**Error! Unknown document property name.**

**Error! Unknown document property name.**

High Level Design & Low Level Design

The purpose of this document is to provide with a template for documenting both HLD & LLD.

**Document Control :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | | | |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |

[1. Introduction 3](#_Toc368912248)

[1.1. Intended Audience 3](#_Toc368912249)

[1.2. Acronyms/Abbreviations 3](#_Toc368912250)

[1.3. Project Purpose 3](#_Toc368912251)

[1.4. Key Project Objectives 3](#_Toc368912252)

[1.5. Project Scope and Limitation 3](#_Toc368912253)

[1.5.1. In Scope 3](#_Toc368912254)

[1.5.2. Out of scope 3](#_Toc368912255)

[1.6. Functional Overview 3](#_Toc368912256)

[1.7. Assumptions, Dependencies & Constraints 3](#_Toc368912257)

[1.8. Risks 3](#_Toc368912258)

[2. Design Overview 3](#_Toc368912259)

[2.1. Design Objectives 3](#_Toc368912260)

[2.1.1. Recommended Architecture 3](#_Toc368912261)

[2.2. Architectural Strategies 3](#_Toc368912262)

[2.2.1. Design Alternative 3](#_Toc368912263)

[2.2.2. Reuse of Existing Common Services/Utilities 3](#_Toc368912264)

[2.2.3. Creation of New Common Services/Utilities 3](#_Toc368912265)

[2.2.4. User Interface Paradigms 3](#_Toc368912266)

[2.2.5. System Interface Paradigms 3](#_Toc368912267)

[2.2.6. Error Detection / Exceptional Handling 3](#_Toc368912268)

[2.2.7. Memory Management 3](#_Toc368912269)

[2.2.8. Performance 3](#_Toc368912270)

[2.2.9. Security 3](#_Toc368912271)

[2.2.10. Concurrency and Synchronization 3](#_Toc368912272)

[2.2.11. Housekeeping and Maintenance 3](#_Toc368912273)

[3. System Architecture 3](#_Toc368912274)

[3.1. System Architecture Diagram. (Not Necessary) 3](#_Toc368912275)

[3.2. System Use-Cases 3](#_Toc368912276)

[3.3. Subsystem Architecture 3](#_Toc368912277)

[3.4. System Interfaces 3](#_Toc368912278)

[3.4.1. Internal Interfaces 3](#_Toc368912279)

[3.4.2. External Interfaces 3](#_Toc368912280)

[4. Detailed System Design 3](#_Toc368912281)

[4.1. Key Entities 3](#_Toc368912282)

[4.2. Detailed-Level Database Design 3](#_Toc368912283)

[4.2.1. Data Mapping Information 3](#_Toc368912284)

[4.2.2. Data Conversion 3](#_Toc368912285)

[4.3. Archival and retention requirements 3](#_Toc368912286)

[4.4. Disaster and Failure Recovery 3](#_Toc368912287)

[4.5. Business Process workflow 3](#_Toc368912288)

[4.6. Business Process Modeling and Management (as applicable) 3](#_Toc368912289)

[4.7. Business Logic 3](#_Toc368912290)

[4.8. Variables 3](#_Toc368912291)

[4.9. Activity / Class Diagrams (as applicable) 3](#_Toc368912292)

[4.10. Data Migration 3](#_Toc368912293)

[4.10.1. Architectural Representation 3](#_Toc368912294)

[4.10.2. Architectural Goals and Constraints 3](#_Toc368912295)

[4.10.3. Logical View 3](#_Toc368912296)

[4.10.4. Architecturally Significant Design Packages 3](#_Toc368912297)

[4.10.5. Data model 3](#_Toc368912298)

[4.10.6. Deployment View 3](#_Toc368912299)

[5. Environment Description 3](#_Toc368912300)

[5.1. Time Zone Support 3](#_Toc368912301)

[5.2. Language Support 3](#_Toc368912302)

[5.3. User Desktop Requirements 3](#_Toc368912303)

[5.4. Server-Side Requirements 3](#_Toc368912304)

[5.4.1. Deployment Considerations 3](#_Toc368912305)

[5.4.2. Application Server Disk Space 3](#_Toc368912306)

[5.4.3. Database Server Disk Space 3](#_Toc368912307)

[5.4.4. Integration Requirements 3](#_Toc368912308)

[5.4.5. Jobs 3](#_Toc368912309)

[5.4.6. Network 3](#_Toc368912310)

[5.4.7. Others 3](#_Toc368912311)

[5.5. Configuration 3](#_Toc368912312)

[5.5.1. Operating System 3](#_Toc368912313)

[5.5.2. Database 3](#_Toc368912314)

[5.5.3. Network 3](#_Toc368912315)

[5.5.4. Desktop 3](#_Toc368912316)

[6. References 3](#_Toc368912317)

[7. Appendix 3](#_Toc368912318)

# 

# Introduction

**A port scanner** tool can scan network ports to identify their connectivity details and display the results to the user in real time. This tool plays an important role in improving network security as it provides complete visibility and control over network ports. Since ports play a crucial part in enabling network communication, the visibility offered by network port scanners helps you identify unauthorized port access, malicious services running in them, and block suspicious ports.

