Table of Contents

List of Figures

Chapter-1

INTRODUCTION

A computer network consists of network elements like nodes, links, routers, IP etc. The network administrator (manager) is the one who manages the network. Administrator faces problems in managing the network. Adding a new network element requires configuration and maintenance. When the network is functioning some of the network elements may go down. It creates obstacles in network functions. Various nodes use the network for variable times; hence for accounting the information about the use of network by every node must be maintained. The administrator or some special user must be able to find out performance of a node. If data in the network is confidential then it must be protected from outside world. Also there are different categories defined for the user and the data should be accessible to them as per their privilege level.

There are different problems that the network/system administrator has to face in their jobs like problem in IP address assignment, information about blocked ports, interface information, destination reachability and communication to the server. The job of handling those task is very difficult and requires extensive knowledge if they are to be handled manually through commands. As the networking tasks have been complex and for the better work performance on defined time administrator are in search of highly reliable and accurate application.

Reliability, accuracy, security and configurability are the major factor to be considered in the network as well as in system environment. The network applications are also designed to meet the aforementioned requirements and to operate specific tasks. Network application like this NETWORK DIAGNOSTIC TOOL which runs on cross platform are highly applicable application which meets the most frequent requirements of the administrator. This NETWORK DIAGNOSTIC TOOL checks the application running platform and executes on that environment. The programming logic in the application calls

the corresponding operating system's command, manipulates the command and displays the result in graphical form leaving the actual operation of the application abstract.

Chapter-2

BACKGROUND

Network Diagnostic Tool is an application program developed in Java programming language with the use of network application programming interface (API). This application program is developed using Net Bean 5.4 and JDK version 1.8.

The purpose of Network Diagnostic Tool (NETWORK DIAGNOSTIC TOOL) application is to provide a GUI interface to handle the common tasks. The application program is so developed that it only prompts the user to feed their input and which is manipulated by the program and the result is displayed in the well formatted form. This eventually reduces the burden for the application user to remember the commands to be executed and execution procedure.

NETWORK DIAGNOSTIC TOOL consist of simple application like ping, Port Scan, IP Configuration, Network Diagnosis and LAN Chat. The ping program helps to determine whether the destination IP or host is reachable or not by sending ICMP echo request. The Port Scan program finds out the status of port within the given range of specified host or IP address. IP Configuration program helps to configure the DHCP, DNS and default gateway in both IPv4 and IPv6. Similarly, Network Interface Info program helps to find out the interface name, host name, MAC address, virtual addressing, multicast and the status of the interface. The LAN Chat helps to communicate with the server by the client machine on the same local area network.

In NETWORK DIAGNOSTIC TOOL input from user is taken in the text field where only valid inputs are considered for further processing. With the bound of protocol used users are allowed to feed any valid inputs. All the protocols and necessary requirements proposed by RFC to run the application are not violated in this program. Corresponding operating system's commands are called in the back end. Due to this users are not necessarily required to know the internal execution of program and respective OS commands used in the program. This has increases the reliability and robustness of the system.

Chapter-3

PROBLEM IDENTIFICATION

3.1 Problem Statement

Many advanced networking applications and networking infrastructure are developed and available for use. The users of such application must be expert and must have extensive knowledge for handling those applications. Therefore to overcome this problem and make non-technical users as well as highly skilled technical users to handle simple administrative and networking jobs this project has been developed. This project is developed with the consideration that users with little knowledge on networking can be fully benefited as we have tried to make this project simple, interactive and user friendly. Moreover this project has integrated different application modules into single module so network/system administrator who switches among different application can use the same functionality through single module.

Most of the applications available today are platform dependent and generally requires higher processing capability and memory of the system. Users have to use different application program for same purpose on different platform which certainly arise the question of robustness of the program. So NETWORK DIAGNOSTIC TOOL project has tried to overcome this shortcoming since this project is platform neutral, memory efficient and run on machines with lower processing power. Additionally there has been a huge investment of money for hiring individual administrator for handling simple jobs. NETWORK DIAGNOSTIC TOOL helps to overcome this situation as semi technical staff can easily handle and perform simple administrative task through the use of this application program.

3.2 Objective

Network Diagnostic Tool has wider application domain and can be used extensively. NETWORK DIAGNOSTIC TOOL has many objectives so that it can be the milestone in network administration. Some of the objectives are mention below:

- To help non-technical users to have better control over their network without any prior knowledge of corresponding OS commands.
- To execute the simple administrative task effectively and efficiently at short time.
- To switch the application user from classical CLI modes to highly interactive GUI mode.
- To integrate the individual application program into single module for faster access and execution.
- To apply the academic knowledge gained throughout the semester into the practically applicable field.

3.3 Scope and Limitation

3.3.1 Scope

System scope defines the application area of the proposed system and the end users. Thus scope helps to determine the prerequisites required to start the project and criteria the product should have to achieve the desired output. The scope of our proposed system are as follows:

- This application program can be used by system/network administrator as well as general user to handle the simple administrative task easily as per their requirement in their organization.

- With the bound of protocol used users are allowed to feed any valid inputs. Therefore there is no illegal case and every organization and institution can use this application program.

- Corresponding operating system's commands are called in the back end. This increases the reliability and robustness of the system. This helps the general users to use this application program to handle their task without prior knowledge of executing commands.

- All the protocols and necessary requirements proposed by RFC to run the application will not be violated. Therefore this application can highly use by network/system administrator for standard output.

- This program can be used by academic institute to teach their students about the network administration.

- This application is applicable for organization that deploy security constraints like blocking some specific ports, assigning static IP address, defining default gateway and DNS server.

3.3.2 Limitations

The limitations that exist in this Network Diagnostic Tool are:

- For exploration and efficient handling of this NETWORK DIAGNOSTIC TOOL users should have the knowledge of corresponding operating system and commands used to carry out the proposed task.
- This application program calls the corresponding OS command, therefore for the security purpose if certain commands are blocked from execution this program does not work correctly.
- This project has included only few features which make user to rely upon other application program to carry out desired work.
- NETWORK DIAGNOSTIC TOOL abstract the internal execution of the program leaving curious user in dilemma.

Chapter-4

REQUIREMENT ANALYSIS AND FEASIBILITY STUDY

4.1 Requirement Analysis

Requirements analysis, also called requirements engineering, is the process of determining needs or condition to meet user expectations for a new or modified product. These requirements must be quantifiable, relevant and detailed.

Requirements analysis involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users, avoidance of feature creep and documentation of all aspects of the project development process from start to finish. Requirements analysis is a team effort that demands a combination of hardware software and human factors engineering expertise as well as skills in dealing with the system.

4.1.1 Software Requirements

Operating System	Windows XP Professional, Vista, Windows 5/8 , Linux
Environment	JAVA
Prerequisites	JDK 1.8

4.1.2 Hardware Requirements

	Minimum	**Recommended**
Processor	400 MHz	1 GHz
O.S.	WIN 9x	Windows >XP ,Linux(latest)
Memory	256 Mb	1 Gb
Hard disk	4.3 Gb	10 Gb

4.2 Feasibility Study

The feasibility study is an evaluation and analysis of the potential of the proposed project which is based on extensive investigation and research to support the process of decision making. Feasibility studies aim to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture, opportunities and threats present in the environment, the resources required to carry through, and ultimately the prospects for success. In its simplest terms, the two criteria to judge feasibility are cost required and value to be attained. So we have carried out the feasibility analysis for our project on the basis of time, cost, and operation which are described in short in below section.

4.2.1 Operational Feasibility

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. Our project is also operationally feasible as it outputs 100% accurate result in accordance to the user input.

4.2.2 Technical Feasibility

The technical feasibility is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of the proposed system. The project that we have developed is technically feasible. As we are experienced in programming language JAVA so technical manpower has no problem. As we have efficient and powerful working PC’s and Laptops for programming and implementation.

4.2.3 Schedule Feasibility

Schedule feasibility means estimating how long the system will take to develop, and can be completed in a given time period. Schedule feasibility is a measure of how reasonable the project timetable is. As far concerned our project was started in July 2013 and is almost completed in December. So it took 5 and half months to complete the project. The deadline of the project is at last week of December. So our project is feasible according to the time as it completed within its deadline.

Chapter-5

SYSTEM DESIGN AND DEVELOPMENT

Network Diagnostic Tool is an application program developed in Java programming language. The graphical user interface is user friendly and highly interactive. In NETWORK DIAGNOSTIC TOOL different application are integrated and managed in the tabbed pane so that user can access the required application arranged in tabbed pane. Whenever user clicks the tabbed pane the corresponding application is opened and GUI for that application prompts the user to feed the input. When the user feeds the input then the corresponding program is executed and the result is displayed accordingly.

