www.geeksforgeeks.org/application-layer-in-osi-model/>

<https://byjusexamprep.com/application-layer-protocols-dns-smtp-pop-ftp-http-i>

<https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it>

<https://www.itrelease.com/2019/09/advantages-and-disadvantages-of-linux-operating-system/>

<https://www.javatpoint.com/advantages-of-linux>