This project is using CLI to perform. Port scanners also play an important role in analyzing network resource usage. It scans your network ports and displays the number of available ports, used ports, and device details for accessing a port such as its IP and MAC addresses. This helps you better track resource usage and enhance capacity planning. Ports are communication endpoints that connect network devices. The general protocols used for port scanning are TCP (transmission control protocol) and UDP (user datagram protocol). They are both data transmission methods for the internet but have different mechanisms.

## Intended Audience

|  |  |
| --- | --- |
| Network administrators |  |
|  |  |

## Acronyms/Abbreviations

|  |  |
| --- | --- |
| CLI | Command Line Interface |
| netstat | Network Statistics |
|  |  |
|  |  |

## Project Purpose

The purpose of port scanning is to acquire information from the servers to which the ports are attached. Port scanning is a network reconnaissance technique designed to identify which ports are open on acomputer. This can enable the scanner to identify the applications running on the system as certain programs listen to ports and react to traffic in certain ways.

## Key Project Objectives

* To develop an application that can be used to dig out detailed information about a particular IP address connection.
* To develop an application to debug the network services.
* To develop an application for monitoring network connections both incoming and outgoing as well as viewing routing tables, interface statistics, etc.

## Project Scope and Limitation

Our project aims to identify any security vulnerabilities on a particular network and to troubleshoot their network-related problems and determine network traffic performance. The application should be as simple as possible so that it can be configured even by a non-technical person.

There is a limitation also that port scans take significant time depending on the number of ports scanned as well as the power of the system testing the networks. This project is to target to the particular device on which it will be used.

### In Scope

* It may help collecting all internet connection’s details for a particular IP address.
* User can choose any specific function for a particular detail of IP address.
* User can get routing table, active socket connection’s details for UDP and TCP, status of the port and pulling and viewing network statistics sorted by protocol.

### Out of scope

* In this application user needs to choose one functionality at a time.
* Port scanning happens to be one of the most popular tactics used by attackers when searching for a vulnerable server to breach.
* To utilize port scanning for securing network from any kind of attack.
* Easy to operate and understand by the user.

## Functional Overview

To create this project “netstat” command is used. Basically, the network statistics (netstat) command is a networking tool used for troubleshooting and configuration that can also serve as a monitoring tool for connections over the network. Both incoming and outgoing connections, routing tables, port listening, and usage statistics are common uses for this command.

## Assumptions, Dependencies & Constraints

With the use of c language and Linux we can create an application where port scanning can take place. This application can be used to pull and view various network related information such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships etc. A user can enter which operation he/she wants to perform, whether the user wants to view the kernel routing tables or wants to get the summary statistics for each protocol or listening and established ports.

## Risks

Open ports are a security risk if services running on these ports are misconfigured, vulnerable or unpatched.

# Design Overview

The project is designed by partitioning the program into functions. These functions are called based on the user choice inputs and his/her requirements. A specific CLI tool played major role in this project i.e. netstat.

## Design Objectives

The following are the function that can be used by the user:

1. listing all ports and connections of their protocol statistics
2. list the Routing Table
3. list all Active TCP Socket Connection
4. list all Active UDP Socket Connection
5. list all Internet connection which are using applications like Firefox / Chrome
6. list all the internet connection where state is LISTEN/ESTABLISHED for TCP/UDP protocols
7. exit from program

Each case has a member function that is declared inside the block. The member function is defined inside the program. Based on the user’s inputs each case calls the function that it is mapped with. When case 8 is chosen by the user then the program.

### Recommended Architecture

The recommended system architecture is as follows.

* 1GB RAM
* 500MHz Processor
* 120GB HDD CPU
* Internet connection
* Linux operating system / CLI

## Architectural Strategies

A port scanner is an application designed to probe a server or host for open ports. Such an application may be used by administrators to verify security policies of their networks and by attackers to identify network services running on a host and exploit vulnerabilities.

A **port scan** is a process that sends client requests to a range of server port addresses on a host, with the goal of finding an active port; this is not a nefarious process in and of itself. The majority of uses of a port scan are not attacks, but rather simple probes to determine services available on a remote machine.