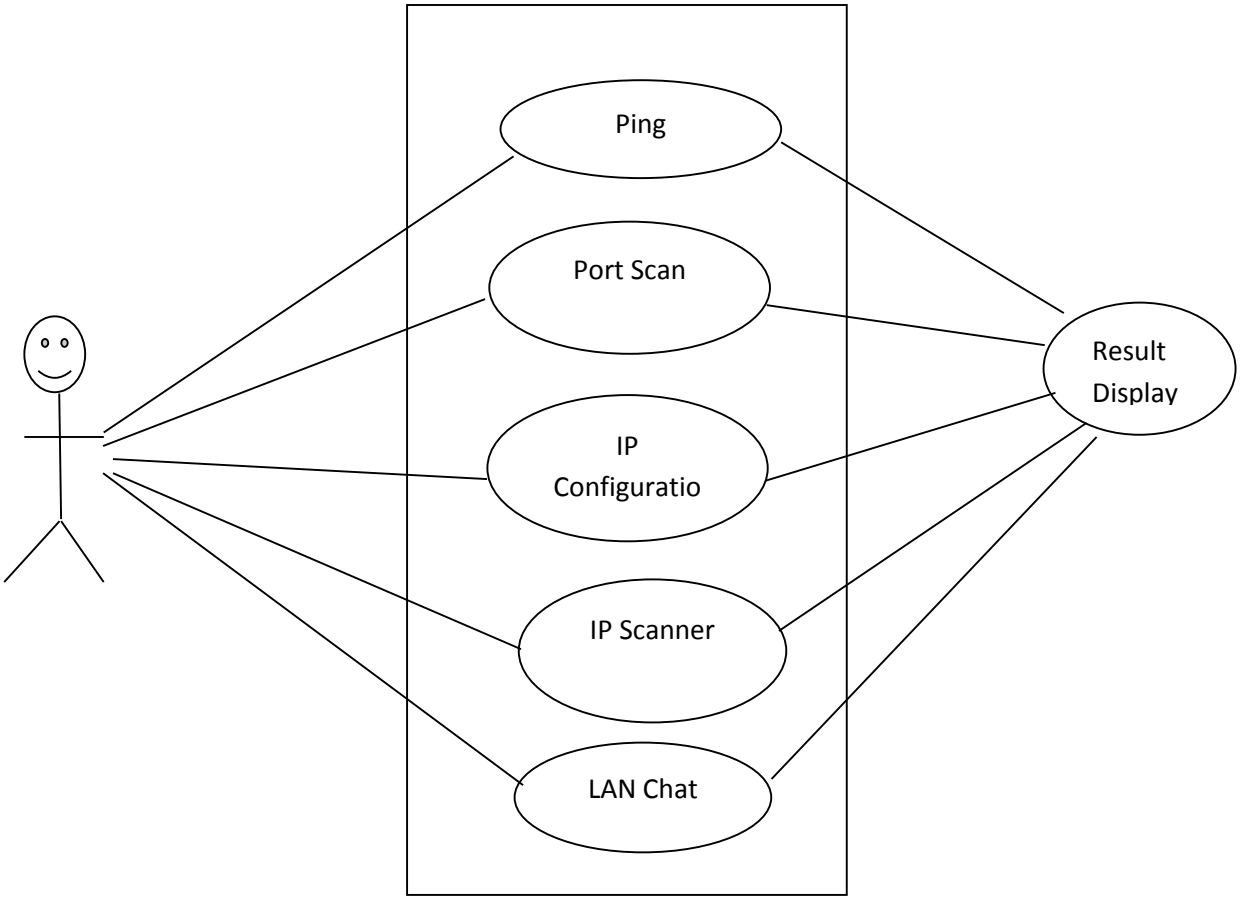

Fig 1: Use Case Diagram for Network Diagnostic Tool

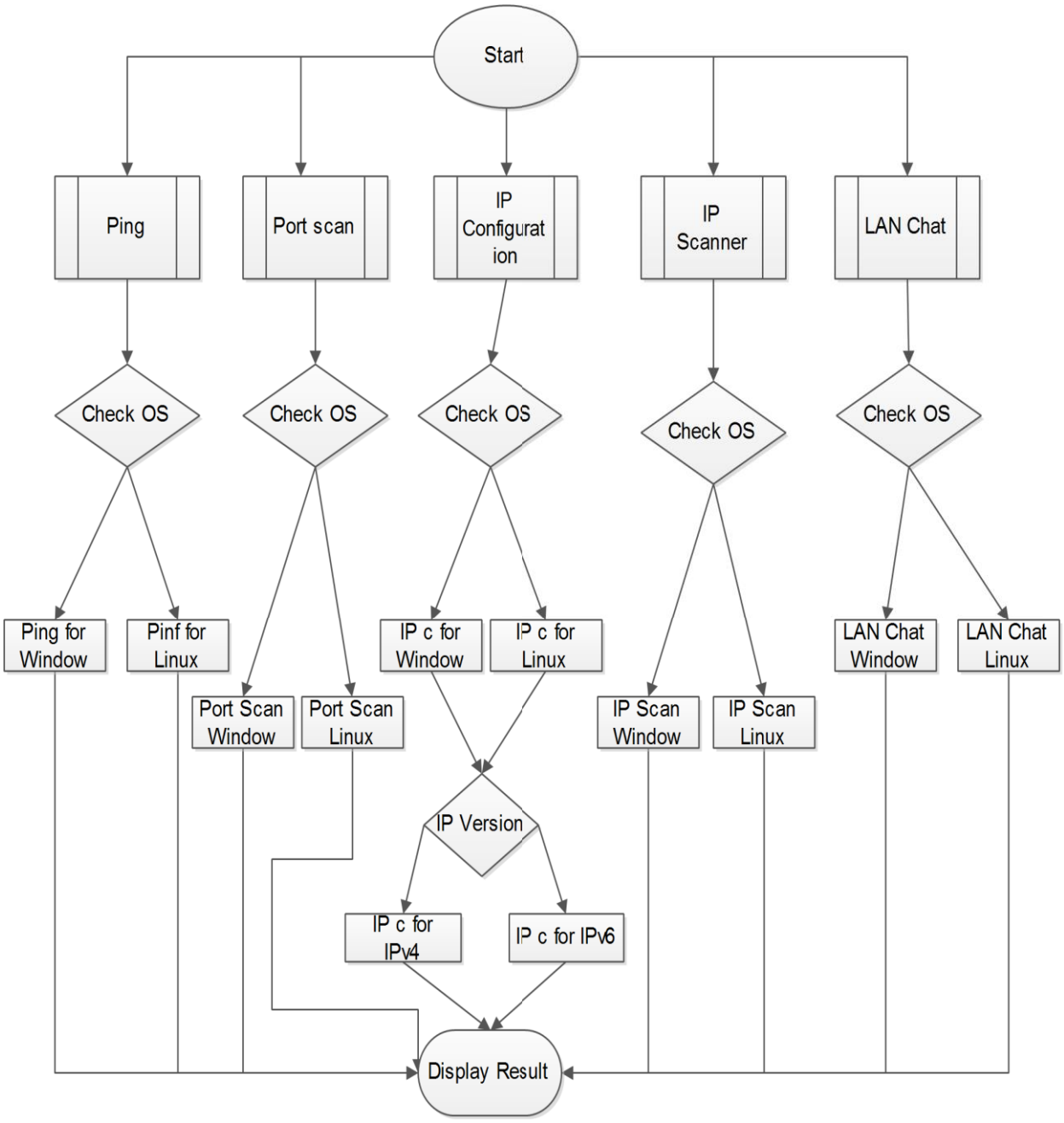

Fig 2: Complete flow-chart of Network Diagnostic Tool

5.1 PING

Ping is a basic program that allows a user to determine whether the particular IP address is reachable or not [1]. Ping is used diagnostically to ensure that a host computer you are trying to reach is actually operating.

Ping command sends an Internet Control Message Protocol (ICMP) ECHO_REQUEST to obtain an ICMP ECHO_RESPONSE from a host or gateway. If the host is operational and on the network, it responds to the echo. Each echo request contains an Internet Protocol (IP) and ICMP header, followed by a ping PID and a time Val structure, and enough bytes to fill out the packet. The default is to continuously send echo requests until an Interrupt is received. [7]

The ping command sends one datagram per second and prints one line of output for every response received. The ping command calculates round-trip times and packet loss statistics, and displays a brief summary on completion. The ping command completes when the program times out or on receipt of a SIGINT signal. The host parameter is either a valid host name or Internet address.

5.1.1 Application Area

Ping program can be used by system/network administrator as well as normal user to determine

- TCP/IP is functioning or not.
- Router is running or not.
- Host is reachable or not.
- Determining the status of the network and various foreign hosts.
- Tracking and isolating hardware and software problems.
- Testing, measuring, and managing networks.

5.1.2 Working Mechanism of Ping Program

Below is the detailed flow chart of Ping Program

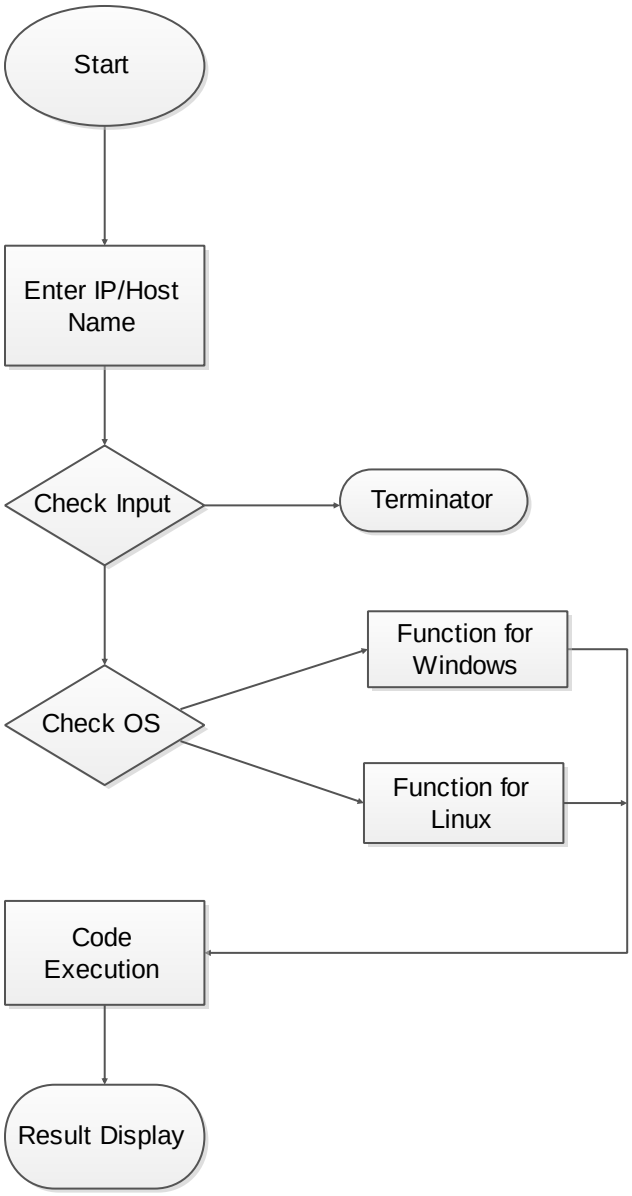

Fig 3: Flow Chart of Ping program

Ping program runs on both Windows and Unix/Linux system. CheckOs function of java programming language determines whether the working platform is Windows or Unix or other. If the platform is Window then for ping the following function is called:

```
private void pingFunctionforWindows() throws IOException {
        pingCmd = pingTextBox.getText();
        System.out.println(pingCmd);
        if ((pingCmd.startsWith("ping "))||
          (pingCmd.startsWith("PING ")))   {
            r = Runtime.getRuntime();
            p = r.exec(pingCmd);

            BufferedReader in = new BufferedReader(
                    new InputStreamReader(p.getInputStream()));
            timeStart = System.currentTimeMillis();
            while ((inputLine = in.readLine()) != null) {
                Result.append(inputLine + "\n");
            }
            timeEnd = System.currentTimeMillis();
            timeGap = timeEnd - timeStart;
            timetoshow = String.valueOf(timeGap);
            Result.append("Total time taken = " + timetoshow + "
                miliseconds");
            Result.append("\n");

        }
        else {
            Result.setText("Ping command not matched");
        }
    }
```

Similarly if the platform is UNIXthen following function is called:

```
private void pingFunctionforLinux() throws IOException {
        pingCmd = pingTextBox.getText();
        System.out.println(pingCmd);
        if ((pingCmd.startsWith("ping "))){
            r = Runtime.getRuntime();
```

```
            p = r.exec(pingCmd);

            BufferedReader in = new BufferedReader(
                    new InputStreamReader(p.getInputStream()));
            timeStart = System.currentTimeMillis();
        while (((inputLine = in.readLine()) != null) && count <= 10) {
            Result.append(inputLine + "\n");
            count++;
            }
            timeEnd = System.currentTimeMillis();
            timeGap = timeEnd - timeStart;
            timetoshow = String.valueOf(timeGap);
                    Result.append("Total time taken = " + timetoshow + "   miliseconds");
            Result.append("\n");
        }
        else {
            Result.setText("Ping command not matched");
        }
    }
}
```

The ping program calls the corresponding operating system ping command, calculates the total time taken for response and generates the result in highly interactive graphical form.

5.1.3 Output

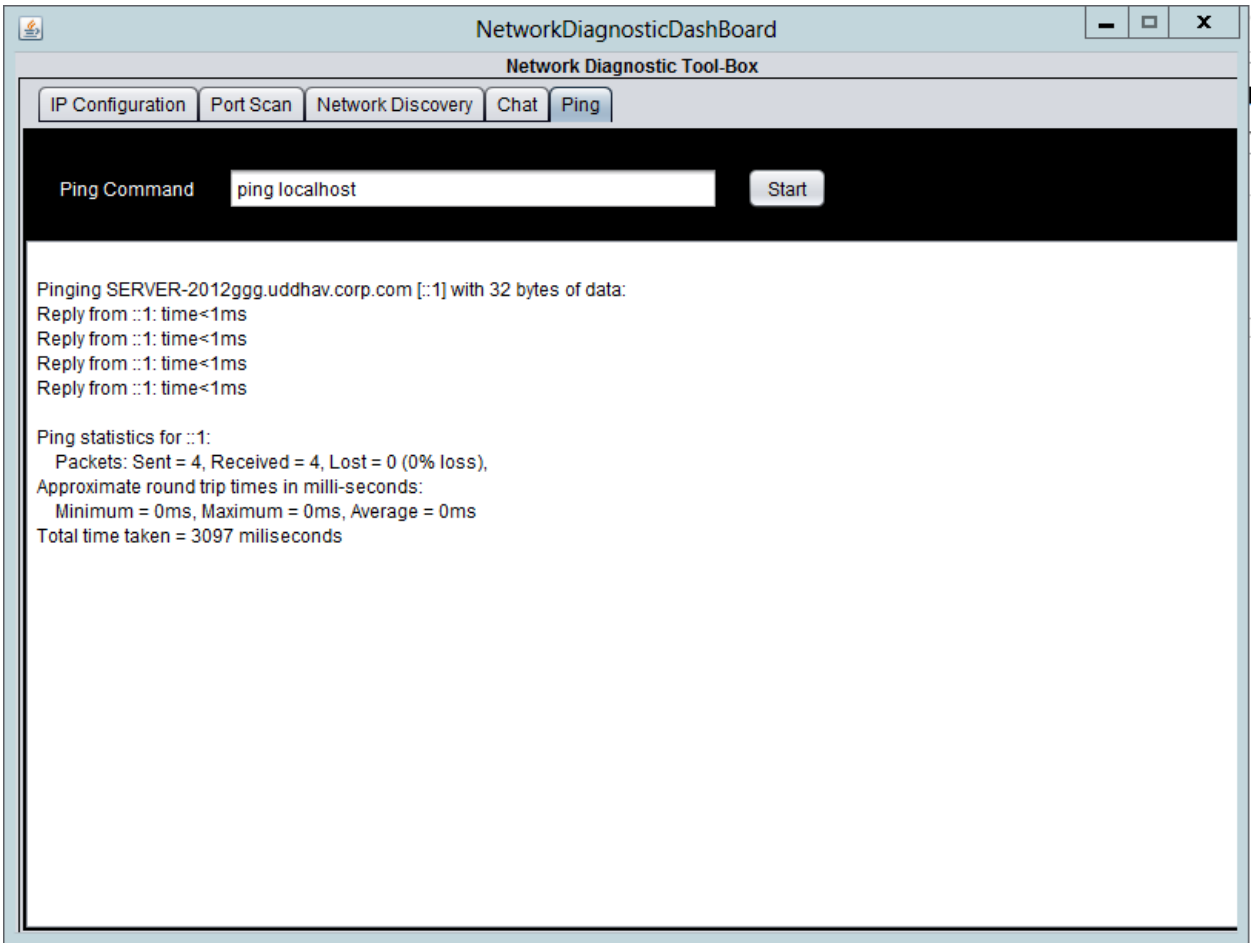

Fig 4: Output of Ping program

5.2 PORT SCAN

The purpose of ports is to uniquely identify different applications or processes running on a single computer and thereby enable them to share a single physical connection to a packet-switched network like the Internet. [3]

The term "port scanning" refers to the technique or attempt to identify open ports and available services on a network host. It can be used by security professionals to audit

network computers for likely vulnerabilities. It can also be used by attackers to check victim hosts for possible exploits. There are 65536 distinct and usable port numbers.

Port Scanning is one of the most popular reconnaissance techniques system/network administrator use to discover services they can break into. All machines connected to a Local Area Network (LAN) or Internet run many services that listen at well-known and not so well known ports. A port scan helps the system/network administrator find which ports are available and what service might be listing to a port. Essentially, a port scan consists of sending a message to each port, one at a time. The kind of response received indicates whether the port is used and can therefore be probed further for weakness.

The Network Diagnostic Tool's port scan program is the best tool for checking for open ports on Windows or UNIX systems. The Port Scan utility in this program determines whether the port associated with specific IP address is listening or not.

5.2.1 Application Area

Port scan is generally used by system/network administrator for determining open ports to verify security policies. System administrators are constantly being advised to check their systems for open ports and services that might be running that are either unintended or unnecessary.

5.2.2 Working Mechanism of Port Scan program

Port scan probes a computer system running TCP/IP to determine which TCP and UDP ports are open and listening, which indicates all of the services that this system is offering to other TCP/IP hosts. As an example, we would expect an e-mail server to be listening on the SMTP and POP3 ports, and a Web server to be listening on the HTTP, and perhaps the SSL/HTTPS, ports. For good or for bad, however, most systems have many more open ports than intended. In this program we first create a Socket on specified port range limit and check whether the port is open or not on a local Linux/Unix or Windows system [5]

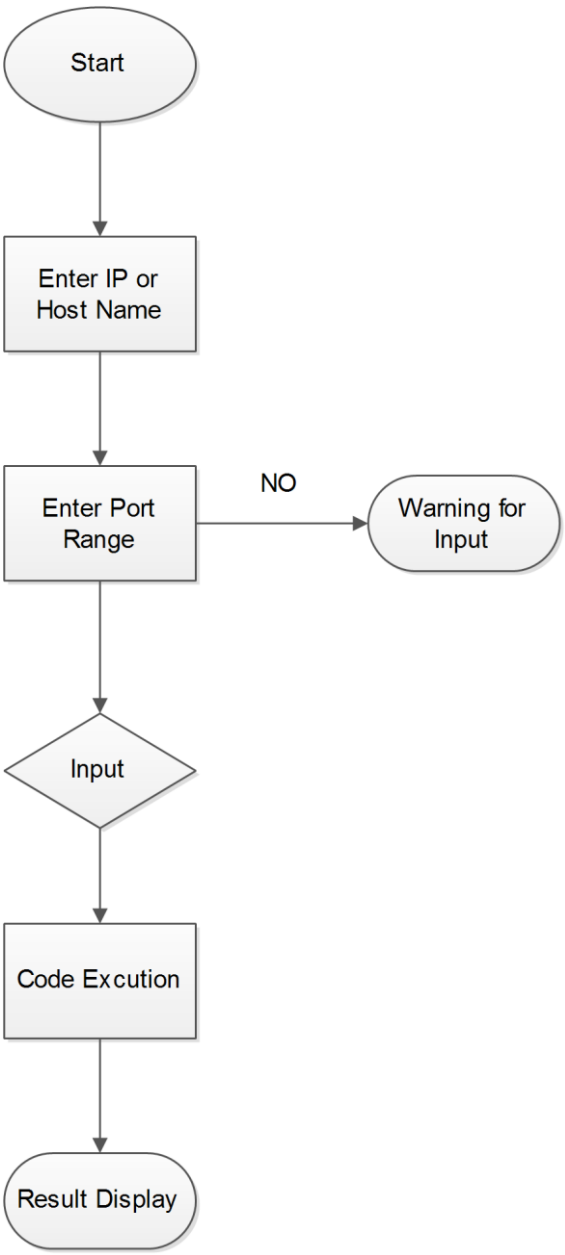

Fig 5: Flow chart for port scan program

```
private void scan(InetAddress remote) {

    int port = 0;
    String hostname = remote.getHostName();
    String resulta = "";
    String resultaaa = "";
```

```
            for (port = Integer.parseInt(textbox1.getText()); port <
            Integer.parseInt(textbox2.getText())+ 1; port++) {
        try {
            Socket s = new Socket(remote, port);
                        resulta = "Server is listening on port " + port + " of " + hostname;
            resultaaa += resulta + "\n";
            Result.setText(resultaaa);
            s.close();
        } catch (IOException ex) {
                        resulta = "Server is not listening on port " + port + " of " + hostname;
            resultaaa += resulta + "\n";
            Result.setText(resultaaa);
        }}
      throw new UnsupportedOperationException("Not supported yet.");
   }
```

In function scan remote is an object of InetAddress class. The getHostName( ) method returns a String that contains the name of the host with the IP address represented by the InetAddress object. For port number specified on the range with user input on textbox1 and textbox2 the code inside the try block is executed. A new Socket object s is constructed with parameter remote (i.e. an object of InetAddress) and port. If the host is listening on the specified port then the result is displayed on TextArea labeled as Result as ‘Server is listening on port’ of given port ‘of’ given host, otherwise ‘Server is not listening on port’ of given port ‘of’ given host is displayed.

5.2.3 Output

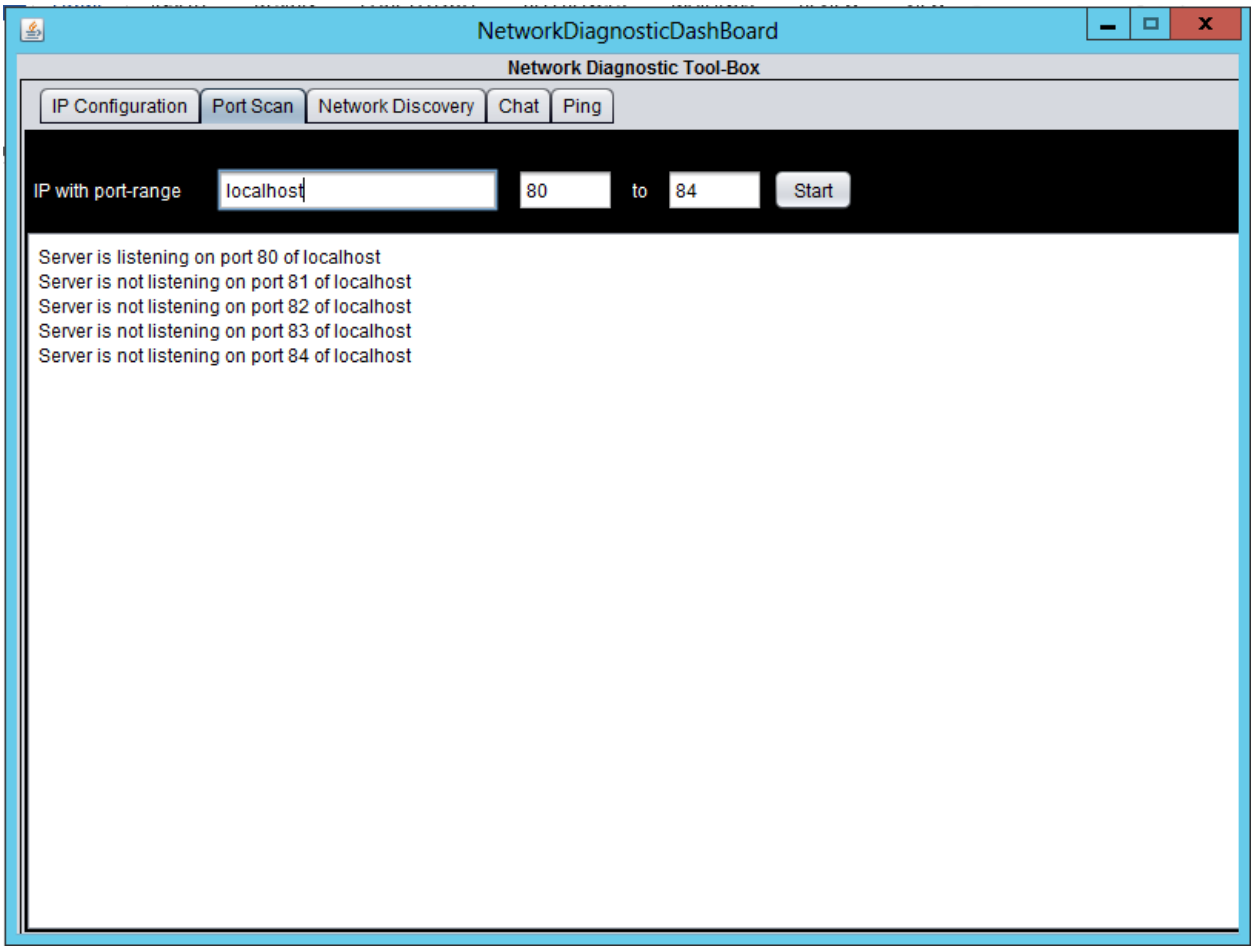

Fig 6: Output of Port Scan program

5.3 IP CONFIGURATION

IP Addresses

An IP address is an identifier for a computer or device on a TCP/IP network. Networks using the TCP/IP protocol route messages based on the IP address of the destination. An IP address can be static or dynamic. A static IP address will never change and it is a permanent Internet address. A dynamic IP address is a temporary address that is assigned each time a computer or device accesses the Internet. Two versions of the Internet Protocol (IP) are in use: IP Version 4 and IP Version 6.

IPV4

IPv4 is the most widely used version of the Internet Protocol. It defines IP addresses in a 32-bit format, which looks like 123.123.123.123. Each three-digit section can include a number from 0 to 255, which means the total number of IPv4 addresses available is 4,294,965,296 (2^32).