### Design Alternative

NA

### Reuse of Existing Common Services/Utilities

The project is reusing the existing common services or utilities. We are using netstat commands repeatedly.

### Creation of New Common Services/Utilities

A text file is being created that is storing results of the project.

### User Interface Paradigms

This system allows users to easily monitoring their networks using internet connection. Just by entering a choice, user can get details about their IP address. We have made a very simple and easy to understand interface. All the essential details about ports of every protocol can be viewed by pressing a key.

### System Interface Paradigms

Good design is good business. If the system has good interface and it satisfies user requirements, then the software can reach new heights. The port scanner offers basic ports tracking and scanning functionality. The user can scan available options and can also save their details on a GUI text file.

### Error Detection / Exceptional Handling

In this system some errors may occur if system requirements are not fulfilled. For example: If user wants to see all the internet connections which are using the browser in the system and the browser is not running on the host, then It will not provide any details about ports of browser. So, this can be handled by running the browser in background.

### Memory Management

NA

### Performance

NA

### Security

NA

### Concurrency and Synchronization

The results of port scanning needs to be appended in a single text file.

### Housekeeping and Maintenance

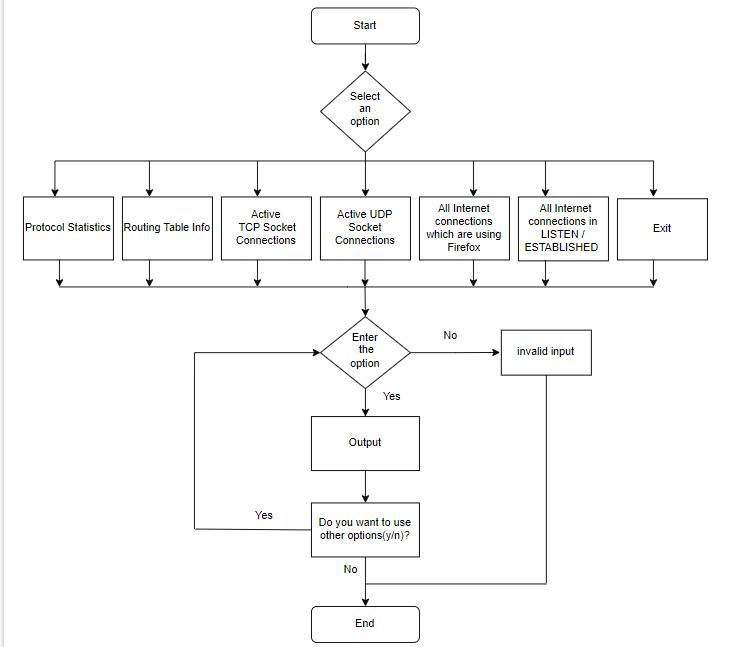
NA

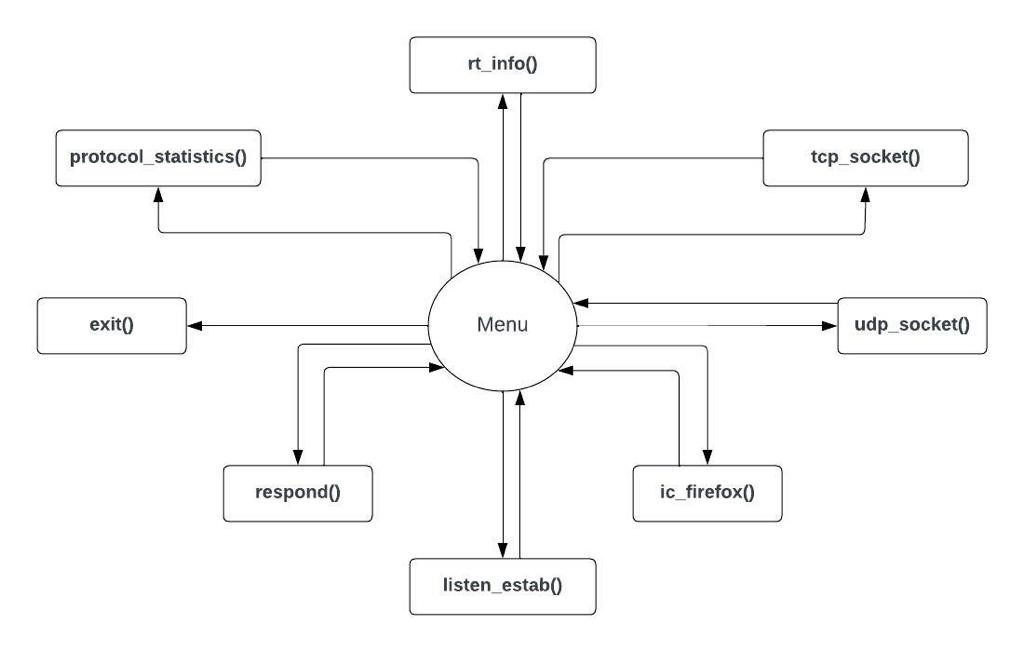
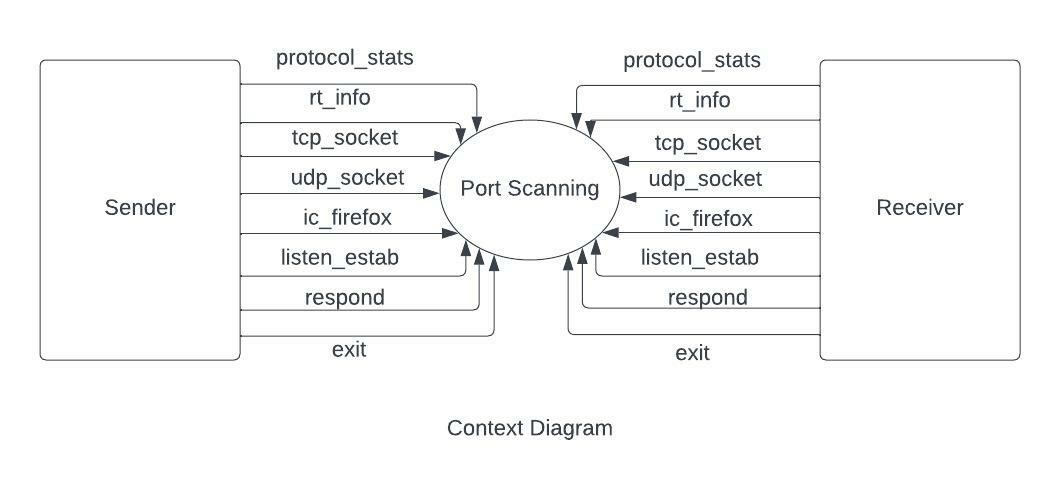
# System Architecture

This project has one main function that is calling a menu function to perform different functions. In the menu functions if the user wants to get details of protocol statistics, then that particular function gets called. This menu function provides 7 functionalities; each functionality is being performed by calling the related function. If user wants to exit from the scanner, he/she will press 7 to exit from the program.