IPv4 uses 32-bit addresses for Ethernet communication in five classes, named A, B, C, D and E. Classes A, B and C have a different bit length for addressing the network host. Class D addresses are reserved for multicasting, while class E addresses are reserved for future use.

Class A has the addressing format of network.node.node.node and subnet mask 255.0.0.0 or /8. Class A network is defined in the first octet between 0 and 125.

Class B has the addressing format of network.network.node.nodeand subnet mask 255.255.0.0 or /16. Class B network is defined when the first byte is configured from 128 to 191.

Class C has the addressing format of network.network.network.node and subnet mask 255.255.255.0 or /24. Class C network is defined when the first byte is configured from 192 to 223.

IPV6

Internet Protocol version 6 (IPv6) is the latest revision of the Internet Protocol (IP), the communications protocol that provides an identification and location system for computers on networks and routes traffic across the Internet. IPv6 was developed by the Internet Engineering Task Force (IETF) to deal with the long-anticipated problem of IPv4 address exhaustion. IPv6 presents a standardized solution to overcome IPv4's limitations. Because of its 128-bit address length, it can define up to 2,128 addresses.

The most obvious improvement in IPv6 over IPv4 is that IP addresses are lengthened from 32 bits to 128 bits. This extension anticipates considerable future growth of the Internet and provides relief for what was perceived as an impending shortage of network addresses. IPv6 also supports auto-configuration to help correct most of the shortcomings in version 4, and it has integrated security and mobility features.

IPv6 addresses are typically written in this format:

hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:hhhh

Each "hhhh" section consists of a four-digit hexadecimal number, which means each digit can be from 0 to 9 and from A to F. An example IPv6 address may look like this:

F504:0000:0000:0000:3458:59A2:D08B:4320

Because IPv6 addresses are so complex, the new system also adds extra security to computers connected to the Internet. Since there are so may IP address possibilities, it is nearly impossible to guess the IP address of another computer while most computer systems today support IPv6, the new Internet protocol has yet to be fully implemented.

IPv6 features include:

- Supports source and destination addresses that are 128 bits (16 bytes) long.
- Requires IPsec support.

- Uses Flow Label field to identify packet flow for QoS (Quality of Service) handling by router.
- Allows the host to send fragments packets but not routers.
- Doesn't include a checksum in the header.
- Uses a link-local scope all-nodes multicast address.
- Does not require manual configuration or DHCP.
- Uses host address (AAAA) resource records in DNS to map host names to IPv6 addresses.
- Uses pointer (PTR) resource records in the IP6.ARPA DNS domain to map IPv6 addresses to host names.
- Supports a 1280-byte packet size (without fragmentation).
- Moves optional data to IPv6 extension headers.
- Uses Multicast Neighbor Solicitation messages to resolve IP addresses to link-layer addresses.
- Uses Multicast Listener Discovery (MLD) messages to manage membership in local subnet groups.
- Uses ICMPv6 Router Solicitation and Router Advertisement messages to determine the
- IP address of the best default gateway.

DHCP

DHCP (Dynamic Host Configuration Protocol) is a communications protocol that lets network administrators centrally manage and automate the assignment of Internet Protocol (IP) addresses in an organization's network. Using the Internet Protocol, each machine that can connect to the Internet needs a unique IP address, which is assigned

when an Internet connection is created for a specific computer. Without DHCP, the IP address must be entered manually at each computer in an organization and a new IP address must be entered each time a computer moves to a new location on the network. DHCP lets a network administrator supervise and distribute IP addresses from a central point and automatically sends a new IP address when a computer is plugged into a different place in the network. [3]

DHCP uses the concept of a "lease" or amount of time that a given IP address will be valid for a computer. The lease time can vary depending on how long a user is likely to require the Internet connection at a particular location. It's especially useful in education and other environments where users change frequently. Using very short leases, DHCP can dynamically reconfigure networks in which there are more computers than there are available IP addresses. The protocol also supports static addresses for computers that need a permanent IP address, such as Web servers.

DNS

The DNS (Domain Name System) is a mechanism for translating Internet domain and host names to IP addresses. DNS automatically converts the names we type in our Web browser address bar to the IP addresses of Web servers hosting those sites.

DNS implements a distributed database to store this name and address information for all public hosts on the Internet. The DNS database resides on a hierarchy of special database servers. When clients like Web browsers issue requests involving Internet host names, a piece of software called the DNS resolver first contacts a DNS server to determine the server's IP address. If the DNS server does not contain the needed mapping, it will in turn forward the request to a different DNS server at the next higher level in the hierarchy. After potentially several forwarding and delegation messages are sent within the DNS hierarchy, the IP address for the given host eventually arrives at the resolver, that in turn completes the request over Internet Protocol.

DNS additionally includes support for caching requests and for redundancy. Most network operating systems support configuration of primary, secondary, and tertiary DNS servers, each of which can service initial requests from clients.

Gateway

The gateway is the computer that routes the traffic from a workstation to the outside network that is serving the Web pages. In homes, the gateway is the ISP that connects the user to the internet.

In enterprises, the gateway node often acts as a proxy server and a firewall. The gateway is also associated with both a router, which use headers and forwarding tables to determine where packets are sent, and a switch, which provides the actual path for the packet in and out of the gateway.

MTU

The maximum transmission unit (MTU) of a communications protocol of a layer is the size in bytes of the largest protocol data unit that the layer can pass onwards.

Loopback

Loopback refer to the routing of electronic signals, digital data streams, or flows of items back to their originating devices or facilities without intentional processing or modification. This is primarily a means of testing the transmission or transportation infrastructure.

Interface

Hardware interfaces exist in computing systems between many of the components such as the various buses, storage devices, other I/O devices, etc. A hardware interface is described by the mechanical, electrical and logical signals at the interface and the protocol for sequencing them.
A software interface may refer to a range of different types of interface at different "levels": an operating system may interface with pieces of hardware. Applications or programs running on the operating system may need to interact via streams, and in object oriented programs, objects within an application may need to interact via methods.

MAC Address

The MAC address is a unique value associated with a network adapter. MAC addresses are also known as hardware addresses or physical addresses. They uniquely identify an adapter on a LAN. MAC addresses are 12-digit hexadecimal numbers (48 bits in length). By convention, MAC addresses are usually written in one of the following two formats:
MM:MM:MM:SS: SS: SS

MM-MM-MM-SS-SS-SS
The first half of a MAC address contains the ID number of the adapter manufacturer. These IDs are regulated by an Internet standards body. The second half of a MAC address represents the serial number assigned to the adapter by the manufacturer.

Virtual Interface

A Virtual Interface or Virtual Network Interface (VIF) is an abstract virtualized representation of a computer network interface that may or may not correspond directly to a physical network interface.

5.3.1 Application Area

IP Configuration program is generally used by network/system administrator for

- To configure DHCP on the host machine
- To assign IPv4 and IPv6 on the end devices
- To define Primary and secondary DNS server
- To define the default gateway
- To have the information of network interface including interface name, interface Id, host name, IP address, multicast address, MTU, status, loopback and virtual address.

5.3.2 Working Mechanism of Network Interface Info

A Network Interface is where one endpoint of a socket is created that binds to a specific IP address. Network Interfaces are usually built-in to TCP/IP devices like routers, switches and computers or is added as external modules like PCI cards when expansion is required. Devices communicate and connect to each other through their hardware interface. Physically this takes the form of RJ-45 ports connected through cabling such as UTP or transmission media like fiber. Devices also employ software Interfaces such as the loopback or software network interfaces bridging virtual machines to local area network or NAT server.

Programmatically, Java provides the NetworkInterface class for accessing the attributes of switching between network interfaces. In addition to the NetworkInterface class, Java provides another class known as InetAddress which can be used to return the properties of an IP address bound to an Interface.getByInetAddress() specifies IP address of interface, getByName() specifies name of interface, getInetAddress() returns an enumeration of InetAddress objects, getInterfaceAddresses() returns InterfaceAddress instances, getNetworkInterfaces() returns the list of interfaces on host, getParent() returns parent of interface, getSubInterfaces() returns subinterfaces of NetworkInterface, isUP() returns true if interface is up and false if not, isLoopback() returns true if interface is loopback, isPointToPoint() returns true if interface is point to point, isVirtual() returns true if Interface is virtual, supportsMulticast() returns true if interface supports multicast, getHardwareAddress() returns MAC address and getMTU() returns maximum transmission unit.[4] Below is the function DisplayInterfaceInfo which output the necessary information for network interface:

```
public void DisplayInterfaceInfo(int INT_NUM) {
        try {
            ALL_INTERFACE = NetworkInterface.getNetworkInterfaces();
            NETINIT = Collections.list(ALL_INTERFACE).get(INT_NUM);
            InterfaceNumber_Box.setText(Integer.toString(INT_NUM+ 1));
            InterfaceName_Box.setText(NETINIT.getDisplayName());
            InterfaceID_Box.setText(NETINIT.getName());
            MAC_BOX.setText(Arrays.toString(NETINIT.getHardwareAddress()));
            ALL_ADDRESSES = NETINIT.getInetAddresses();
            for (InetAddress X : Collections.list(ALL_ADDRESSES)) {
                IP_BOX.setText(X.getHostAddress());
                HostName_Box.setText(X.getHostName());
            }
            MTU_Box.setText(Integer.toString(NETINIT.getMTU()));
            String STATUS;
            if (NETINIT.isUp()) {
                STATUS = "Up!";
            } else {
                STATUS = "Down";
            }
            Status_Box.setText(STATUS);
```

```
        String POINTTOPOINT;
        if (NETINIT.isPointToPoint()) {
            POINTTOPOINT = "Yes!";
        } else {
            POINTTOPOINT = "No!";
        }
        PointToPoint_Box.setText(POINTTOPOINT);

        String MULTICAST;
        if (NETINIT.supportsMulticast()) {
            MULTICAST = "Yes!";
        } else {
            MULTICAST = "No!";
        }
        Multicast_Box.setText(MULTICAST);
        String LOOPBACK;
        if (NETINIT.isLoopback()) {
            LOOPBACK = "Yes!";
        } else {
            LOOPBACK = "No!";

        }
        Loopback_Box.setText(LOOPBACK);
        String ISVIRTUAL;
        if (NETINIT.isVirtual()) {
            ISVIRTUAL = "Yes!";

        } else {
            ISVIRTUAL = "No!";

        }
        Virtual_Box.setText(ISVIRTUAL);
    } catch (SocketException x) {
    }
}
```

5.3.4 Output

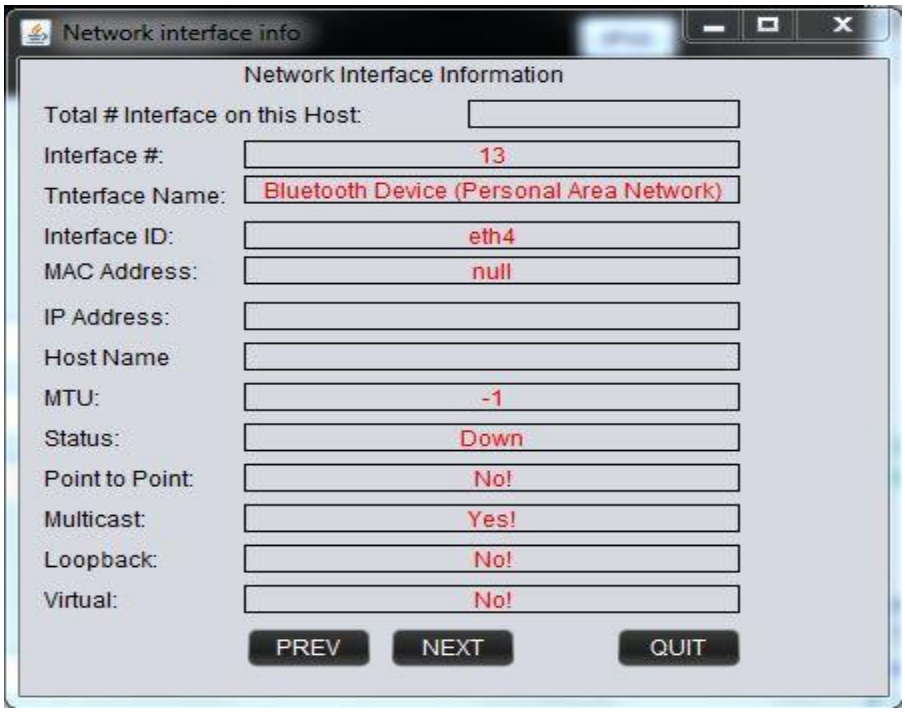

Fig 7: Output of Network interface info program

5.3.5 Working Mechanism of IP Configuration Program

This program helps to configure DHCP, DNS and default gateway in IPv4 and IPv6 on both windows and Linux. In Linux for IPv4 DNS configuration we simply need to add its IP address to the file /etc/resolv.conf. In order to configure DHCP address we will need to edit the **'interfaces'** file **/etc/network/interfaces**. Similarly for IPv6 the DHCPv6 server configuration file can be found at /etc/dhcp/dhcpd6.conf. The sample server configuration file can be found at /usr/share/doc/dhcp-<version>/dhcpd6.conf.sample.[5] For DHCP configuration in IPv6 following command is executed:

Command = "ifconfig " + nameinterface + " add " + ipValue + "/" + netmaskValue + " pointtopoint " + gatewayValue;

In windows for IPv4 to enable DHCP following command is run.[7]

```
netsh interface ip set address name="Local Area Connection" dhcp
```

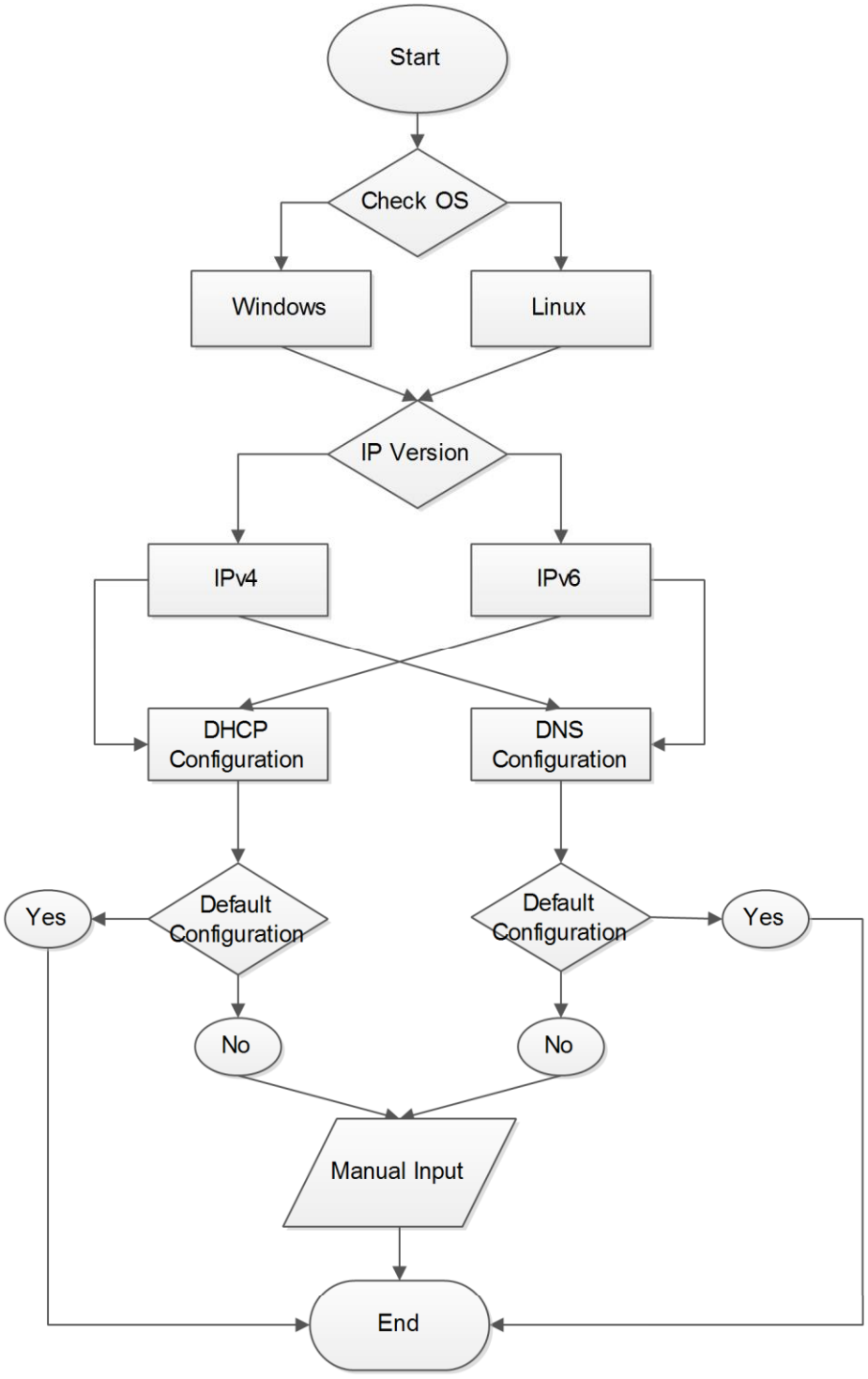

Fig 8: Flow-chart of IP configuration program

There are two commands for DNS since administrators typically configure a primary and secondary DNS server.

For the primary DNS

netsh interface ip set dns name="Local Area Connection" static ip address

For the secondary

netsh interface ip add dns name="Local Area Connection" ip address index=2

To configure the computer to use DNS from DHCP

netsh interface ip set dnsservers name="Local Area Connection" source=dhcp

For IPv6 in windows for DHCP configuration following command is run:

For wlan0

command1 = "netsh interface ipv6 set address " + "\"" + nameinterface + "\" " + ipValue + "/" + netmaskValue;

command2 = "netsh interface ipv6 add route ::/0 interface=" + "\"" + nameinterface + "\" " + gatewayValue;

For eth0

command1 = "netsh interface ipv6 set address "+ "\"" + nameinterface + "\" " + ipValue + "/" + netmaskValue;

command2 = "netsh interface ipv6 add route ::/0 interface=" + "\"" + nameinterface + "\" " + gatewayValue;

Similarly for DNS configuration following command is run:

For wlan0

command1 = "netsh interface ipv6 add dns \"" + nameinterface + "\" " + dnsone + " index=1";

command2 = "netsh interface ipv6 add dns \"" + nameinterface + "\" " + dnstwo + " index=2";

For eth0

command1 = "netsh interface ipv6 add dns \"" + nameinterface + "\" " + dnsone + " index=1";

command2 = "netsh interface ipv6 add dns \"" + nameinterface + "\" " + dnstwo + " index=2";

5.3.6 Output

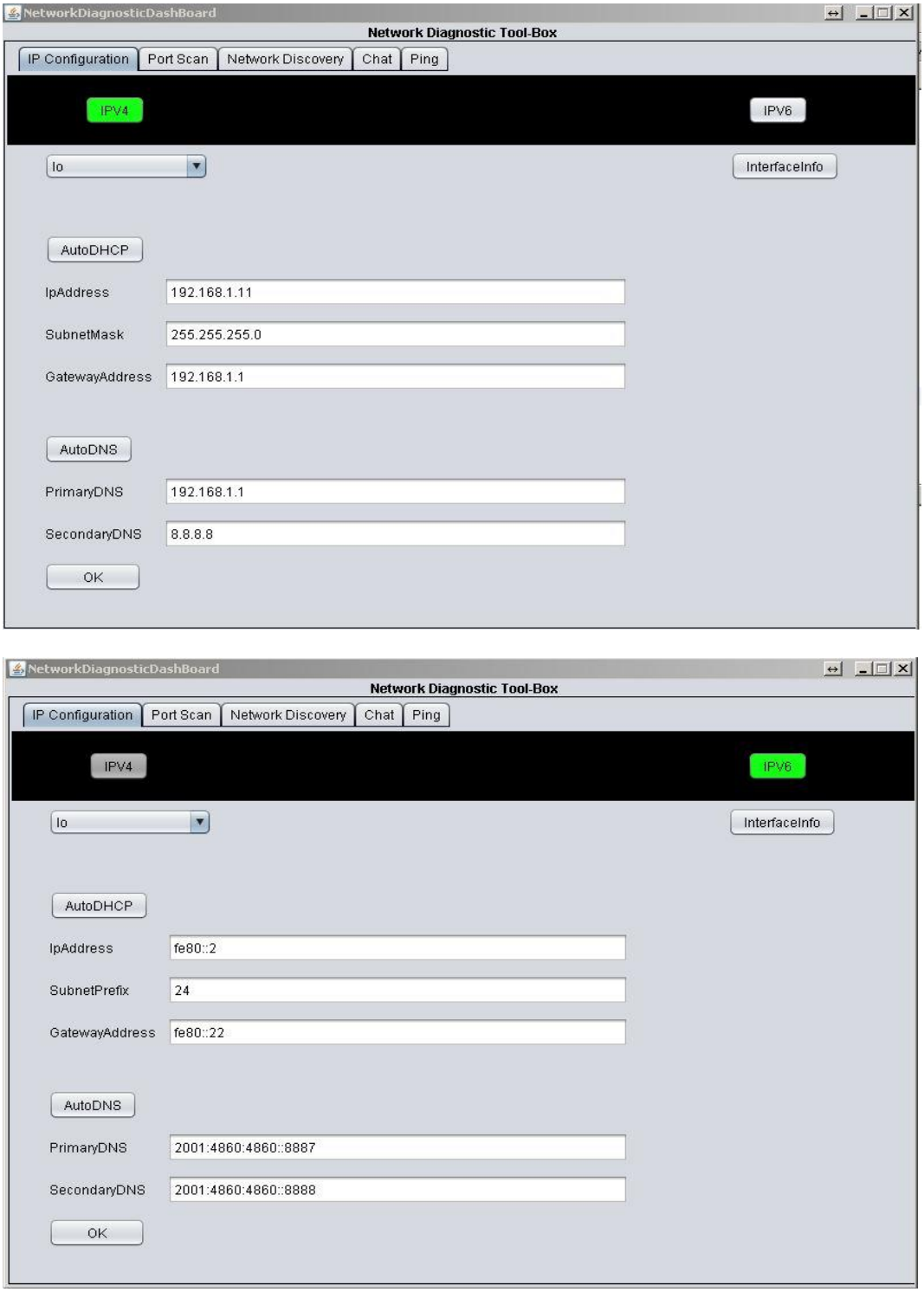

Fig9: Interface of IP Configuration

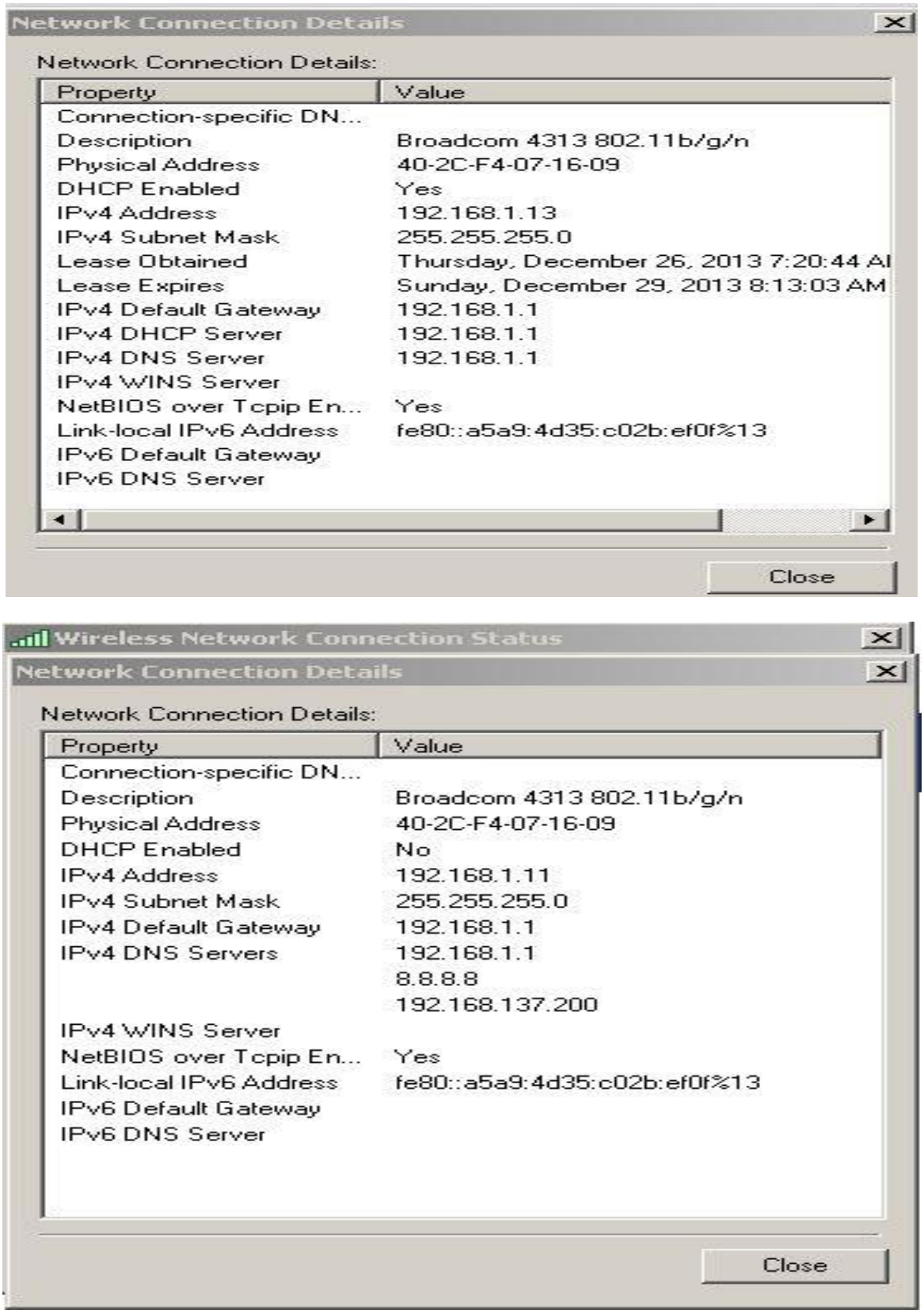

Fig 10: Output of IP Configuration program

5.4 LAN CHAT

As the world starts moving faster, we all want to catch up with the latest technologies. Indeed, investing in faster, more reliable and less human-dependent technologies, is a great opportunity to win the race and beat the competition.

Common means of communication, such as verbal instructions, phone conversations and e-mail messaging, are far from perfection when it comes to delivering short messages to multiple individuals. Internet-based instant messaging solutions are good but they rely on third-party services which makes the entire system unreliable in question. The best solution shall be an instant messaging (IM) software working in local network. [6]

5.4.1 Application Area

The program LAN Chat is designed to support exchanging messages in the regime of real time in a small office, in an educational institution, in general in any office with local networks. This unique software affords an opportunity for users of the local networks to communicate easily. LAN Chat requires does not require connecting to the Internet. The program needs only a simple one-range network. The user can take complete control over the program and there's an opportunity to make all kinds of settings to his own liking. This project is an example of a chat server. It is made up of two applications the client application, which runs on the user's PC and server application, which runs on any PC on the network. To start chatting client should get connected to server.

5.4.2 Working Mechanism of LAN Chat Program

There are two GUI Programs. One Server program and second Client program. Server Program opens port and can accepts up to 100 clients.

Client program's start button will make client thread to listen on client socket. If any message is written to the socket then it will fetches message using DatainputStream and then it show fetched messages to the Client message showing place. Send button of

client Program will write out messages typed by user in client message writing place into the socket.

Finally server listens messages from socket and again broadcasts to the client. So that all client listener thread in the client program again can fetch that data.

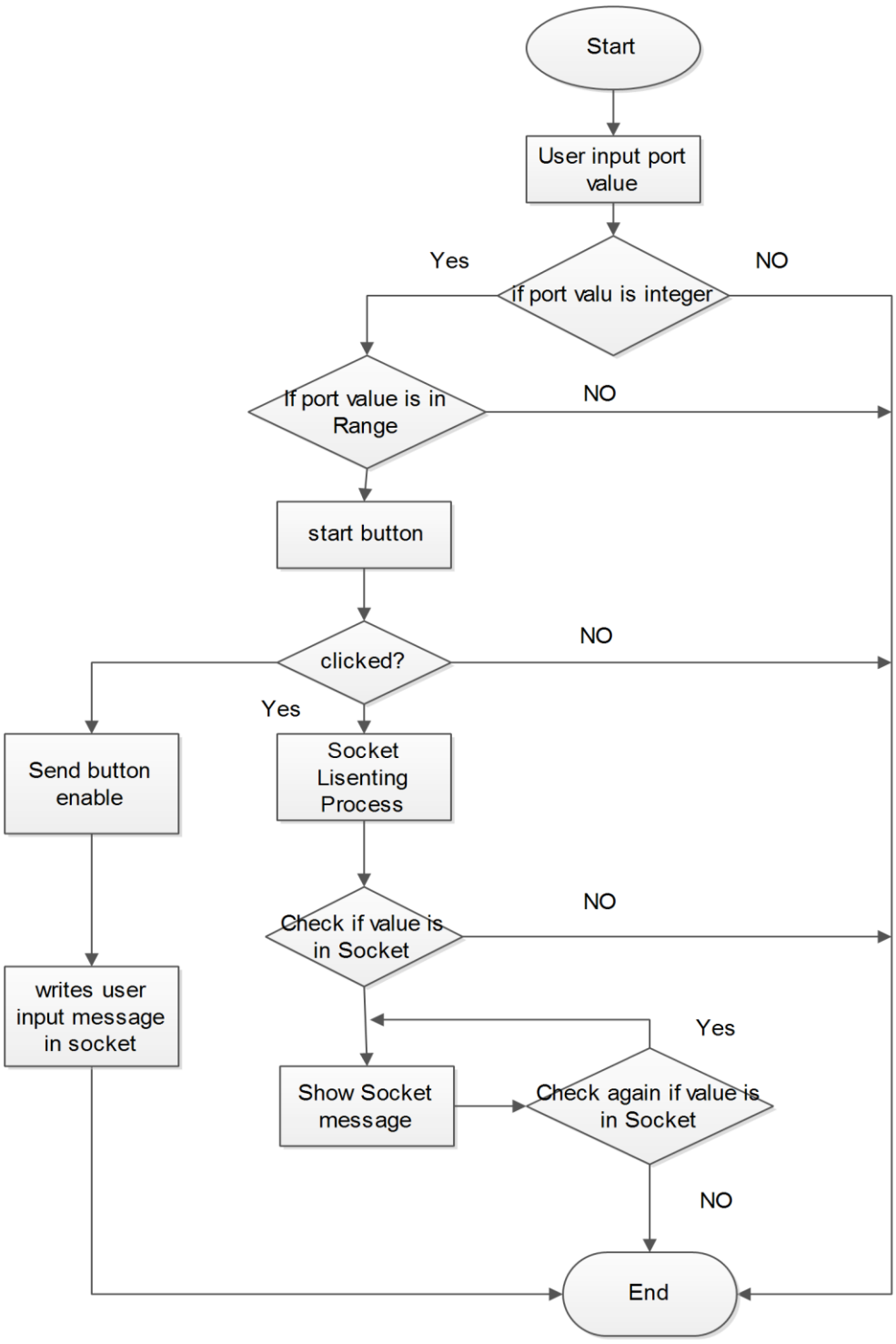

Fig11: Flow Chart of LAN Server Program

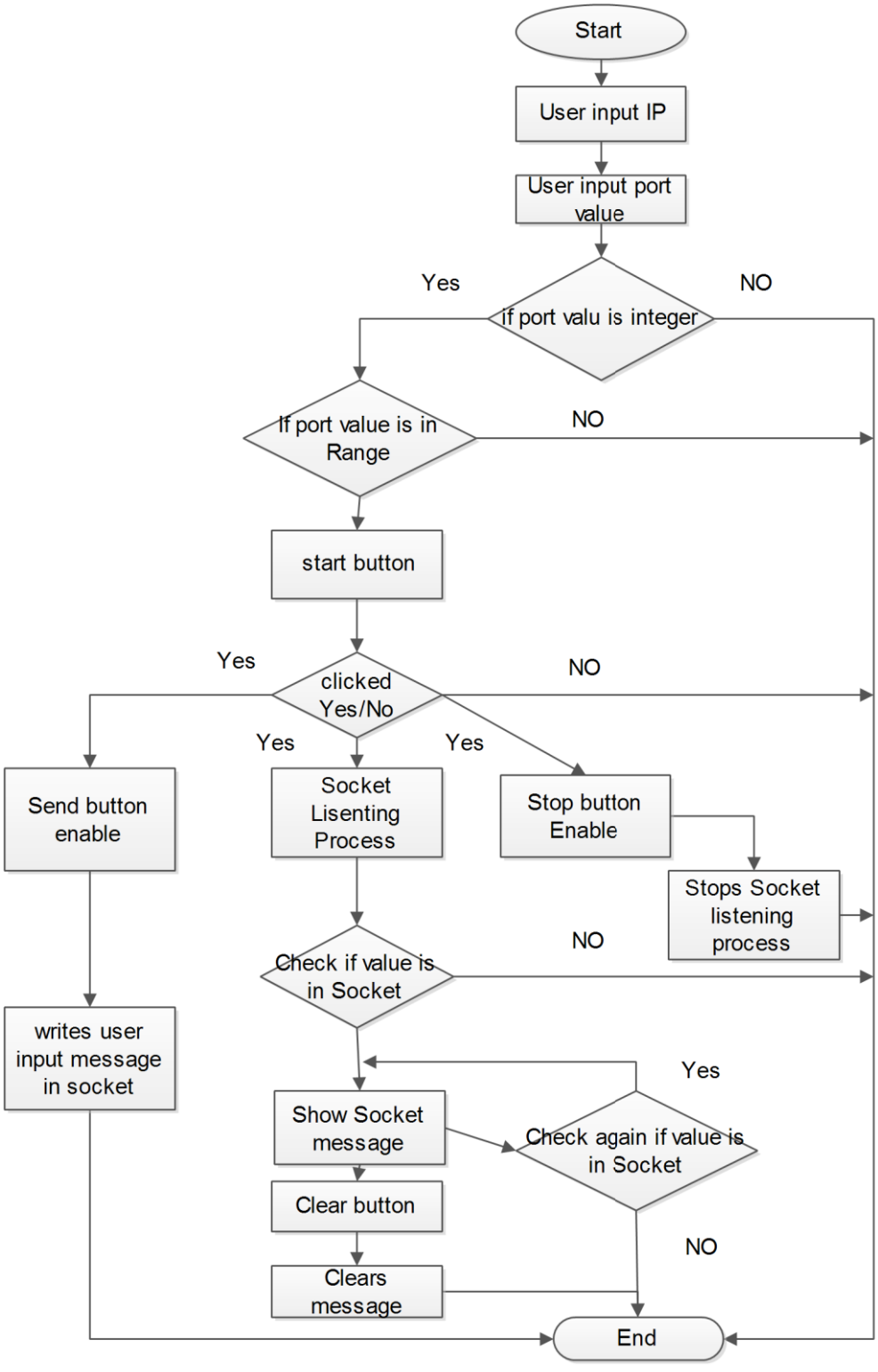

Fig 12: Flow Chart of LAN Client Program

There is SocketListener function in both server and client program which listens if message is contained by the socket or not? If socket contains message then it displays in corresponding textarea as in Client program,

```
class ClientSocketWithListener extends Thread{

private BufferedReader in;

private PrintWriter out;

private String msg;

ClientSocketWithListener(Socket clientsock) {

try {

in= new BufferedReader(newInputStreamReader(clientsock.getInputStream()));

out = new PrintWriter(clientsock.getOutputStream(), true);

} catch (IOException e) {

System.err.println(e);

}

}

@Override

public void run() {

while (true) {

try {

while ((msg = in.readLine()) != null) {

Chat.msgshowingAreabodyPanellChat.append("From Server :: "+msg);

}

} catch (IOException e) {

System.err.println(e);

}

}    }}
```

Now creating this class's instance and running start() function from this stance will show messages of socket in showing area.

There is send button in both program. Clicking on this button message written by the user is written to the socket which is displayed in other side by socket listener thread function.

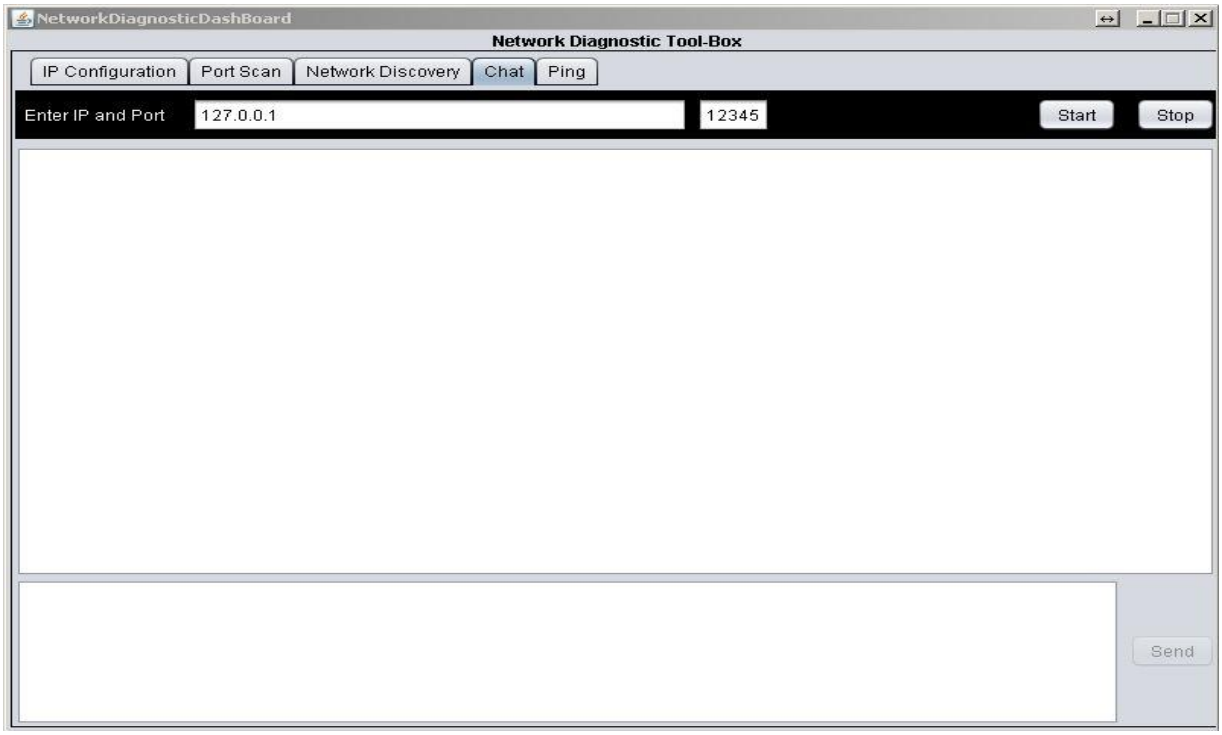

Fig 13: Interface of LAN Chat Client

5.5 NETWORK DISCOVERY

Network Discovery is a very fast and robust application program for finding the connected devices especially via router. It is intended for both system administrators and general users to monitor and manage their networks. Powered with multi-thread scan technology, this program can scan hundreds computers per second.[3] It simply pings each IP address to check if it's alive or not.

5.5.1 Application Area

This Network Discovery program is used by network/system administrator to find the ip address of machines connected in a LAN via router so that they monitor the performance of their network and apply security as well as administrative restrictions.

5.5.2 Working Mechanism of Network Discovery

We have pinged each IPs in the Defined Network's subnet, Where the user enters subnet value from the network 192.168.1.0 the user inputs 192.168.1.X and subnet value then generated with 192.168.1 leaving behind X part and index I is run from 1 to 254. And each final string is assigned with host variable. Now each host is pinged inside Thread. Each thread prints 5 lines to make user more interactive with our project. And ping's third returned line will give us our network LAN's all host.

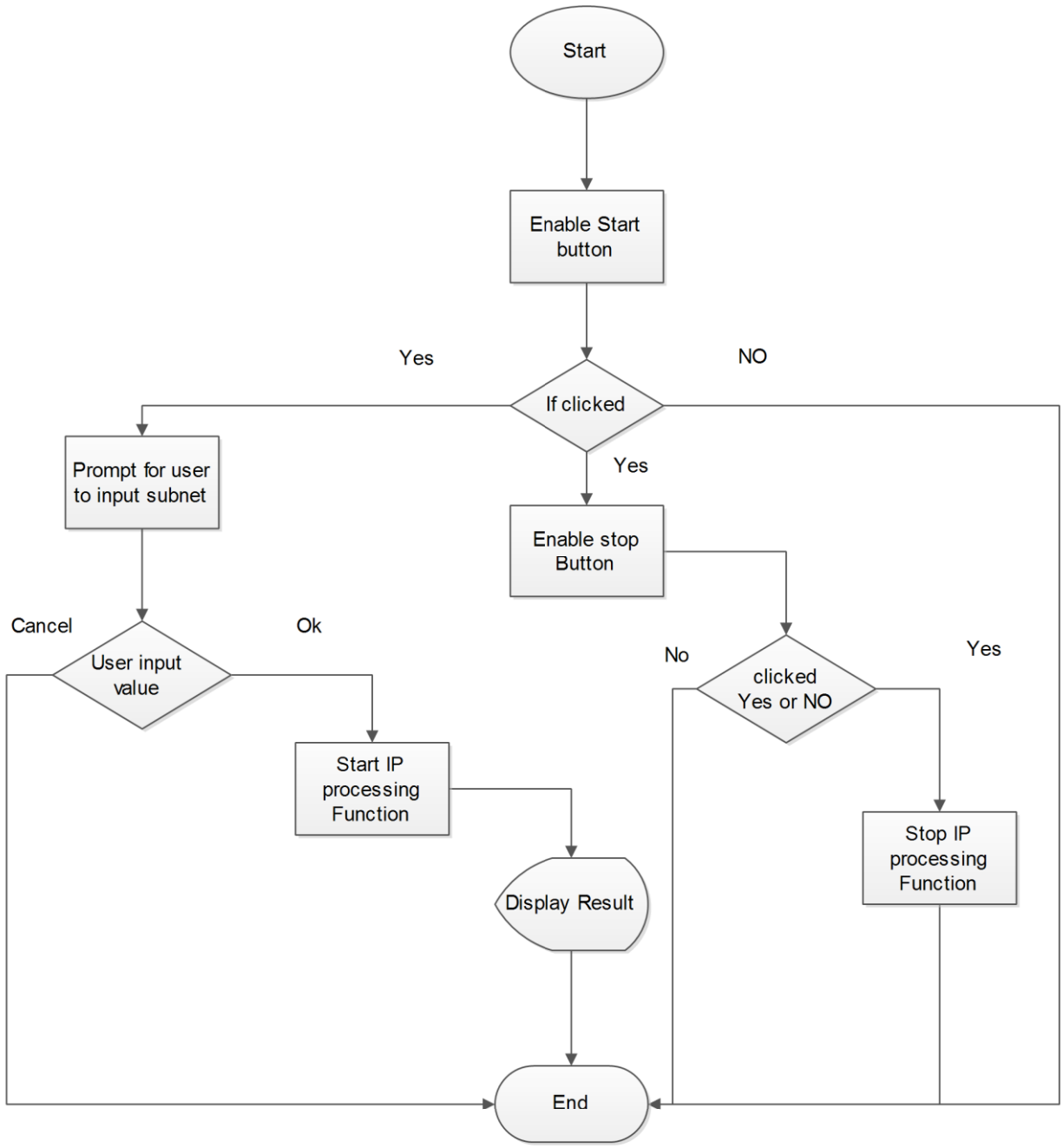

Fig 14: Flow Chart of Network Discovery

On clicking start button of Network Discovery will scans all ip in user inputted subnet and displays in table. On scanning process, We implemented many numbers of threads. Each threads scans for five ip and prints out these records in Table. So for class c there will be total Math.ceil(254/5)+1 number of threads. We implemented threads concept for

each five ips to make our program more interactive to the users. Our scanning function as,

```
    noofthreads = (254 / 5) + 1;
Runnable r = null;//[] = new Runnable[noofthreads+1];
timeout = 1000;
t = new Thread[noofthreads + 1];
int k = 1;
for (int i = 1; i <= noofthreads; i += 5) {
    threadObject tObj = new threadObject(model, subnetvalue1, i, i + 5, timeout);
    tObj.start();
}
```

And our threadObject class as,

```
class threadObject extends Thread {

    private final DefaultTableModel model;
    private final int timeout;
    private final String subnet;
    private final int range_start, range_end;

    public threadObject(DefaultTableModel model, String subnet, int range_start, int range_end, int timeout) {
        this.model = model;
        this.timeout = timeout;
        this.subnet = subnet;
        this.range_start = range_start;
        this.range_end = range_end;
```

```
    }

    @Override
    public void run() {
        for (int i = range_start; i < range_end; i++) {
            String host;
            host = subnet + "." + i;
            Process process;

            Runtime runTime = Runtime.getRuntime();
            try {
                process = runTime.exec("ping " + host);
//                if  ping then add into table
                BufferedReader br = new BufferedReader(new InputStreamReader(process.getInputStream()));
                br.readLine();
                br.readLine();
                String output = br.readLine();
                if ((!output.contains(" Destination host unreachable.")) && (!output.contains("Request timed out."))) {
                    model.addRow(new Object[]{host});
                }
            } catch (IOException ex) {
            }
        }
    }}
```

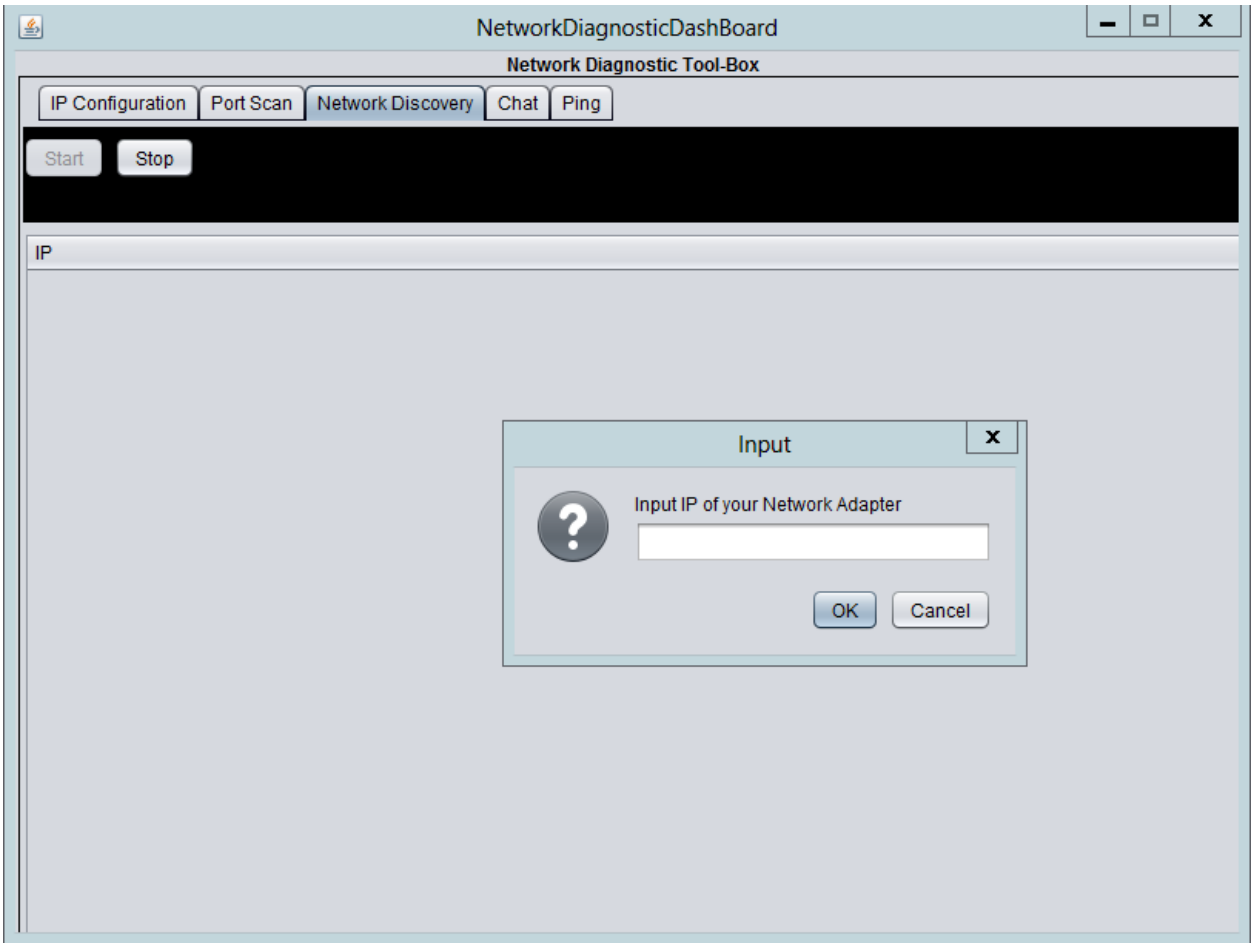

Fig 15: Interface for Network Discovery program

5.5.3 Output

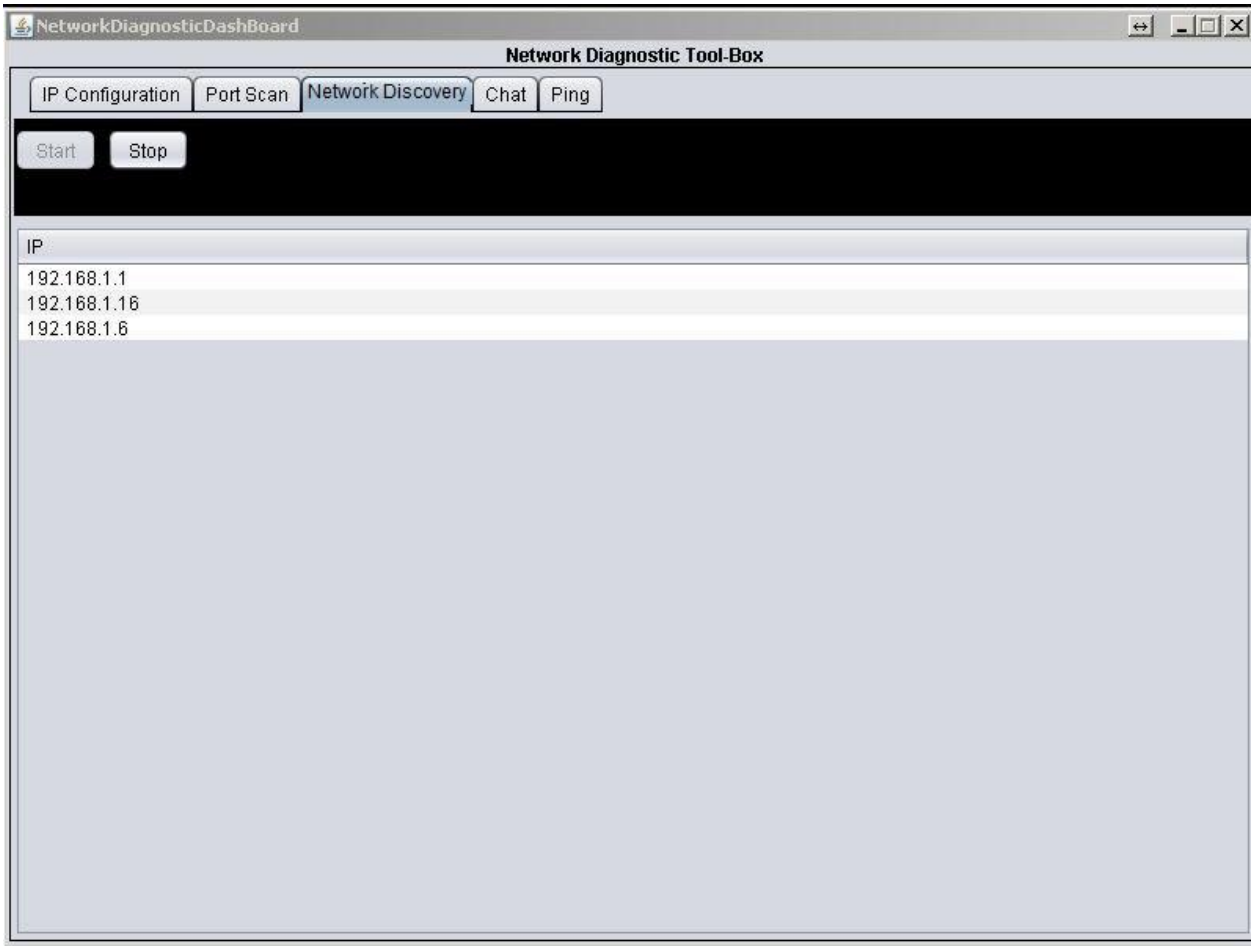

Fig 16: Output of Network Discovery Program

Chapter-6

TESTING AND EXPERIMENTAL EVALUATION

When the system was tested with varying sample inputs few inconsistencies were seen. Starting with the setup interface we began our testing process. Output needed to be displayed was varied. First we were not able to change the required IP according to user input .There was difficulty on selecting the required interface. Therefore by the addition of network_interface_info we gathered the information about available interface their MAC, IP address as well their status. Similarly in ping program at first we only displayed the reachability information but due to the suggestion given by the reviewer we included the ping statistics. A lot of time has been allocated in Network Discovery and LAN chat to validate their result and to provide maximum performance.

After the NETWORK DIAGNOSTIC TOOL is tested with varying samples by different reviewer using different testing mechanism, the system is now almost 100% efficient and generates optimal output. This project is tested and evaluated by different person and the inconsistencies and errors in the program are eliminated. Therefore this project is ready use for administrative as well as general purpose

Chapter-7

PROBLEMS ENCOUNTERED

Difficulties and uncertainties have always been a fact arising in a system to be built as the system is designed to be dynamic one. This added dynamically will definitely improve the quality of the system, but during the latter stages of development do introduce some constraints that occur due to various reasons. Two of the major issues discussed are as follows:

Performance/Behavior Issue

The application has to depend on the operating system architecture and commands executed by them. So it became difficult to understand the internal architecture and working principle of proprietary system.

Management/Technical Issue

Limited time for the project development does slow down the speed of the work among the team. It becomes quite tough to complete all documentation, software creation and enhancement within the allotted span of time. There exist a lot of ideas but one cannot implement them due to time constraint. Due to time limitation there were many other features to be added to the project.

Chapter-8

PROJECT SCHEDULING

	Task	Start	End	Dur
	Network Diagnostic dashboard	7/1/13	12/17/13	117
1	Identification and selection	7/1/13	7/8/13	5
2	Initialization and Planning	7/8/13	7/12/13	5
3	System Design	7/12/13	8/6/13	18
4	Interface Design	8/7/13	8/30/13	18
5	Coding	8/30/13	11/14/13	52
6	Testing	11/15/13	11/25/13	7
7	Documentation	11/26/13	12/9/13	9
8	Maintenance	12/10/13	12/13/13	4
9	Final review and Editing	12/14/13	12/17/13	2

Fig17: Project Scheduling

Chapter- 9

RECOMMENDATION AND CONCLUSION

Recommendation

We sincerely recommend the upcoming batches to work in this type of interesting and challenging projects and know what's happening around the networking areas and implement such system in their own field. This project would make them familiar with the modern technologies and to cope with the most rising, interesting and competitive of computer science.

This type of projects will definitely provide an exposure the student to the current industrial requirement and demands. They will enjoy the art of learning new things applicable to the project that would be of great boon for their bright future.

Conclusion

We conclude that we developed this project Network Diagnostic Tool. This project has given us a chance to put into practice of the software development concepts that we have only been studied in theories. We are trained to work effectively as part of team, interacting with users, developing specification and documents developing prototypes and improving our writing and oral presentation skills.

Throughout this project, we have developed Network Diagnostic Tool. The system is proposed during title selection of projects. During this project, we expect a chance to sharpen our skills on technical, analysis or interpersonal skills. The physical visual system is developed using java programming language. While developing the system, we encountered a lot of problems in java coding. There were errors in programming and coding. We found references from websites and related teachers. Through the process,

we have expanded our knowledge in JAVA programming. We have also known other various networks connectivity between java programming and network interfaces.

Finally the project is completed with the effort of all group members working together. A good teamwork is contributed in this project where group members were co-operated and well hold their responsibility**.**

FUTURE WORK

In this NETWORK DIAGNOSTIC TOOL we have developed user friendly interface and included application program that handle simple administrative tasks. This project covers the fundamental programs frequently used by network/system administrator with highly interactive interface. Even though, there are many application that are most frequently used but are not developed and integrated into this project like trace route, packet capturing, illegal access of application and routing mechanism.

So in future we are going to add those functionalities on our project. We will embed the functionalities for defining the default route as well as user configured route. Similarly we will include a program that defines the security constraints and determines whether the authorized user is accessing the system's application or not and try to capture their log. We will also enhance the chat application so that in addition to simple text communication we can transfer images, files and also include voice communication using VOIP.

BIBLIOGRAPHY

Keneth E. Kendall, Julie E. Kendall "System Analysis and Design" -Pearson Education 2009

Todd Lamme "Cisco Certified Network Associate" -BPB publication 2006

Cay S. Horstmann , Gary Cornell "Core Java-Volume I" -Pearson Education 2012

Cay S. Horstmann, Gary Cornell "Core Java-Volume II" -Pearson Education 2012

James F. Kurose, Keith W. Ross "Computer Networking A Top Down Approach" -Pearson

Education 2012

Elliotte Rusty Harold "Java Network Programming" -O'RELLY 2013

John J. Amoss, Daniel Minoli "Handbook of IPv4 to IPv6 Transition" -Auerbach publication 2013

Olaf Kirch & Terry Dawson “Linux Network Administrator's Guide, 2nd Edition"- O'REILLY 2002

REFERENCES

[1] http://www.wikipedia.org

[2] http://www.cisco.com/en/US/docs/security/fwsm/fwsm32/asdm52f/user/guide/dhcp_dns.html

[3] http://stackoverflow.com

[4] http://www.cs.princeton.edu/courses/archive/spr05/cos126/cmd-prompt.html

[5] http://www.linuxhelp.net/

[6] http://www.computerhope.com/unix.htm

[7] http://www.windowsnetworking.com/