After each function there is respond function that is being used to take user input to perform more operation or to exit from program at that stage. Functions are repeatedly calling to perform in this project. The major responsibility is carrying by netstat commands and those CLI commands are used in C program with system function. System function is executing the netstat commands.

## System Architecture Diagram. (Not Necessary)





## System Use-Cases

The main purpose of a use case diagram is to show what system functions are performed for which actors.

## Subsystem Architecture

Inside every function, the netstat commands are used. A port scanner will send a TCP (Transmission Control Protocol) or UDP (User Datagram Protocol) network packet and inquire the port about their status. The response can be as the following states:

* Open – An open port implies that when someone tries to connect to that port on the server, the server might respond in some way.
* Closed – As the name suggests, a closed port indicates that the server isn’t responding to any connections.
* Filtered – A filtered port indicates that a firewall or some antivirus/anti-malware program is blocking the port to avoid certain connections.

## System Interfaces

[A good design ensures that all the System’s Interfaces are well documented. List out the details of all the System Interfaces, interface Design along with diagrammatic representation if possible with details of flow, frequency etc.]

### Internal Interfaces

The internal interfaces comprise interfaces through which the system interacts with the clients, through which it provides them services i.e; netstat tool and internet connection.

### External Interfaces

The external interface comprises interfaces through which the users interact with the system i.e; Linux operating system and Internet connection.

# Detailed System Design

The detailed design of this system is as explained below:

* Using 2-character array data-structures of pre-defined size to store the values and an integer value for each case. We are using total 8 functions in this project. We are using one array to store output of command and another one to store the result of the program into a text file. A menu function which has 8 functions we are calling that by the integer value which we are used in the switch case.
* In the first case we are calling the function protocol\_statistics in which we are using the Linux command of netstat for protocol statistics details of our device and copy that executed part in the text file using strcpy. Strcpy is used for coping the string.
* In the second case we are calling the function rt\_info in which we are using the Linux command of netstat for routing table information on our device and copy that executed part in the text file using strcpy.
* In the third case we are calling the function tcp\_socket in which we are using the Linux command of netstat for active TCP socket connections on our device and copy that executed part in the text file using strcpy.
* In the fourth case we are calling the function udp\_socket in which we are using the Linux command of netstat for active UDP socket connections on our device and copy that executed part in the text file using strcpy.
* In the fifth case we are calling the function ic\_firefox in which we are using the Linux command of netstat for connections that are using browser on our device and copy that executed part in the text file using strcpy.
* In the sixth case we are calling the function listen\_estab in which we are using the Linux command of netstat for all the internet connections which are connected to our device and copy that executed part in the text file using strcpy.
* The seventh functions that is used in this system is respond function. Using this function if else loop is implemented for continuing the program or exiting the program.
* The eighth function is sleep function.

## Key Entities

NA

## Detailed-Level Database Design

NA

### Data Mapping Information

NA

### Data Conversion

NA

## Archival and retention requirements

NA

## Disaster and Failure Recovery

NA

## Business Process workflow

NA

## Business Process Modeling and Management (as applicable)

NA

## Business Logic

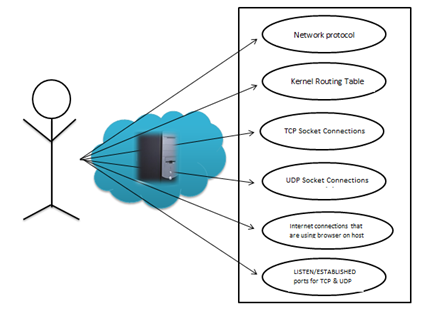
NA

## Variables

The project has used few variable :

* cmd[] and out[] - two character type arrays
* ch and response - two integers

## Activity / Class Diagrams (as applicable)



## Data Migration

NA

### Architectural Representation

### Architectural Goals and Constraints

### Logical View

The main function is starting the port scanner and calling menu function to perform. Menu function is having other functions that will provide desired results.

* menu(): - user will input his/her choice for available options listed below: -
* protocol statistics (): - it can show us protocol statistical details of our device like the current number of established connections using each protocol.
* rt\_info (): - it can show us routing table information of our device the information necessary to forward a packet along the best path toward its destination.
* tcp\_socket (): - it can show us active TCP socket connections on our device.
* udp\_socket (): -it can show us active UDP socket connections on our device.
* ic\_firefox (): - it can show connections that are using the browser on our device.
* listen\_estab (): - it can show us the state of all the internet connections of our device.
* exit (): - it can come out of the program.

Listed above, all the functions are calling respond function.

* respond(): - it will ask user to continue the program further or to exit from the program.

### Architecturally Significant Design Packages

NA

### Data model

**Legacy system data model**

**Proposed system data model**

**Interface data model**

NA

### Deployment View

In order to run this project, user needs an internet connection and command line interface. If there is no internet connection, user will not be unable to perform scanning.

# Environment Description

The Environment used is the Linux Operating System

## Time Zone Support

It will support time zone as per Indian standard time (IST) in (GMT +5:30) and UST standard.

## Language Support

C language is used in this project. It was created in the 1970s by Dennis Ritchie, and remains very widely used and influential.

## User Desktop Requirements

Operating System: Linux or Windows with Linux Subsystem

Processor: Minimum 1GHz and More

Hard Drive: 30GB and More

Memory(RAM): Minimum 1GB and More

## Server-Side Requirements

Uninterrupted Internet connectivity

### Deployment Considerations

500Mhz Processor

120GB HDD CPU

4GB RAM

Network connectivity

### Application Server Disk Space

Disk space – Minimum 100GB

### Database Server Disk Space

NA

### Integration Requirements

NA

### Jobs

Copying command into a variable using string function - strcpy

### Network

Internet connectivity is required.

### Others

NA

## Configuration

NA

### Operating System

Linux Operating System

Processor: Minimum 1GHz and More

Hard Drive: 32GB and More

Memory(RAM): Minimum 1GB and More

### Database

NA

### Network

Only stable internet connectivity is required.

### Desktop

[Describe the desktop configuration requirements here. Details of Application software required and other configurations.]

# References

<https://www.rapidfiretools.com/blog/2021/03/22/port-scanning/>

<https://en.wikipedia.org/wiki/Port_scanner>

<https://www.redhat.com/sysadmin/netstat>

# Appendix

[This section should provide a complete list of all documents or links on the Internet where related material can be found.]

**Change Log**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QMS Template Version Control (Maintained by QA)** | | | | | |
|  |  |  |  |  |  |
| **Date** | **Version** | **Author** | | **Description** | |
| 22-12-2022 | 0.1 | Group - D | | Initial Version | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |